



ETEC PROFESSOR CAMARGO ARANHA – SÃO PAULO
MTEC – PROGRAMA NOVOTEC INTEGRADO
DESENVOLVIMENTO DE SISTEMAS INTEGRADO AO ENSINO MÉDIO

DISCPLINA: Qualidade e Teste de Software

Professor: Dr. Luiz Lima e Davi Vilar

01000011
01101111
01101101
01110000
01110101
01110100
01100001
11100111
11100011
01101111

Qualidade e Teste de SW

Uma imersão ao mundo da Qualidade

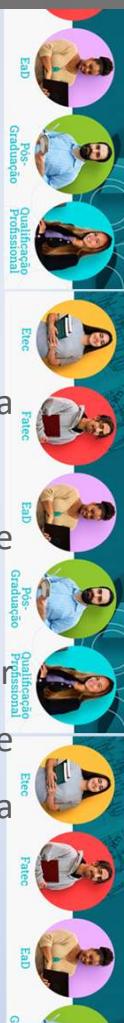
Apresentação do Professor

Luiz Lima
Professor



✉ aula.prof.luiz@gmail.com

- Técnico em Eletrônica (1989)
- Graduação: Tecnologia em Processamento de Dados (2001).
- Especialização: Formação em Educação à Distância (2011).
- Mestrado: Engenharia de Produção - Sistema Especialista AITOD baseado na Lógica Paraconsistente Anotada Evidencial Et (2018).
- Doutorado: Engenharia de Produção - Auxílio Computadorizado no Diagnósticos Precoce de Câncer de Próstata por meio de Redes Neurais Artificiais Paraconsistentes (2022).
- Atualmente: pesquisador do grupo "IA Responsável" em nosso país, criado pelo professor Doutor Virgilio Almeida (titular da Cátedra Oscar Sala IEA-USP). E participante do Grupo de Pesquisa no CNPq em Organização, Logística, Gestão, Participação e Controle no Programa de Alimentação Escolar com uso IA/LP/Metaverso/LGPD.
- Professor em outras Disciplinas de DS, ADM, GESTÃO,...



EXPERIÊNCIA ACADÊMICA



CNPq
Conselho Nacional de Desenvolvimento Científico e Tecnológico

Dados gerais | Formação | Atuação | Projetos | Produções | Inovação | Educação e Popularização de C & T | Eventos

Luiz Antonio de Lima

Endereço para acessar este CV: <http://lattes.cnpq.br/8893376836766464>

ID Lattes: **8893376836766464**

Última atualização do currículo em 08/02/2023

COLÉGIO PAN TERRA

ie A Institute of Advanced Studies of the University of São Paulo

Responsible AI in the Global South



Fatec



UNIP

Anhanguera

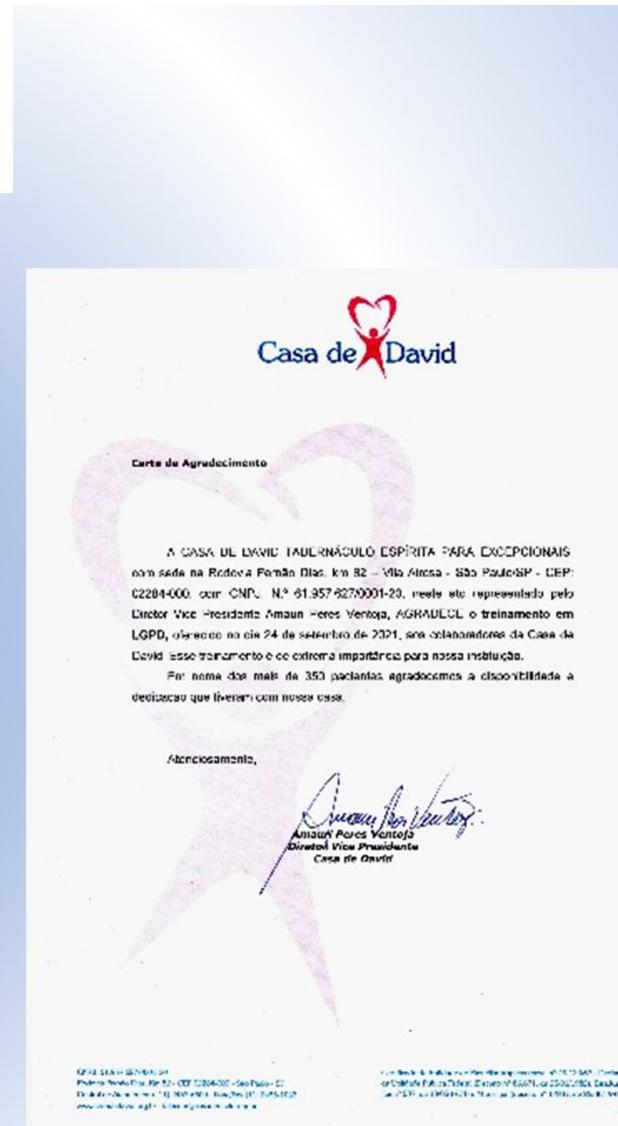
kroton
paixão por educar

cogna
EDUCAÇÃO

ānimA
EDUCAÇÃO

FIAP









Um pouco de artigos

TESE: Auxílio Computadorizado no Diagnósticos Precoce de Câncer de Próstata por meio de Redes Neurais Artificiais Paraconsistentes ([PDF](#))

DISSERTAÇÃO: Sistema Especialista AITOD baseado na Lógica Paraconsistente Anotada Evidencial Et ([PDF](#))

2022
PLATAFORMA
Sucupira
2018

<http://dx.doi.org/10.12819/2022.19.3.8>

SAKAMOTO, LILIAM SAYURI ; Abe, Jair Minoro ; LIMA, LUIZ ANTÔNIO DE ; SOUZA, NILSON AMADO DE ; SOUZA, JONATAS SANTOS DE . *Uso da Lógica Paraconsistente Anotada Evidencial Et em Sistemas de Logística*. Revista FSA (Faculdade Santo Agostinho), v. 19, p. 150-164, 2022.

<http://dx.doi.org/10.37118/ijdr.23962.02.2022>

INSUA, H. G. ; ABE, J. M. ; LIMA, L. A. **PRODUTIVIDADE DA POLÍCIA CIVIL DO ESTADO DE SÃO PAULO: UMA ANÁLISE**. INTERNATIONAL JOURNAL OF DEVELOPMENT RESEARCH, v. 12, p. 1-4, 2022.

2022
B2
revistafsa
Qualis-CAPES (A2) (BRAZIL)
<https://doi.org/10.37118> Crossref

<http://dx.doi.org/10.12819/2021.18.7.11>

LIMA, L. A. ; ABE, JAIR MINORO ; MARTINEZ, A. A. G. ; SOUZA, JONATAS SANTOS DE ; BERNARDINI, FLÁVIO AMADEU ; SOUZA, NILSON AMADO DE ; Sakamoto, Liliam Sayuri . *Study of Pann Components in Image Treatment for Medical Diagnostic Decision-Making*. REVISTA FSA (FACULDADE SANTO AGOSTINHO), v. 18, p. 173-186, 2021

SOUZA, JONATAS SANTOS DE ; ABE, JAIR MINORO ; SOUZA, NILSON AMADO DE ; LIMA, L. A. ; MARTINEZ, A. A. G. ; BERNARDINI, FLÁVIO AMADEU ; SOUZA, VAGNER PEREIRA DE ; Sakamoto, Liliam Sayuri . *O Uso do Arduino para Controlar um Sistema de Irrigação Baseado em Lógica Et*. REVISTA FSA (FACULDADE SANTO AGOSTINHO), v. 18, p. 199-216, 2021.

2021
B2
revistafsa
Educação
DOAJ
REDIB
CrossRef
University of São Paulo
Scielo

<http://dx.doi.org/10.1016/j.inpa.2020.10.003>

DE ALENCAR NÄÄS, IRENILZA; DA SILVA LIMA, NILSA DUARTE; GONÇALVES, RODRIGO FRANCO; ANTONIO DE LIMA, LUIZ ; UNGARO, HENRY; MINORO ABE, JAIR . *Lameness prediction in broiler chicken using a machine learning technique*. INFORMATION PROCESSING IN AGRICULTURE, v. 8, p. 409-418, 2021.

2021
A2
revistafsa
ScienceDirect



PROFISSÃO



CAPÍTULO 3

PERFIL PROFISSIONAL DE CONCLUSÃO

3^a SÉRIE

O TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS é o profissional que analisa e projeta sistemas. Constrói, documenta, realiza testes e mantém sistemas de informação. Utiliza ambientes de desenvolvimento e linguagens de programação específica. Modela, implementa e mantém bancos de dados.

ENSINO MÉDIO COM HABILITAÇÃO PROFISSIONAL DE TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

Ao longo da Educação Básica, as aprendizagens essenciais definidas na BNCC devem concorrer para assegurar aos estudantes o desenvolvimento de dez competências gerais, que consubstanciam, no âmbito pedagógico, os direitos de aprendizagem e desenvolvimento.



PROFISSÃO

PERFIL PROFISSIONAL DA QUALIFICAÇÃO

1ª SÉRIE

Qualificação Profissional Técnica de Nível Médio de **AUXILIAR EM DESENVOLVIMENTO DE SISTEMAS**

O **AUXILIAR EM DESENVOLVIMENTO DE SISTEMAS** é o profissional que desenvolve programas e auxilia na análise de sistemas e modelagem de bancos de dados.

ÁREA DE ATIVIDADES

- A – PROJETAR SISTEMAS DE INFORMAÇÃO
- B – DESENVOLVER SISTEMAS
- C – DESENVOLVER BANCO DE DADOS
- D – PESQUISAR E MANTER-SE ATUALIZADO EM RELAÇÃO A PRINCÍPIOS DA ÉTICA NAS RELAÇÕES DE TRABALHO

PROFISSÃO

MERCADO DE TRABALHO

- ❖ Empresas e departamentos de desenvolvimento de sistemas em organizações governamentais e não governamentais, podendo também atuar como profissional autônomo.

COMPETÊNCIAS PESSOAIS/SOCIOEMOCIONAIS

- ❖ Demonstrar ética profissional.
- ❖ Demonstrar autonomia intelectual.
- ❖ Evidenciar resiliência no desenvolvimento do trabalho.
- ❖ Demonstrar capacidade de lidar com situações novas e inusitadas.
- ❖ Demonstrar proatividade e iniciativa no desenvolvimento de atividades.
- ❖ Manter-se atualizado a respeito de novas tecnologias referentes à área de atuação.
- ❖ Apresentar argumentos logicamente encadeados a respeito de um determinado assunto.
- ❖ Demonstrar capacidade de adotar em tempo hábil a solução mais adequada entre possíveis alternativas.



PROFISSÃO

Ao concluir a **ENSINO MÉDIO COM HABILITAÇÃO PROFISSIONAL DE TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS**, o aluno deverá ter construído as seguintes competências:

- Documentar sistemas de informação.
- Analisar modelos de sistemas embarcados.
- Desenvolver aplicações com microcontroladores.
- Avaliar e selecionar técnicas de teste de *software*.
- Elaborar modelo de negócios para uma empresa de *software*.
- Desenvolver sistemas, implementando rotinas de segurança de dados.
- Avaliar as fontes e recursos necessários para o desenvolvimento de projetos.
- Avaliar a execução e os resultados obtidos de forma quantitativa e qualitativa.
- Analisar dados e informações obtidas de pesquisas empíricas e bibliográficas.
- Desenvolver serviços para o usuário, utilizando recursos dos dispositivos móveis.
- Desenvolver sistemas multicamadas, utilizando *framework* de desenvolvimento web.
- Utilizar princípios inovadores de Empreendedorismo na criação de projetos/startups de tecnologia.
- Planejar as fases de execução de projetos com base na natureza e na complexidade das atividades.
- Configurar os principais serviços de redes de comunicação de dados e internet para o desenvolvimento de sistemas.
- Propor soluções parametrizadas por viabilidade técnica e econômica aos problemas identificados no âmbito da área profissional.

ATRIBUIÇÕES E RESPONSABILIDADES

3ª SÉRIE

- ❖ Desenvolver sistemas embarcados.
- ❖ Planejar projeto para sistemas computacionais.
- ❖ Implementar rotinas de segurança da informação.
- ❖ Desenvolver projetos para sistemas computacionais.
- ❖ Elaborar e manter sistemas de informação para Web.
- ❖ Testar softwares para melhoria da qualidade de sistemas.
- ❖ Utilizar protocolos de redes e internet para comunicação de dados.
- ❖ Elaborar registros e planilhas de acompanhamento e controle das atividades.

ÁREA DE ATIVIDADES

A – PROJETAR SISTEMAS DE INFORMAÇÃO

B – DESENVOLVER SISTEMAS

C – DESENVOLVER BANCO DE DADOS

D – PESQUISAR E MANTER-SE ATUALIZADO EM RELAÇÃO A PRINCÍPIOS DA ÉTICA NAS RELAÇÕES DE TRABALHO





ACRITÉRIOS DE VALIAÇÃO

CAPÍTULO 6 CRITÉRIOS DE AVALIAÇÃO DE APRENDIZAGEM

Menção	Conceito	Definição Operacional
MB	Muito Bom	O aluno obteve excelente desempenho no desenvolvimento das competências do componente curricular no período.
B	Bom	O aluno obteve bom desempenho no desenvolvimento das competências do componente curricular no período.
R	Regular	O aluno obteve desempenho regular no desenvolvimento das competências do componente curricular no período.
I	Insatisfatório	O aluno obteve desempenho insatisfatório no desenvolvimento das competências do componente curricular no período.

A avaliação, elemento fundamental para acompanhamento e redirecionamento do processo de desenvolvimento de competências, estará voltada para a construção dos perfis de conclusão estabelecidos para as diferentes habilitações profissionais e as respectivas qualificações previstas.

Constitui-se num processo contínuo e permanente com a utilização de instrumentos diversificados – textos, provas, relatórios, autoavaliação, roteiros, pesquisas, portfólio, projetos, entre outros – que permitam analisar de forma ampla o desenvolvimento de competências em diferentes indivíduos e em diferentes situações de aprendizagem.





ACRITÉRIOS DE VALIAÇÃO

O caráter diagnóstico dessa avaliação permite subsidiar as decisões dos Conselhos de Classe e das Comissões de Professores acerca dos processos regimentalmente previstos de:

- classificação;
- reclassificação;
- aproveitamento de estudos.

aproveitamento insatisfatório,

Permite também orientar/reorientar os processos de:

- recuperação contínua;
- progressão parcial.

o instituto da **Reclassificação** permite ao aluno a matrícula em série diversa daquela em que está classificado, expressa em parecer elaborado por Comissão de Professores, fundamentada nos resultados de diferentes avaliações realizadas.

o instituto de **Aproveitamento de Estudos** permite reconhecer como válidas as competências desenvolvidas em outros cursos – dentro do sistema formal ou informal de ensino, dentro da formação inicial e continuada de trabalhadores, etapas ou módulos das habilitações profissionais de nível técnico ou as adquiridas no trabalho.

o instituto da **Progressão Parcial** cria condições para que os alunos com menção insatisfatória em até três componentes curriculares possam, concomitantemente, cursar a série seguinte, ouvido o Conselho de Classe.

Menção	Conceito	Definição Operacional
MB	Muito Bom	O aluno obteve excelente desempenho no desenvolvimento das competências do componente curricular no período.
B	Bom	O aluno obteve bom desempenho no desenvolvimento das competências do componente curricular no período.
R	Regular	O aluno obteve desempenho regular no desenvolvimento das competências do componente curricular no período.
I	Insatisfatório	O aluno obteve desempenho insatisfatório no desenvolvimento das competências do componente curricular no período.



AVALIAÇÃO

Menção	Conceito	Definição Operacional
MB	Muito Bom	O aluno obteve excelente desempenho no desenvolvimento das competências do componente curricular no período.
B	Bom	O aluno obteve bom desempenho no desenvolvimento das competências do componente curricular no período.
R	Regular	O aluno obteve desempenho regular no desenvolvimento das competências do componente curricular no período.
I	Insatisfatório	O aluno obteve desempenho insatisfatório no desenvolvimento das competências do componente curricular no período.

Será considerado concluinte do curso ou classificado para a série seguinte o aluno que obtiver aproveitamento suficiente **para promoção – MB, B ou R** – e a frequência mínima estabelecida.

A **frequência mínima** exigida será de **75% (setenta e cinco)** do total das horas efetivamente trabalhadas pela escola, calculada sobre a totalidade dos componentes curriculares de cada série e terá apuração independente do aproveitamento.





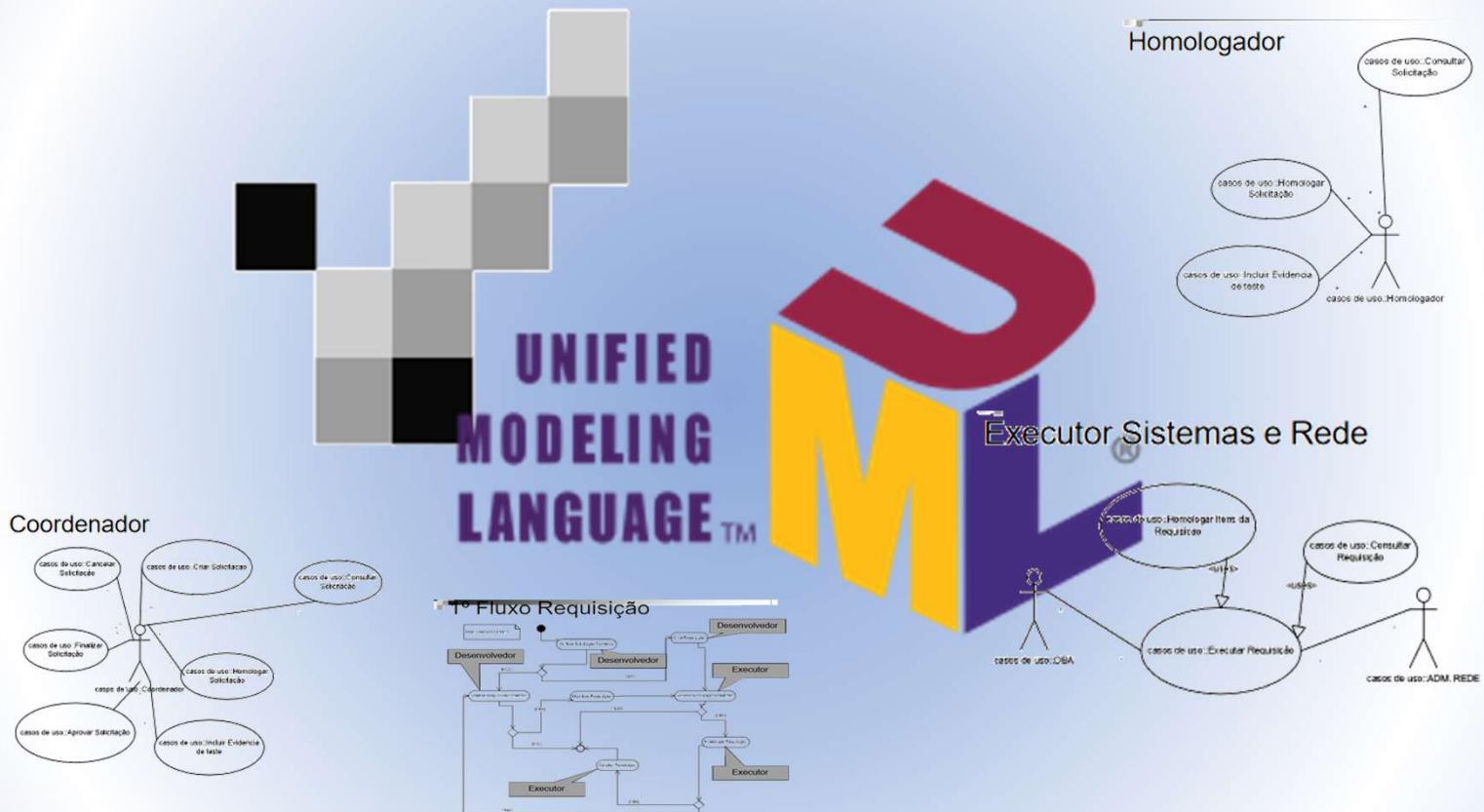
BIBLIOGRAFIA

- [1] DELAMARO, M. E.; MALDONADO, J. C.; JINO, M. Introdução ao Teste de Software. Campus, 2007.
- [2] KOSCIANSKI, A; SOARES, M S. Qualidade de Software. Novatec, 2007.
- [3] SILVA, Ivan Jose de Mecenas; OLIVEIRA, Viviane. Qualidade em Software. Alta Books, 2005.

Bibliografia complementar:

- [4] GOUVEIA, D; SARACEVIC, F; BOCARSLEY, J B. Software test engineering with IBM Rational Functional Tester: the definitive resource. Prentice Hall, 2009.
- [5] Eletrônicos: MPS.BR. Para obter guias e demais informações. Disponível em: [. Acesso em: 28 mai 2009.](#)





<https://www.omg.org/spec/UML/2.5.1/>

O documento contém a especificação atual da UML 2.5.1.

Linguagem de Modelagem Unificada



Linguagem de modelagem unificada (UML)

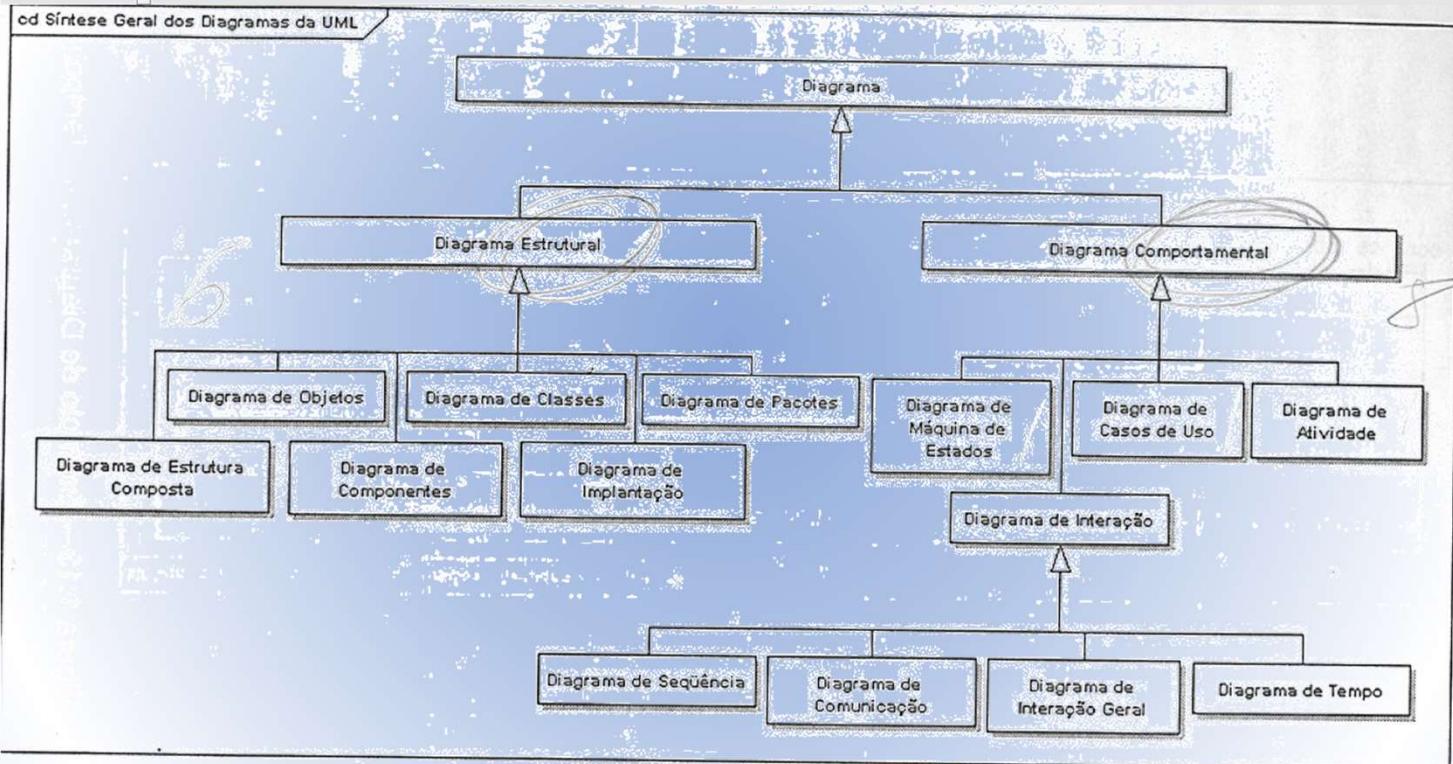
1.4 A Linguagem de Modelagem Unificada (UML)

A construção da UML teve muitos contribuintes, mas os principais atores no processo foram Grady Booch, James Rumbaugh e Ivar Jacobson. Esses três pesquisadores são frequentemente chamados de “os três amigos”. No processo de definição da UML, procurou-se aproveitar o melhor das características das notações preexistentes, principalmente das técnicas propostas anteriormente pelos três amigos (essas técnicas eram conhecidas pelos nomes *Booch Method*, *OMT* e *OOSE*). A notação definida para a UML é uma união de diversas notações preexistentes, com alguns elementos removidos e outros elementos adicionados com o objetivo de torná-la mais expressiva.



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







1. Diagrama de Caso de Uso:

Usado nas fases de levantamento e análise de requisitos do sistema.

2. Diagrama de Classes:

Define a estrutura das classes utilizadas pelo sistema, determinando os atributos e métodos.

3. Diagrama de Objetos:

Associado ao diagrama de classes fornece a visão dos valores armazenados pelos objetos.

4. Diagrama de Estrutura Composta:

Usado para modelar colaborações, descreve uma visão de um conjunto de instâncias que cooperam entre si para executar uma função específica.



5. Diagrama de Sequência:

Preocupa-se com a ordem temporal em que as mensagens são trocados entre os objetos envolvidos.

6. Diagrama de Comunicação:

Complementa o diagrama de sequência e a sua diferença está no fato de não se preocupar com a temporalidade do processo.

7. Diagrama de Máquina de Estados:

Este diagrama procura acompanhar as mudanças sofridas nos estados de uma instância de uma classe.

8. Diagrama de Atividade:

Preocupa-se em descrever os passos percorridos para a conclusão de uma atividade específica.



9. Diagrama de Interação Geral:

Fornece uma visão geral dentro de um sistema ou processo de negócio.

10. Diagrama de Componentes:

Esta associado a linguagem de programação e representa os componentes do sistema, tais como: código fonte, bibliotecas, formulários, arquivos de ajuda.

11. Diagrama de Implantação:

Determina a representação das características físicas, tais como: servidores, estações, topologias, protocolos de comunicações.

12. Diagrama de Pacotes:

Representa os subsistemas ou sub-módulos englobados por um sistema.

13. Diagrama de Tempo:

Utilizado para demonstrar a mudança no estado de um objeto no tempo em resposta a eventos externos.



Para que possamos utilizar e desenvolver projeto OO faremos uso da linguagem de modelagem unificado que é uma linguagem visual utilizada para modelar sistemas computacional, que tem como objetivo fornecer múltiplas visões do sistema a ser modelado,

O avaliado e modelando-o sob diversos aspectos, e assim atingir a modelagem, permitindo que cada diagrama complete os demais.

Diagrama**Estrutura**

- Objetos
- Classe
- Pacotes
- Estrutura
- Composta*
- Componentes
- Implantação

Comportamental

- Maquina de estado*
- Caso de uso
- Atividade*
- Interação
- Sequencia
- Comunicação*
- Interação geral*
- Tempo

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Linguagem de modelagem unificada (UML)

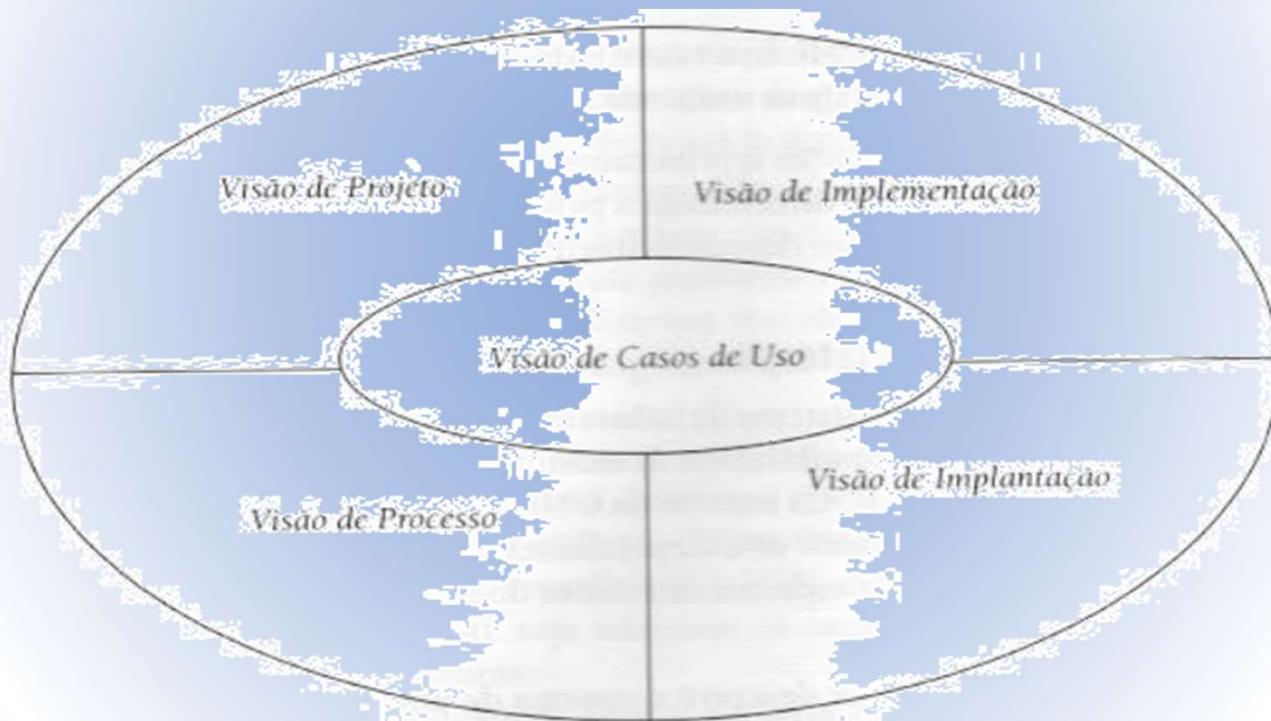


Figura 1-5 Visões (perspectivas) de um sistema de software.

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



Linguagem de modelagem unificada (UML)

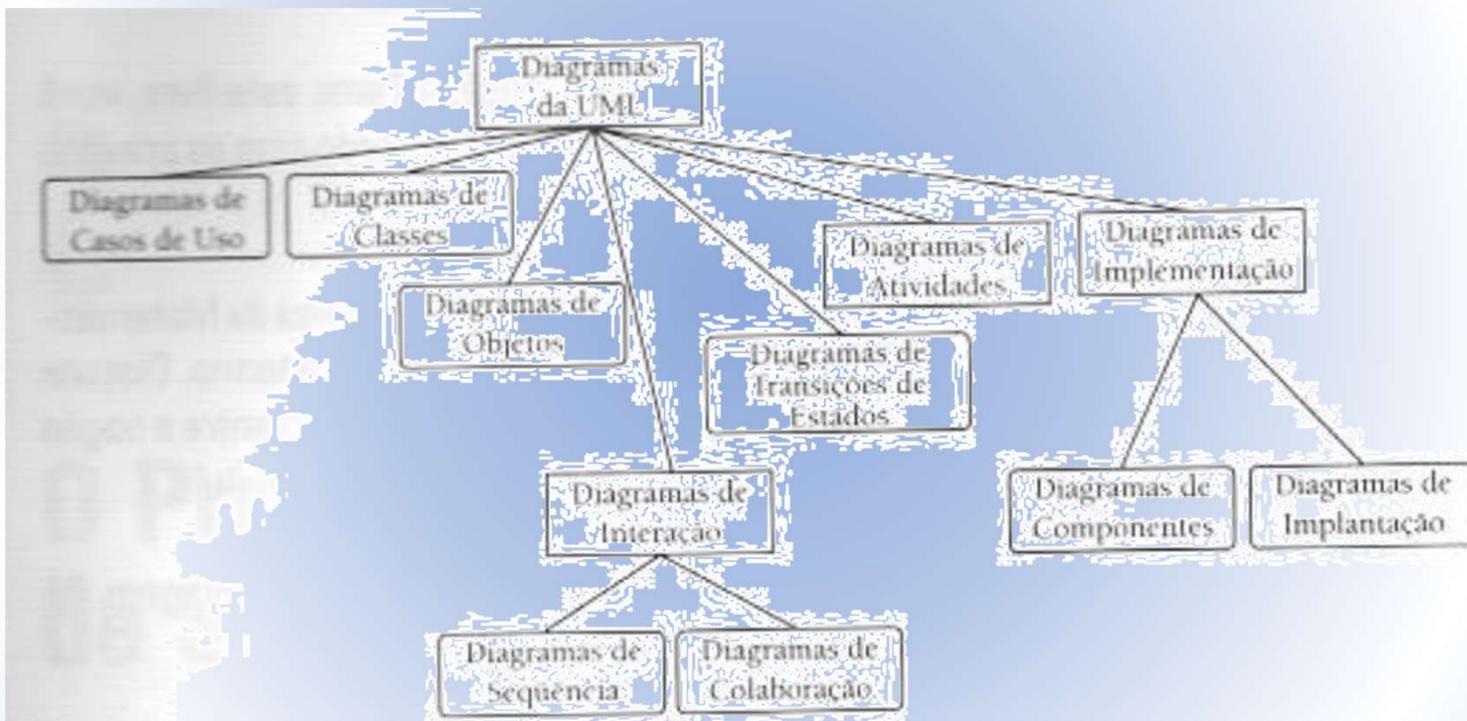


Figura 1-6 Diagramas definidos pela UML.

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



Linguagem de modelagem unificada (UML)

A linguagem de modelagem unificada (UML, do inglês unified modeling language) é “uma linguagem-padrão para descrever/documentar projeto de software. **A UML pode ser usada para visualizar, especificar, construir e documentar os artefatos** de um sistema de **software** intensivo” [Boo05]. Em outras palavras, assim como os arquitetos criam plantas e projetos para serem usados por uma empresa de construção, os **arquitetos de software** criam **diagramas UML** para **ajudar** os desenvolvedores de software a **construir o software**.

- diagrama de atividade
- diagrama de classe
- diagrama de comunicação
- diagrama de implantação
- diagrama de sequência
- diagrama de estado
- diagrama de caso de uso
- dependência
- generalização
- frames de interação
- multiplicidade
- estereótipo
- raias

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.





UML - Diagrama de Classes

Um diagrama de classe fornece uma **visão estática ou estrutural do sistema**. Ele não mostra a natureza dinâmica das comunicações entre os objetos das classes no diagrama. Os **atributos** podem ser valores que a classe calcula a partir de suas variáveis de instância ou valores que a classe pode obter de outros objetos pelos quais é composta.

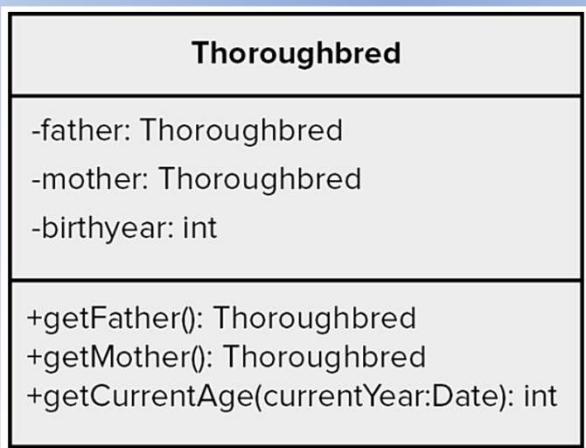


Figura A1.1
Um diagrama para a classe Thoroughbred.

Obs: quarta seção na parte inferior da caixa de classe pode ser usada para listar (CRC) as responsabilidades da classe.

A parte superior contém o **nome** da classe.

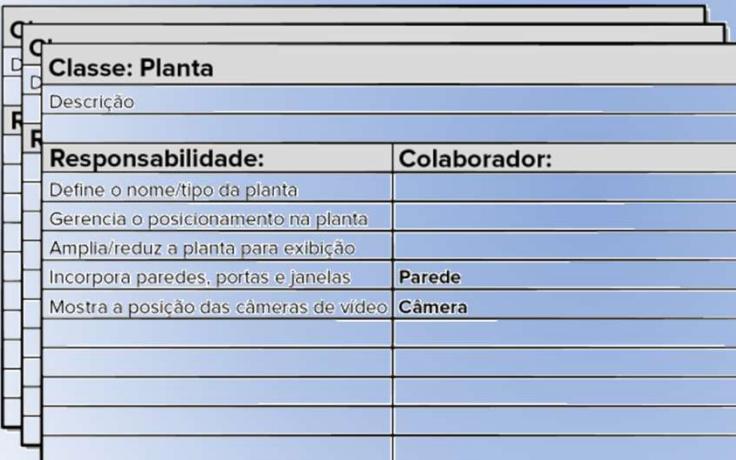
A seção do meio lista os **atributos** da classe.

A terceira seção do diagrama de classes contém as **operações** ou comportamentos da classe.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.

UML - Diagrama de Classes

Obs: quarta seção na parte inferior da caixa de classe pode ser usada para listar (**CRC**) as responsabilidades da classe.



Classes. Deve seguir diretrizes básicas para a identificação de classes e objetos.

Responsabilidades.

Diretrizes básicas para a identificação das responsabilidades (atributos e operações)

Colaborações. responsabilidades de **duas formas**: (1) uma classe pode usar suas **próprias operações** para manipular seus próprios atributos, cumprindo, portanto, determinada responsabilidade; ou (2) uma classe pode **colaborar** com **outras classes**. As colaborações são identificadas determinando se uma classe pode ou não cumprir cada responsabilidade por si só. Caso não possa, ela precisa interagir com outra classe.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Classes

O tipo vem após o **nome** e é separado por dois-pontos. A **visibilidade** é indicada pelos sinais:
“-”private, **“#”protected**,
“~”package ou “+”public

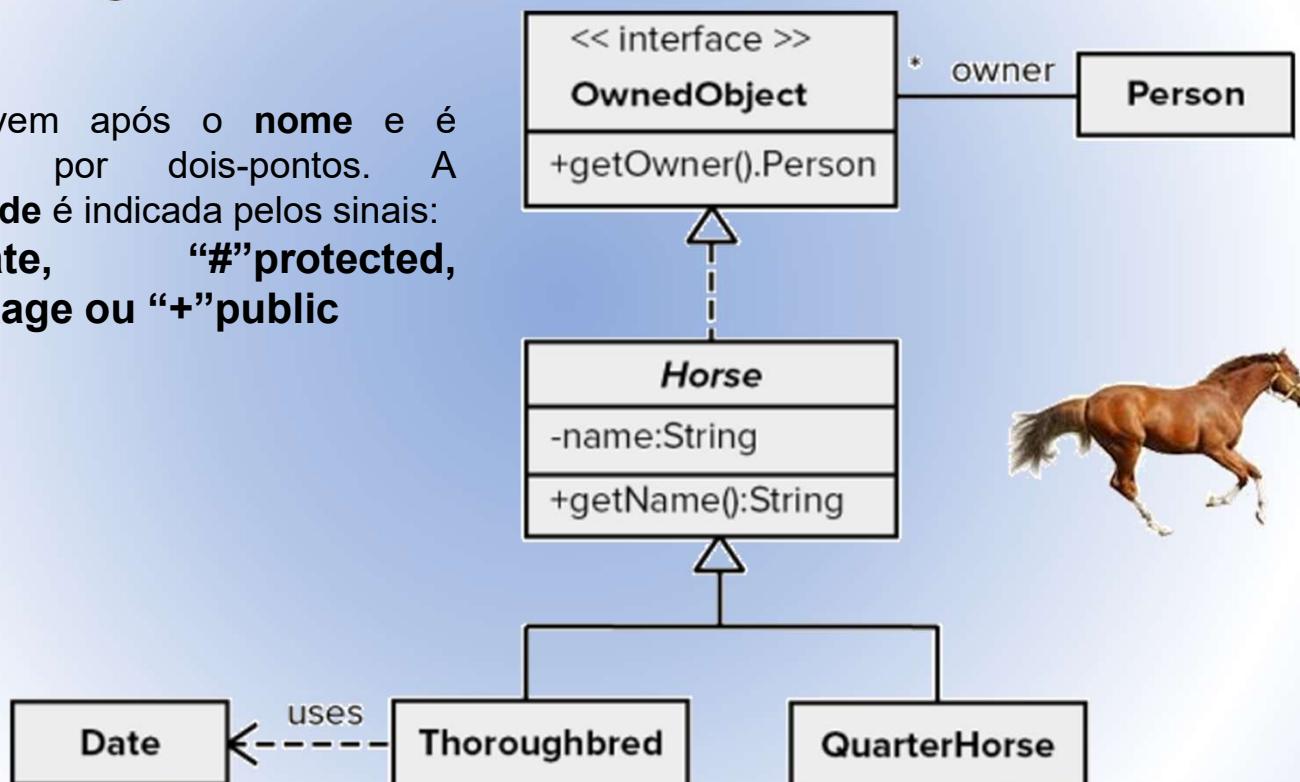


Figura A1.2

Um diagrama de classe referente a cavalos.

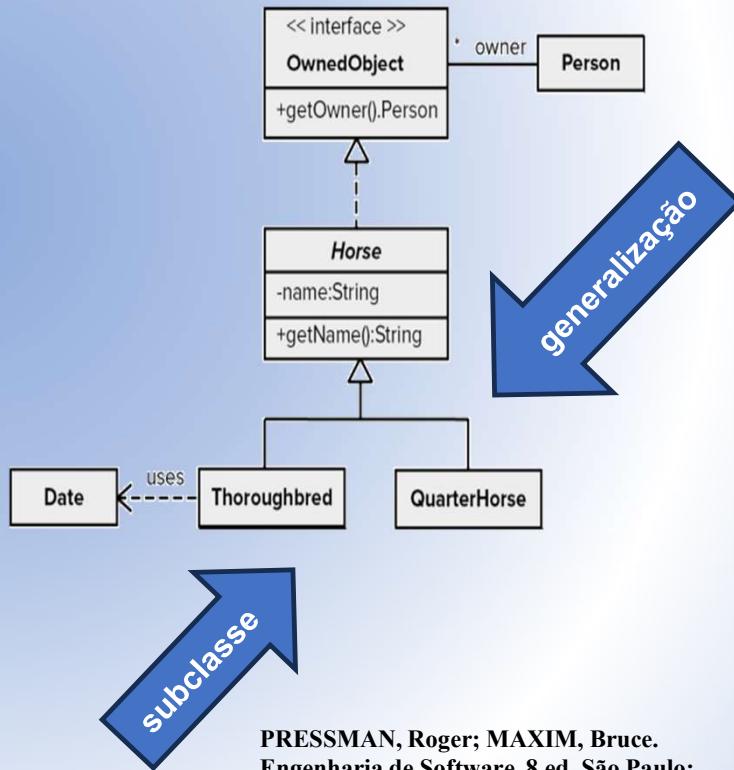
PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Classes

Os diagramas de classe também podem exibir relações entre classes. Uma **classe que seja subclasse de outra classe** é conectada a ela por uma **seta com uma linha sólida como eixo e com uma ponta triangular vazia**.

A seta aponta da subclasse para a superclasse. Em **UML**, uma **relação** como essa é **chamada** de **generalização**.



PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Classes

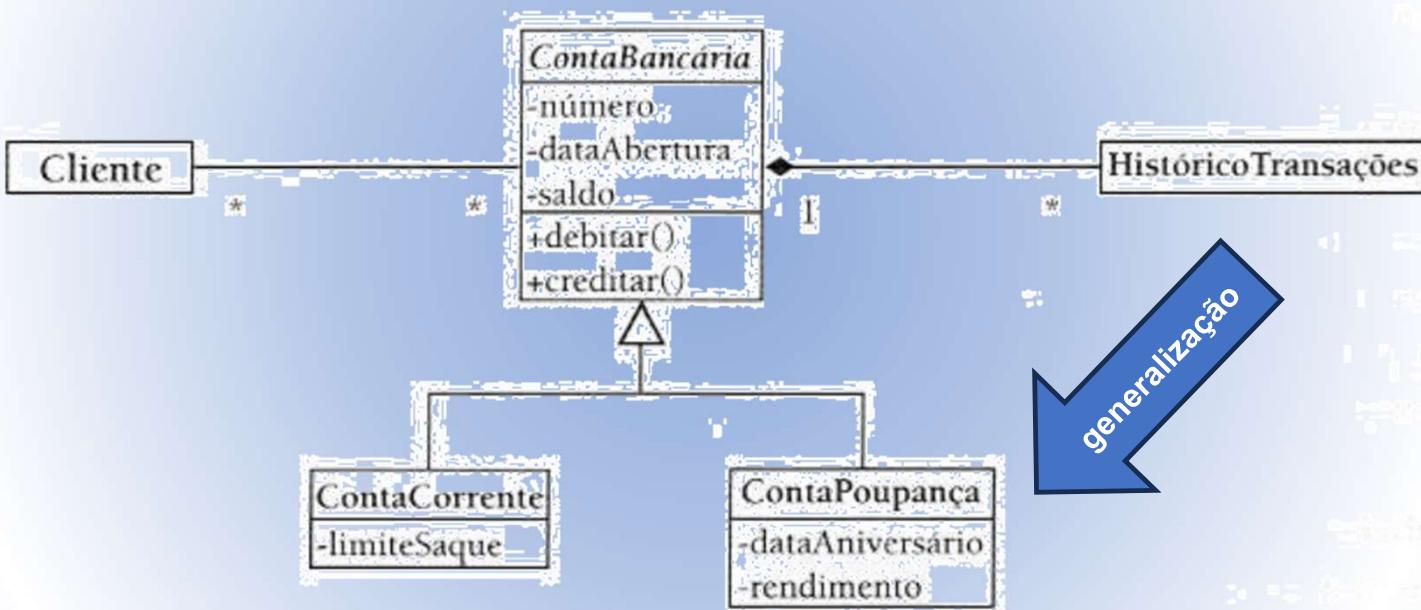


Figura 9-10 Conformidade das subclasses à superclasse.

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



UML - Diagrama de Classes

A **multiplicidade** de uma extremidade de uma associação **indica o número** de objetos daquela classe associados à outra classe.

Uma multiplicidade especificada como “0..1” significa que há 0 ou 1 objeto na extremidade da associação.

A multiplicidade especificada por “1..*” significa um ou mais, e a multiplicidade especificada por “0..*”, ou apenas “*”, significa zero ou mais.

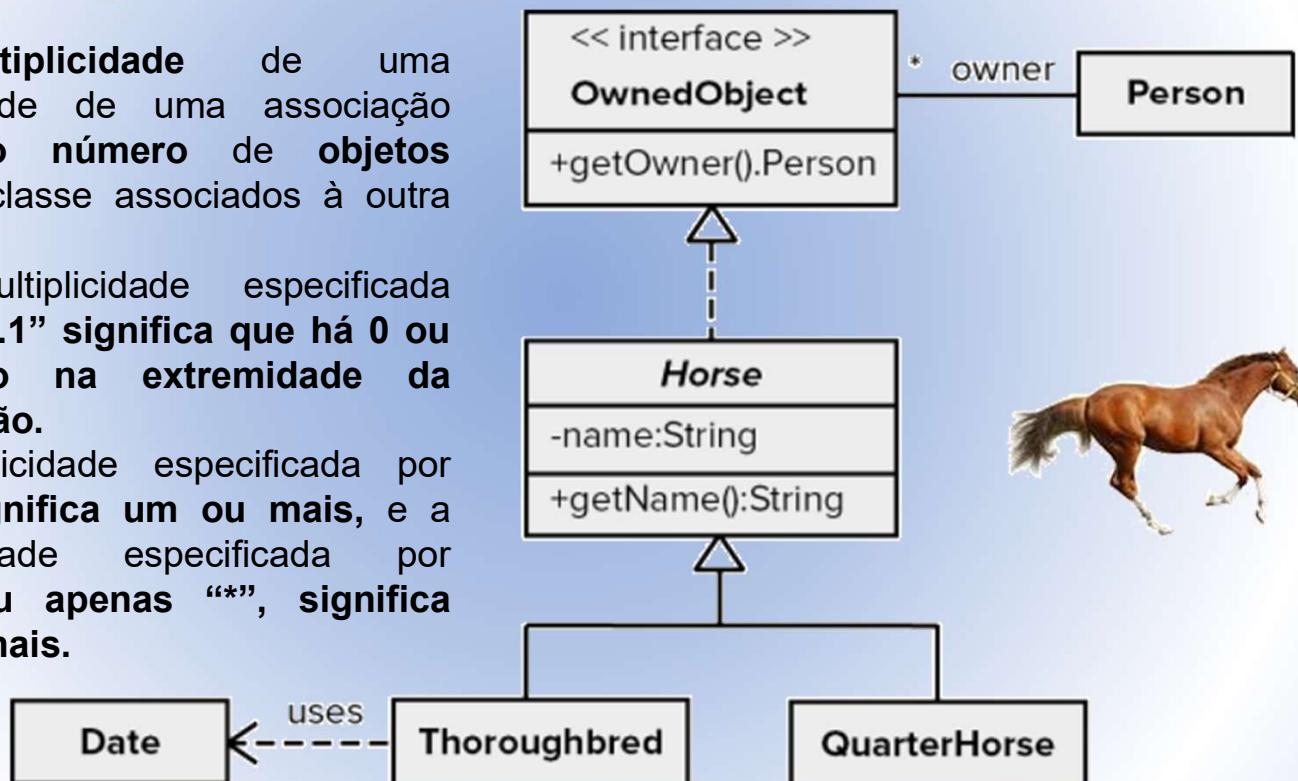


Figura A1.2

Um diagrama de classe referente a cavalos.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Classes

A **multiplicidade** de uma extremidade de uma associação **indica o número** de **objetos** daquela classe associados à outra classe.

Uma multiplicidade especificada **como “0..1”** significa que há 0 ou 1 objeto na **extremidade da associação**.

A multiplicidade especificada por “1..*” significa **um ou mais**, e a multiplicidade especificada por “0..*”, ou apenas “*”, significa **zero ou mais**.

Tabela 5-1 Simbologia para representar multiplicidades

Nome	Simbologia
Apenas Um	1
Zero ou Muitos	0..*
Um ou Muitos	1..*
Zero ou Um	0..1
Intervalo Específico	1..1

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

UML - Diagrama de Classes

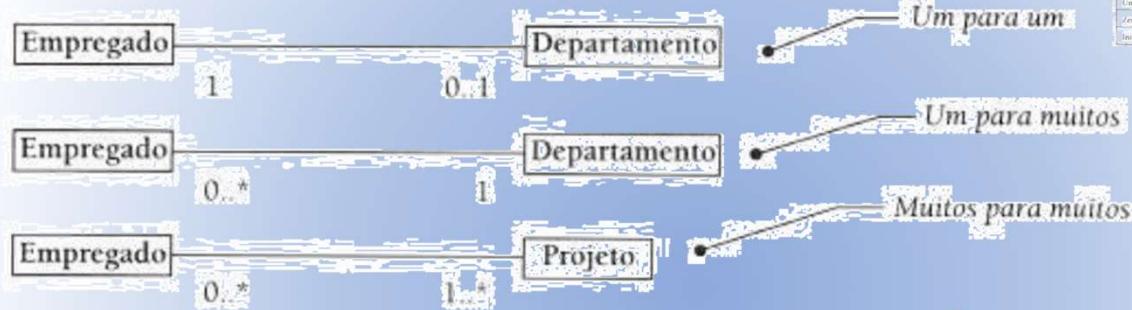


Figura 5-5 Exemplos de cada tipo de conectividade.



Figura 5-3 Exemplo de utilização dos símbolos de multiplicidade.



Figura 5-4 Exemplo de utilização de intervalo de valores para multiplicidade.

Tabela 5-1 Símbologia para representar multiplicidades

Nome	Símbologia
Apenas Um	1
Zero ou Mais	0..*
Um ou Mais	1..*
Zero ou Um	0..1
Intervalo Específico	U..L



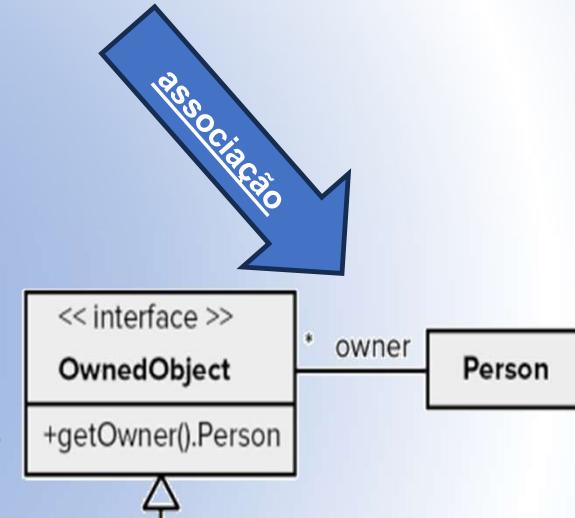
BEZERRA, Eduardo. Princípios de Análise e Projeto de Sistemas com UML. 3 ed. Rio de Janeiro: Elsevier, 2015.

UML - Diagrama de Classes

Se uma extremidade de uma associação apresenta multiplicidade maior do que 1, os objetos da classe aos quais se faz referência na extremidade da associação provavelmente estão armazenados em uma coleção, como um conjunto ou uma lista ordenada.

Uma agregação é um tipo especial de associação representada por um losango vazio em uma extremidade do ícone. Ela indica uma relação “todo/parte”, em que a classe para a qual a seta aponta é considerada uma “parte” da classe na extremidade do losango da associação.

Uma composição é uma agregação indicando forte relação de propriedade entre as partes.



Usou-se um * como multiplicidade na extremidade OwnedObject da associação com a classe Person, porque uma Person pode possuir zero ou mais objetos.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Classes



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



UML - Diagrama de Classes



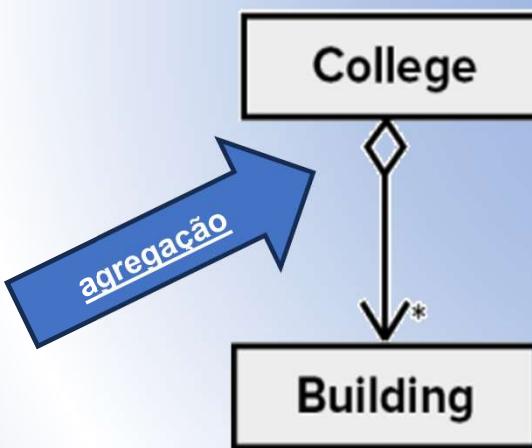
Figura 5-9 Exemplo de classe associativa.

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



UML - Diagrama de Classes

Uma agregação é um tipo especial de associação representada por um losango vazio em uma extremidade do ícone. Ela indica uma relação “todo/parte”, em que a classe para a qual a seta aponta é considerada uma “parte” da classe na extremidade do losango da associação.



Uma classe **College** tem uma agregação de objetos **Building** que representam os edifícios que formam o campus. A universidade (College) tem também uma coleção de cursos.

{deve ocorrer em um edifício}

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Classes

Uma agregação é um tipo especial de associação representada por um losango vazio em uma extremidade do ícone. Ela indica uma relação “todo/parte”, em que a classe para a qual a seta aponta é considerada uma “parte” da classe na extremidade do losango da associação.



Figura 5-8 Exemplo de agregação. Entre Jogador (parte) e Equipe (todo)
e entre Equipe (parte) e AssociaçãoEsportiva (todo).

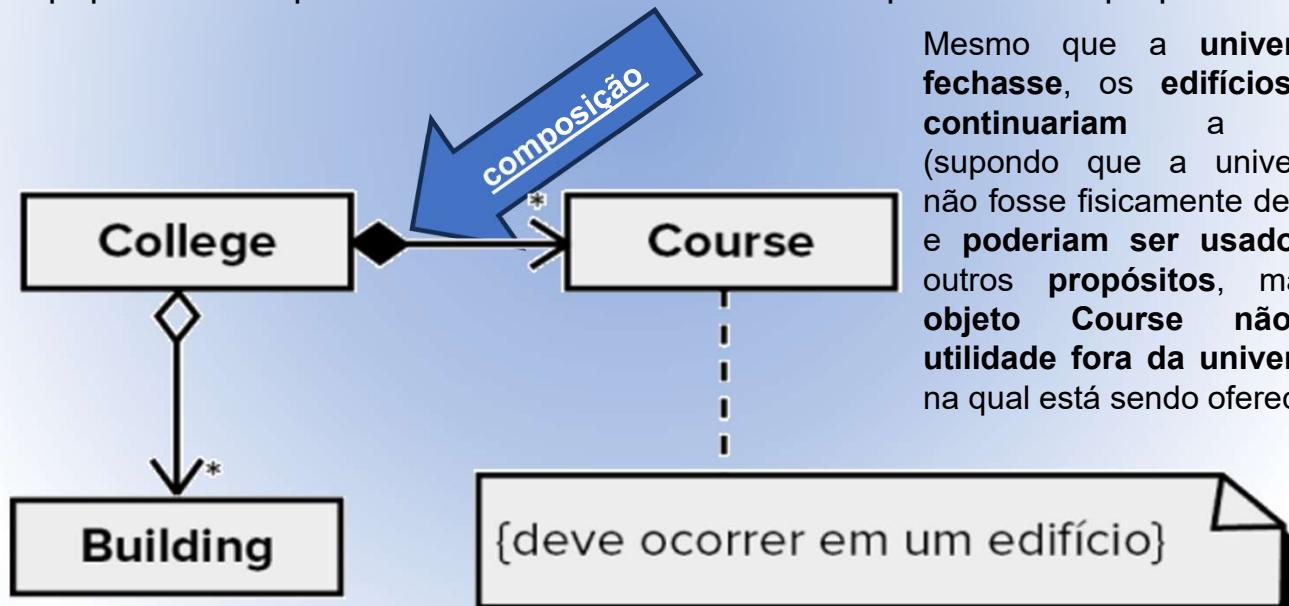


BEZERRA, Eduardo. Princípios de Análise e Projeto de Sistemas com UML. 3 ed. Rio de Janeiro: Elsevier, 2015.



UML - Diagrama de Classes

Uma composição é uma agregação indicando forte relação de propriedade entre as partes. Em uma composição, as partes vivem e morrem com o proprietário porque não têm um papel a desempenhar no sistema de software independente do proprietário.



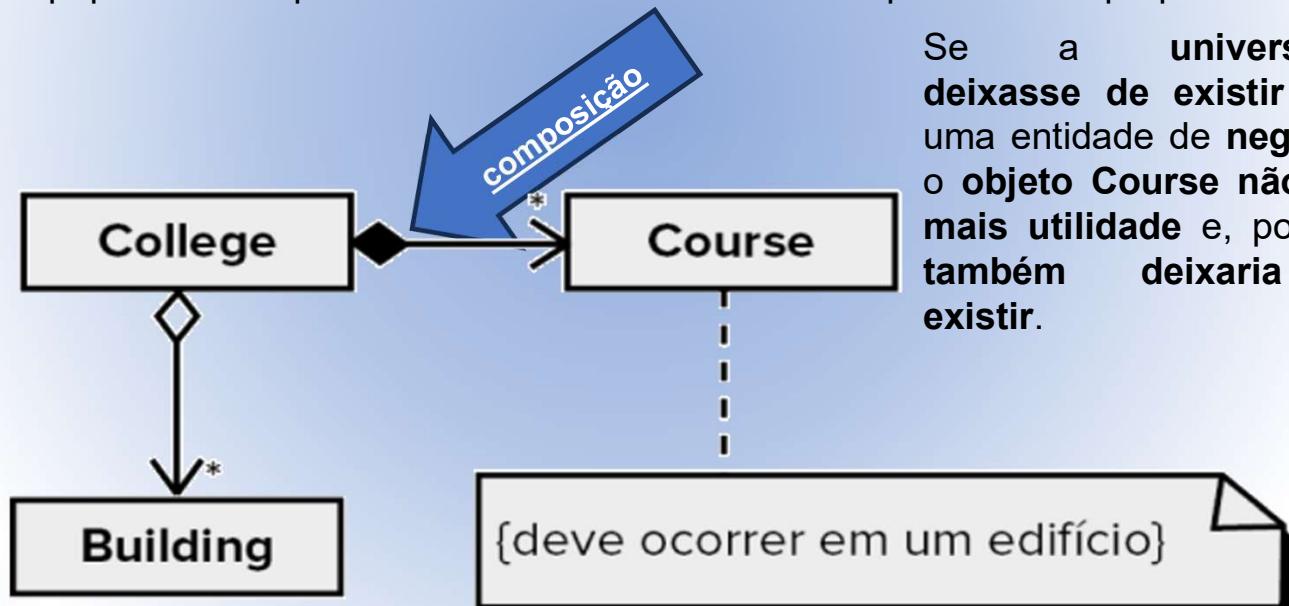
Mesmo que a **universidade fechasse**, os **edifícios** ainda **continuariam** a **existir** (supondo que a universidade não fosse fisicamente destruída) e **poderiam ser usados** para outros **propósitos**, mas um **objeto Course** não tem **utilidade** fora da **universidade** na qual está sendo oferecido.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Classes

Uma composição é uma agregação indicando forte relação de propriedade entre as partes. Em uma composição, as partes vivem e morrem com o proprietário porque não têm um papel a desempenhar no sistema de software independente do proprietário.



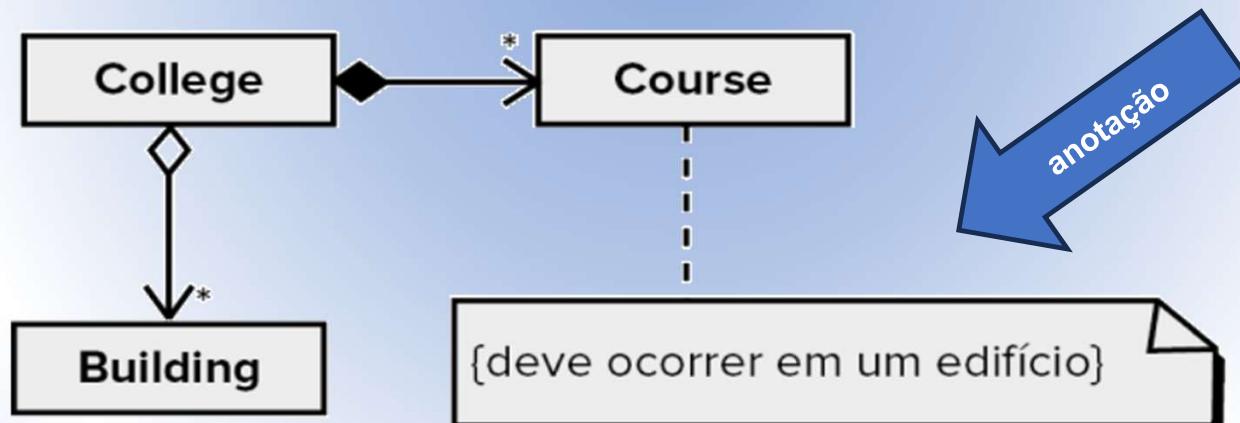
Se a universidade deixasse de existir como uma entidade de negócios, o objeto Course não teria mais utilidade e, portanto, também deixaria de existir.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Classes

Uma **anotação** (note), pode ter **informações** sobre o papel de uma **classe** ou **restrições** (estará entre chaves) que **todos** os **objetos** daquela **classe** devem satisfazer. Representada por uma caixa com um canto dobrado e conectada a outros ícones por uma linha pontilhada. É similar a um **comentário** de linguagem de programação.



PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.





5. Teoria relacional: dependências funcionais e multivaloradas, formas normais.

Grau de um Relacionamento

Indica o número de conjuntos-entidade (classes distintas de objetos) cujas instâncias podem estar associadas umas as outras através de um relacionamento.

Relacionamento Unário

Relacionamento Binário

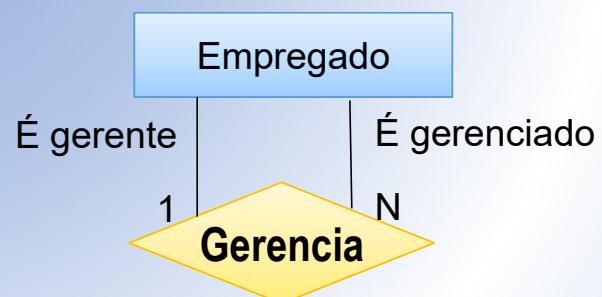
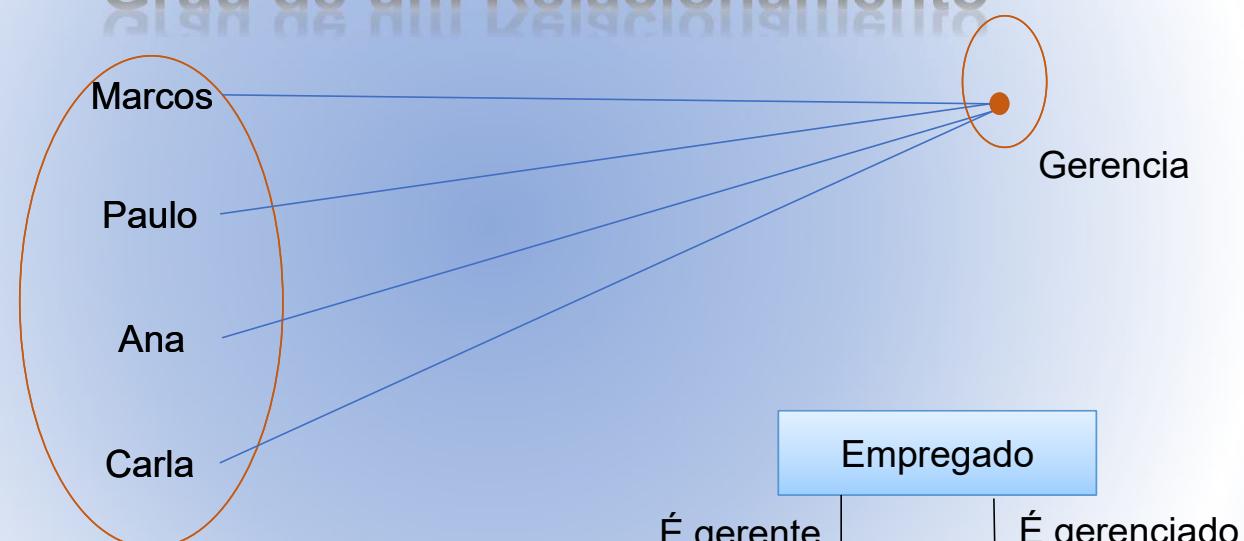
Relacionamento Ternário



5. Teoria relacional: dependências funcionais e multivaloradas, formas normais.

Grau de um Relacionamento

Funcionário

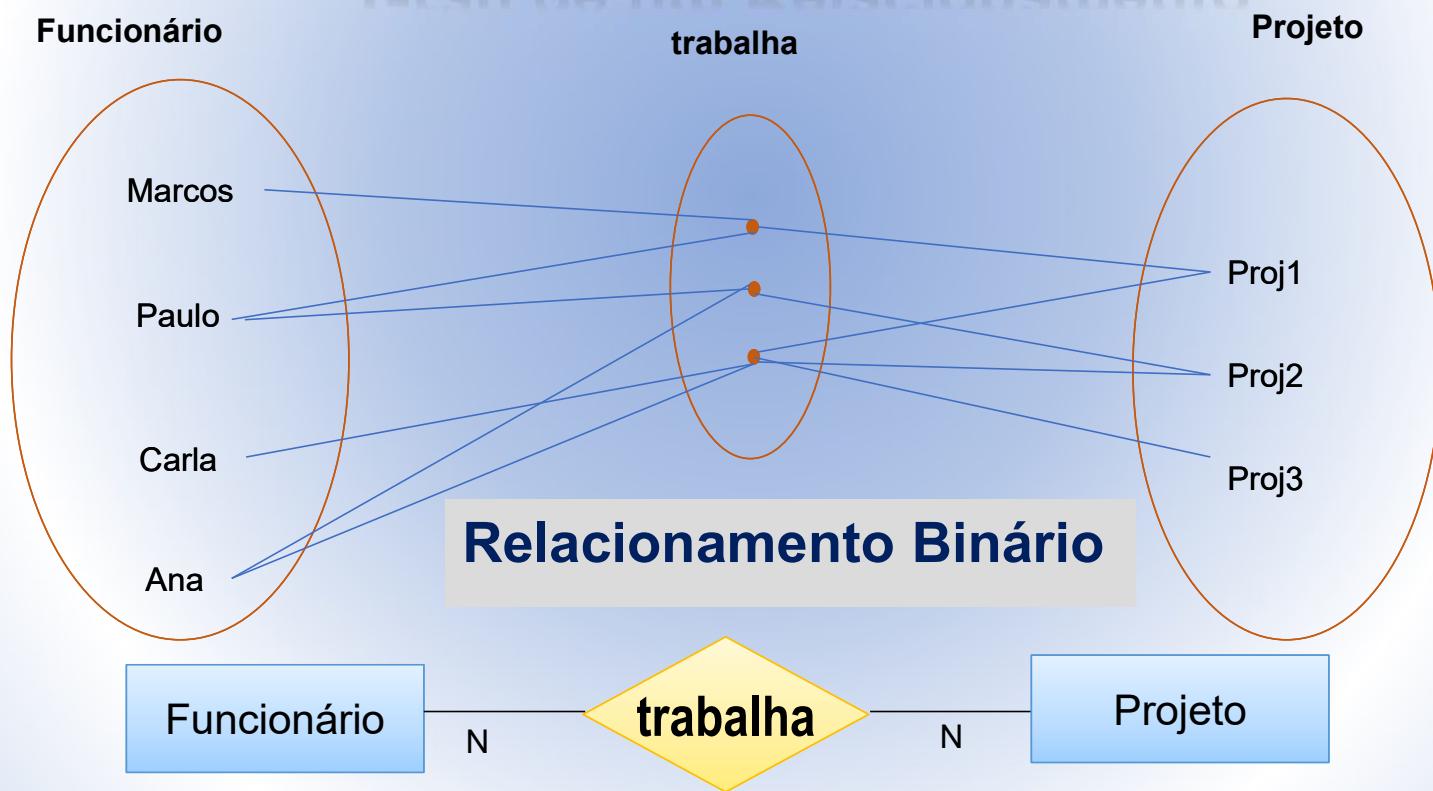


Relacionamento Unário



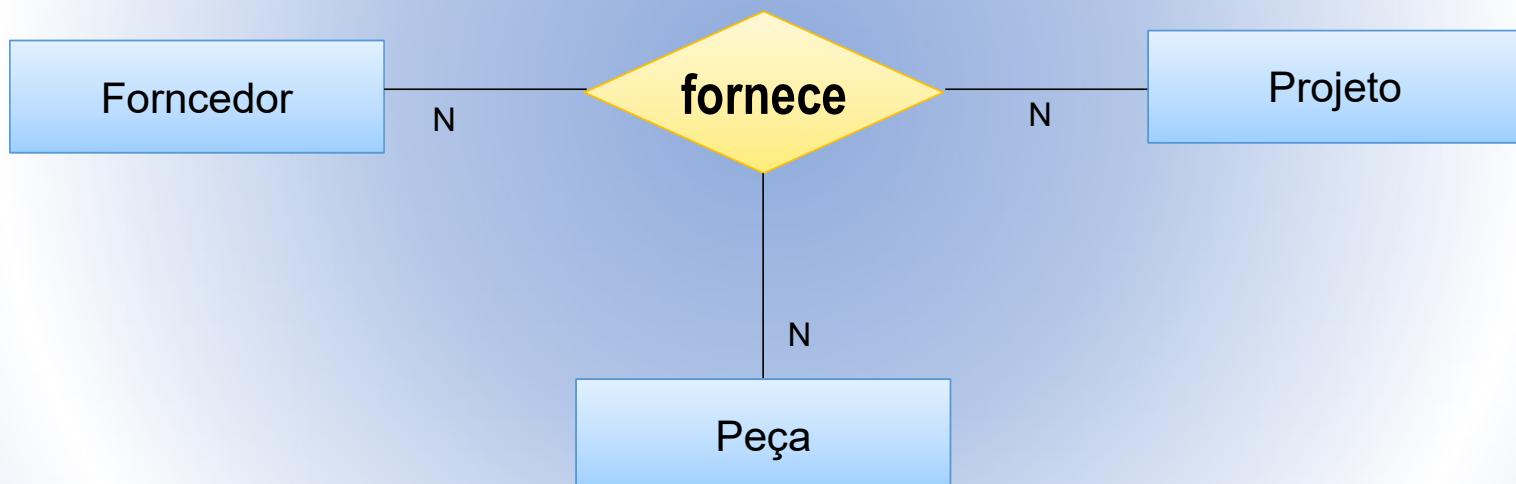
5. Teoria relacional: dependências funcionais e multivaloradas, formas normais.

Grau de um Relacionamento



5. Teoria relacional: dependências funcionais e multivaloradas, formas normais.

Grau de um Relacionamento



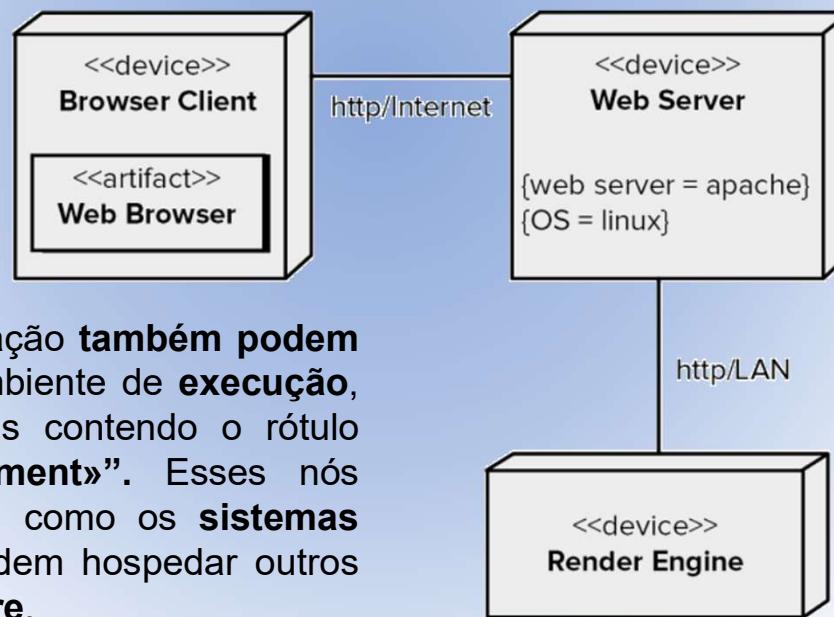
Relacionamento Ternário

Considere Entidade com letra
maiúscula



UML - Diagrama de Implantação

Focalizam a estrutura do sistema de software e **são úteis** para mostrar a **distribuição física** de um **sistema de software entre plataformas de hardware e ambientes de execução**.



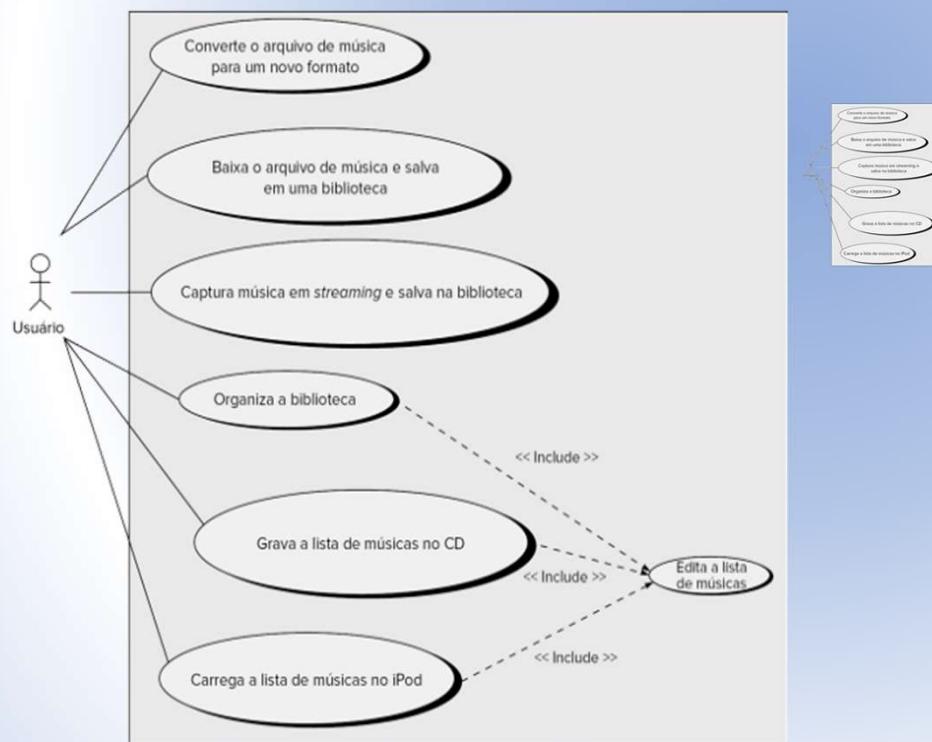
Diagramas de implantação **também podem mostrar** os nós do ambiente de **execução**, desenhados em caixas contendo o rótulo **“execution environment”**. Esses nós representam sistemas, como os **sistemas operacionais**, que podem hospedar outros **programas de software**.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Caso de Uso

Ajudam a determinar a **funcionalidade** e as **características** do software sob o **ponto de vista do usuário**.



No diagrama de caso de uso, os **casos** de uso são mostrados como **elipses**.

Os **atores** são conectados por linhas aos **casos** de uso que eles **executam**.

A **inclusão** é indicada nos diagramas de caso de uso, por meio de uma **seta tracejada** identificada como **«include»**, conectando um caso de uso a outro.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Caso de Uso

O relacionamento de extensão, em que um caso de uso A estende um caso de uso B, é representado por uma seta direcionada de A para B. Essa seta é rotulada com o estereótipo <<estende>>. A Figura 4-5 ilustra a representação desse relacionamento. Essa figura mostra que os casos de uso Corrigir Ortografia e Substituir Texto têm sequências de interações que são opcionalmente utilizadas quando o ator Escritor estiver utilizando a caso de uso Editar Documento.



Figura 4-5 Exemplo de relacionamento de extensão

No diagrama de caso de uso, os **casos** de uso são mostrados como **elipses**.

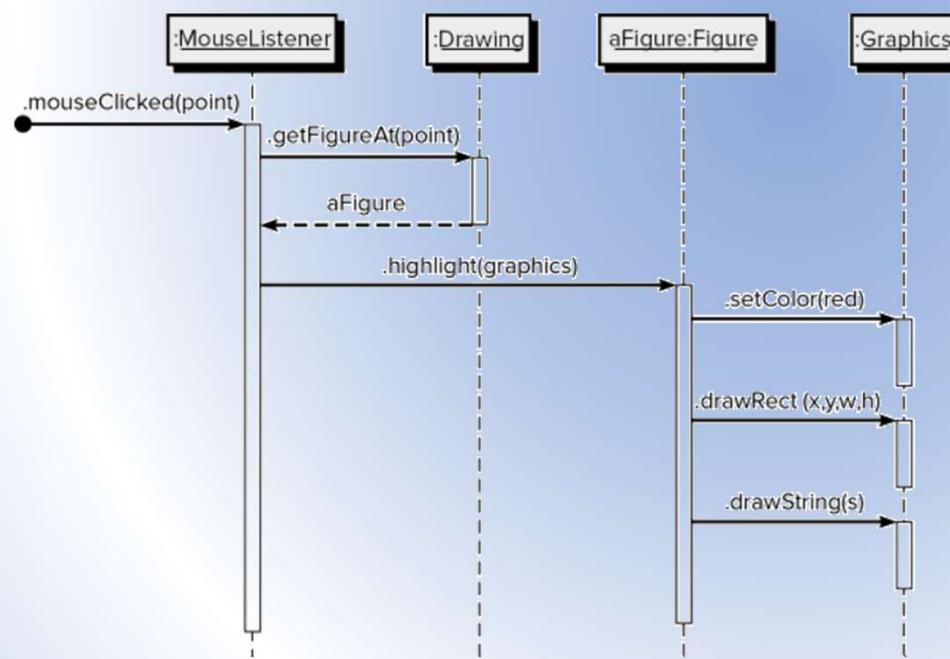
Os **atores** são conectados por linhas aos **casos** de uso que eles **executam**.

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



UML - Diagrama de Sequência

É utilizado para indicar as comunicações dinâmicas entre objetos durante a execução de uma tarefa. Ele mostra a ordem temporal na qual as mensagens são enviadas entre os objetos para executar aquela tarefa.



Se a caixa representa um objeto, dentro da caixa **pode-se** opcionalmente declarar o tipo do objeto, precedido de dois-pontos.

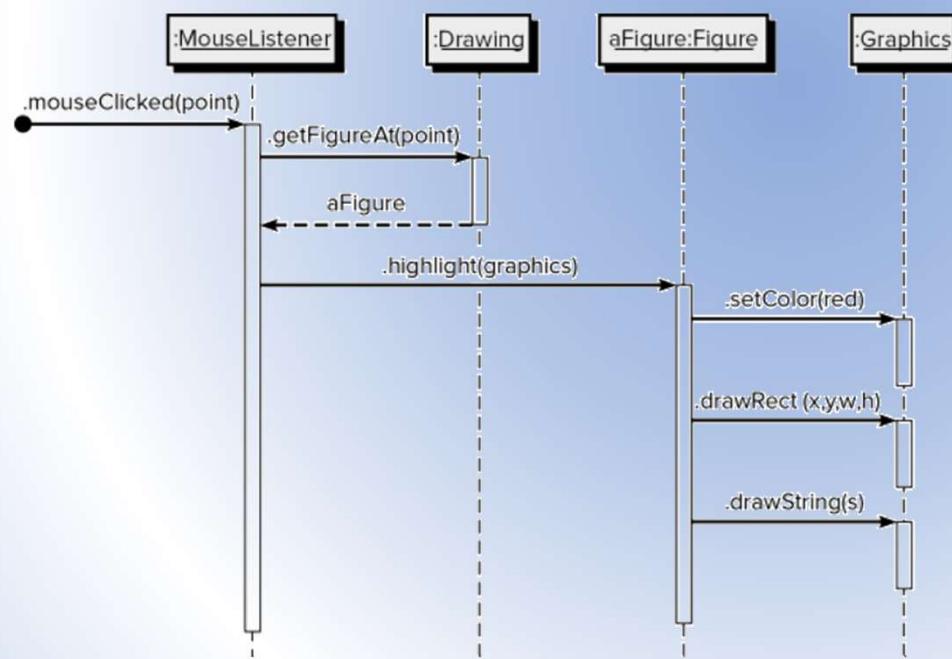
Abaixo de cada caixa há uma linha tracejada chamada de **linha de vida do objeto**. O eixo **vertical** no diagrama de sequência **corresponde ao tempo**, e o tempo **aumenta à medida que se caminha para baixo**.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Sequência

É utilizado para indicar as comunicações dinâmicas entre objetos durante a execução de uma tarefa. Ele mostra a ordem temporal na qual as mensagens são enviadas entre os objetos para executar aquela tarefa.



Um diagrama de sequência mostra chamadas de método usando setas horizontais do chamador para o chamado, identificadas com o nome do método e, opcionalmente, incluindo seus parâmetros, seus tipos e o tipo de retorno.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Sequência

É utilizado para indicar as comunicações dinâmicas entre objetos durante a execução de uma tarefa. Ele mostra a ordem temporal na qual as mensagens são enviadas entre os objetos para executar aquela tarefa.

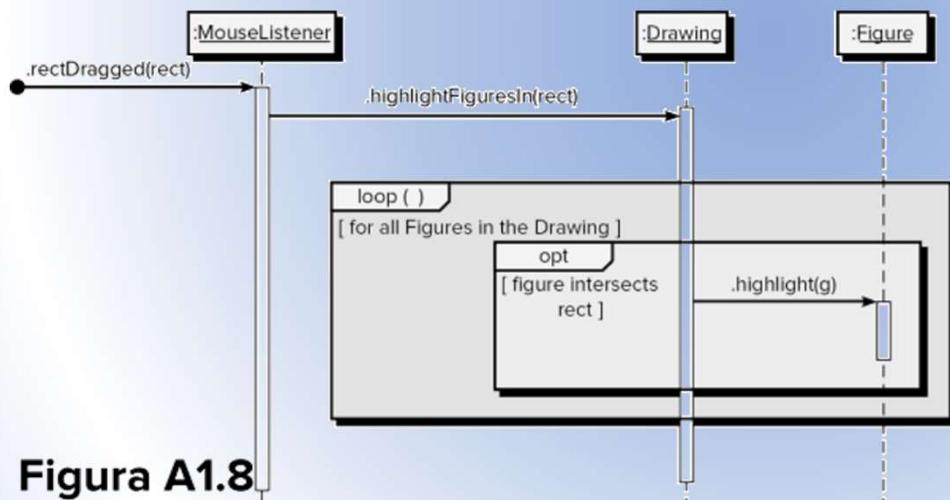


Figura A1.8

Um diagrama de sequência com dois frames de interação.

Se você ainda quer incluir laços, condicionais e outras estruturas de controle em um diagrama de sequência, use frames de interação (interaction frames), que são retângulos que envolvem as partes do diagrama e são identificados com o tipo de estrutura de controle que representam.

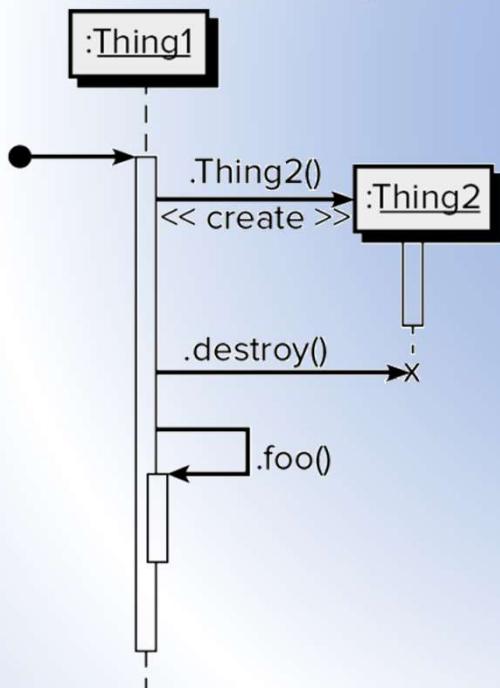
PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.





UML - Diagrama de Sequência

É utilizado para indicar as comunicações dinâmicas entre objetos durante a execução de uma tarefa. Ele mostra a ordem temporal na qual as mensagens são enviadas entre os objetos para executar aquela tarefa.



Você pode mostrar um objeto fazendo-o enviar a si próprio uma mensagem, com uma seta saindo do objeto, virando para baixo e, em seguida, apontando de volta para o mesmo objeto.

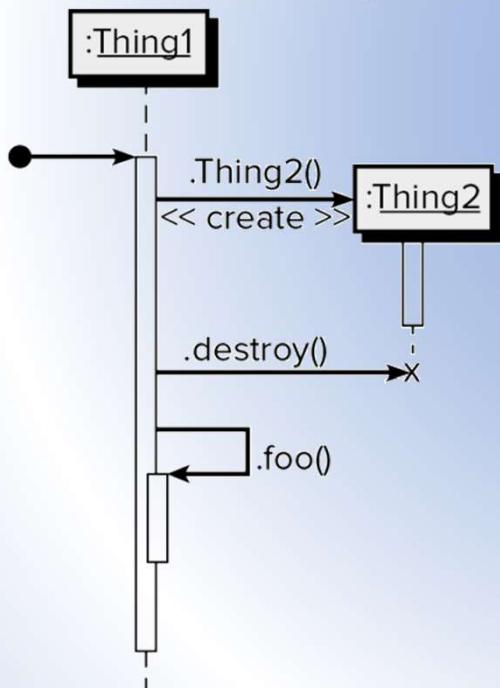
Você pode mostrar a criação do objeto traçando uma seta identificada de forma apropriada (p. ex., com um rótulo «create») para a caixa de um objeto. Nesse caso, a caixa aparecerá no diagrama abaixo das caixas que correspondem a objetos que já existiam quando a ação começa.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Sequência

É utilizado para **indicar** as **comunicações dinâmicas** entre objetos durante a **execução de uma tarefa**. Ele **mostra a ordem temporal** na qual as **mensagens** são **enviadas** entre os objetos para **executar** aquela **tarefa**.



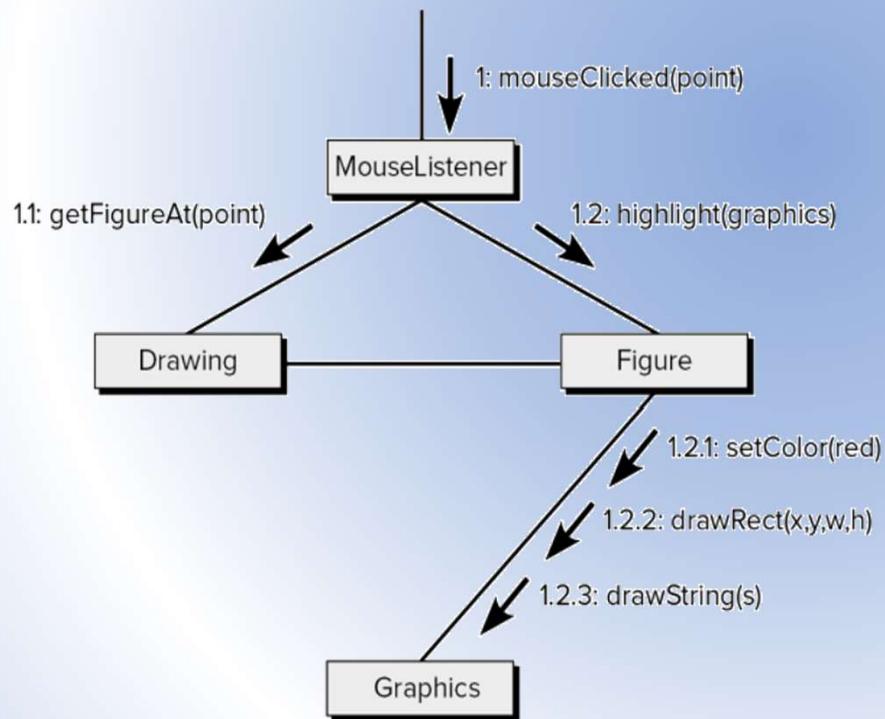
Mensagens **síncronas** são exibidas com **pontas de setas cheias**, enquanto mensagens **assíncronas** são mostradas com **pontas de setas em traço**.

Você pode representar a destruição de um **objeto** com **um X grande** no final da sua linha de vida. **Outros** objetos **podem destruir** um **objeto** e, nesse caso, uma seta aponta do outro objeto para o X. Um X é útil também para **indicar que um objeto não é mais utilizável** e está pronto para ser enviado à coleta de lixo.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.

UML - Diagrama de Comunicação

Fornece outra indicação da **ordem temporal das comunicações**, mas dá ênfase às **relações entre os objetos e classes** em vez da ordem temporal.



A seta é **identificada com um número e um nome de mensagem**. Se a mensagem que chega for identificada com o **número 1** e se ela **faz o objeto receptor invocar outras mensagens** em outros objetos, aquelas **mensagens são representadas por setas do emissor para o receptor com uma linha de associação** e recebem **números 1.1, 1.2 e assim por diante**,

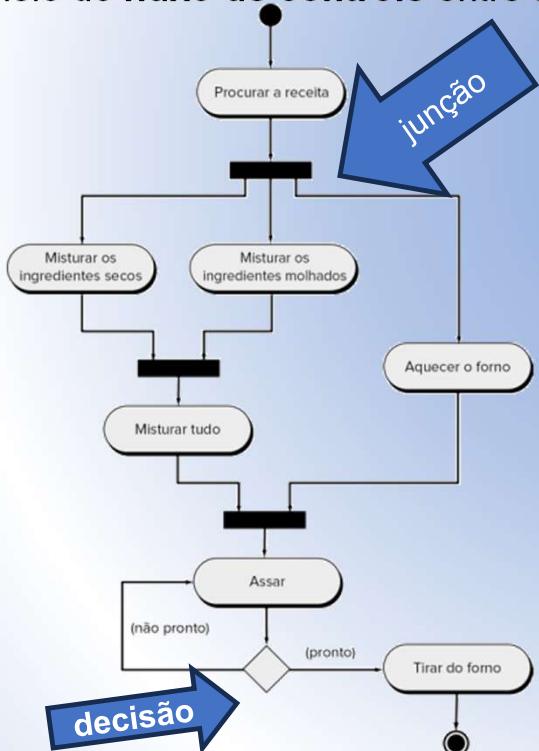
PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.





UML - Diagrama de Atividade

Mostra o **comportamento dinâmico** de um sistema ou de parte de um sistema por meio do **fluxo de controle** entre ações que o sistema executa.



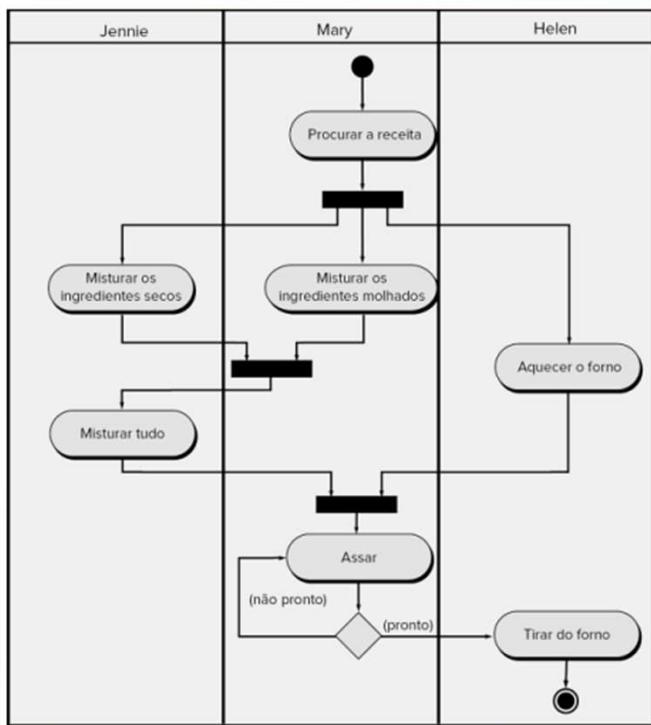
Uma junção (join) é uma maneira de **sincronizar fluxos de controle concorrentes**. Ela é representada por uma **barra preta horizontal** com duas ou mais setas chegando e uma seta saindo.

Um nó **decisão** corresponde a uma ramificação no fluxo de controle baseada em uma condição. O fluxo de controle segue a seta que sai cuja condição é verdadeira (true).

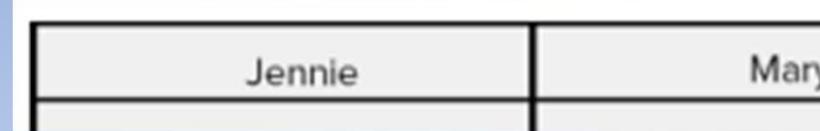
PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.

UML - Diagrama de Atividade

Mostra o **comportamento dinâmico** de um sistema ou de parte de um sistema por meio do **fluxo de controle** entre ações que o sistema executa.



Mas se você quiser **indicar** como as **ações** são divididas entre os participantes, decore o diagrama de atividades com **raias** (swimlanes), são formadas dividindo-se o diagrama em tiras ou “faixas”,



PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Atividade

Mostra o **comportamento dinâmico** de um sistema ou de parte de um sistema por meio do **fluxo de controle** entre ações que o sistema executa.

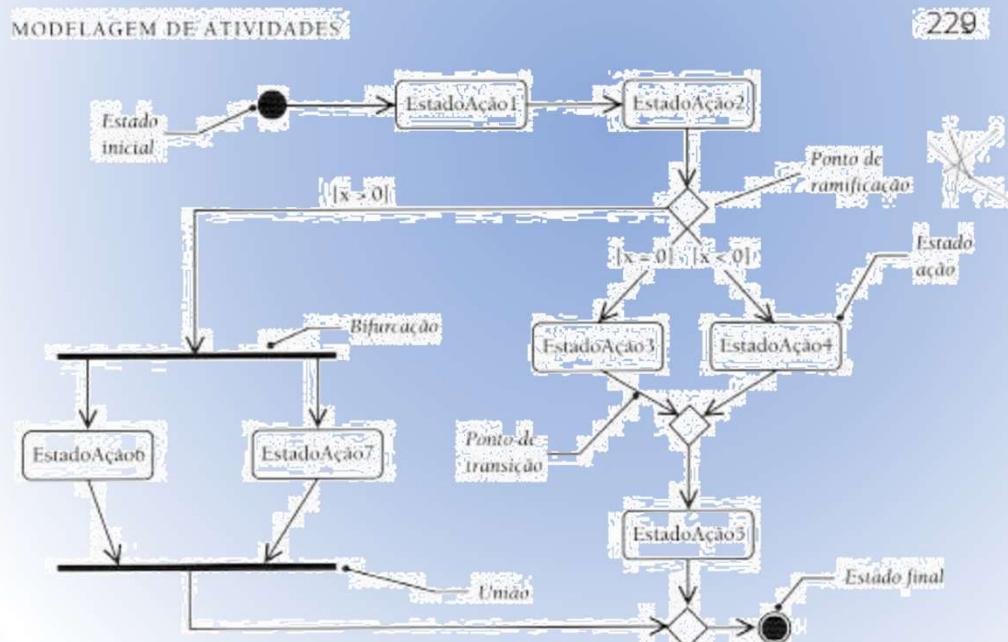


Figura 11-1 Elementos de um diagrama de atividade.

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



UML - Diagrama de Atividade

Mostra o **comportamento dinâmico** de um sistema ou de parte de um sistema por meio do **fluxo de controle** entre ações que o sistema executa.

MODELAGEM DE ATIVIDADES

231

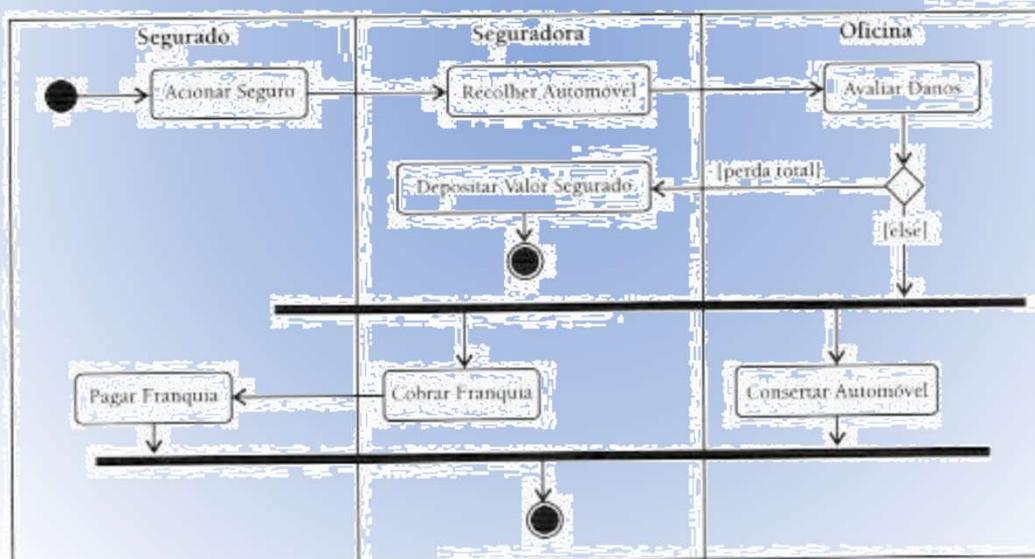


Figura 11-2 Raias de natação.

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



UML - Diagrama de Estado



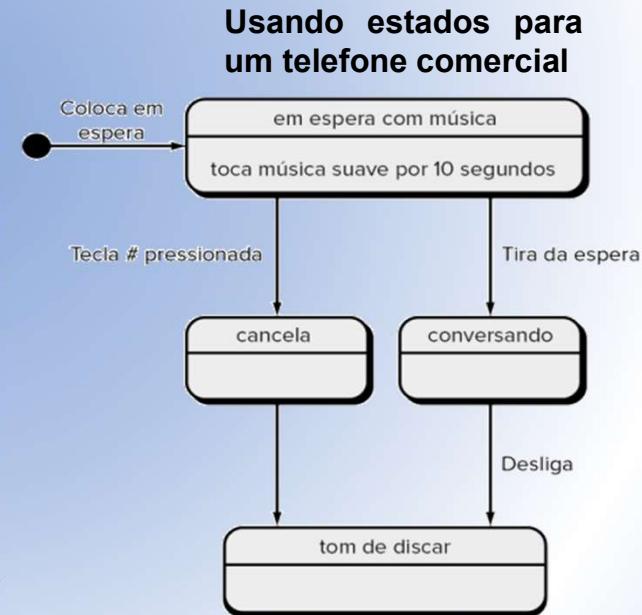
PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Estado

O **comportamento** de um objeto em determinado instante frequentemente depende do seu estado, ou seja, dos **valores de suas variáveis naquele instante**.

Quando uma **chamada** é colocada em **espera**, vai para o **estado Em espera com música** (toca música suave por 10 segundos). Após 10 segundos, a do-activity do estado é **completada**, e o **estado** se comporta como um **estado normal** de não atividade. Se a pessoa que está **chamando** pressiona a **tecla #** quando a chamada está no **estado Em espera com música**, a **chamada faz uma transição** para o **estado Cancela**, e o telefone **passa** imediatamente para o **estado tom de discar**. Se a **tecla #** é pressionada **antes** de completar os 10 segundos de música de espera, a do-activity é **interrompida**, e a **música para imediatamente**.

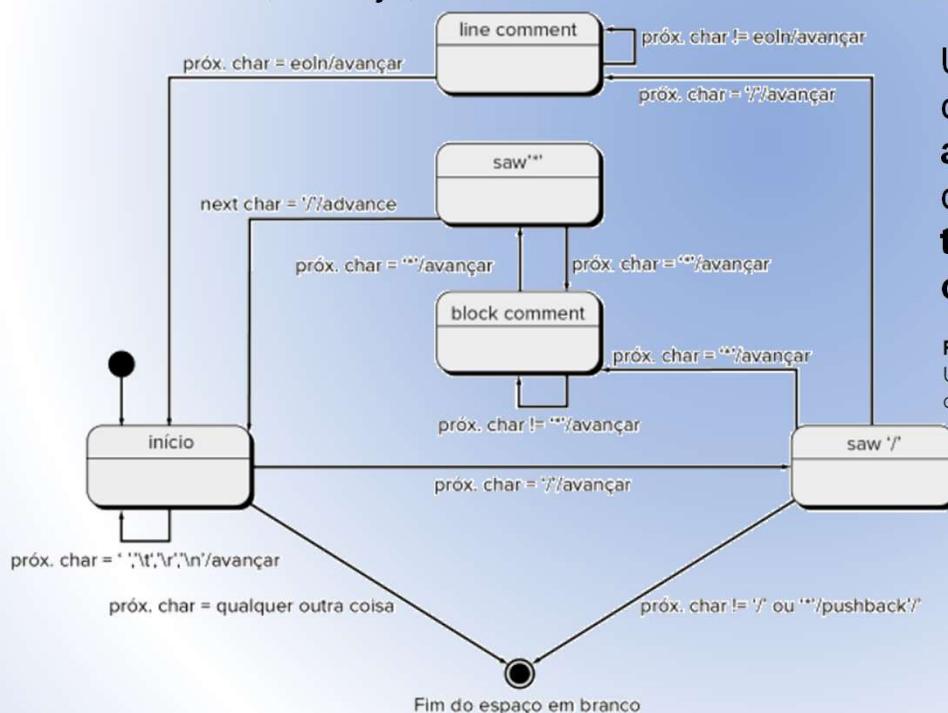


PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Estado

O **comportamento** de um objeto em determinado instante frequentemente depende do seu estado, ou seja, dos **valores de suas variáveis naquele instante**.



Um diagrama de estado **modela os estados de um objeto**, as **ações executadas** dependendo daqueles estados e as **transições entre os estados** do objeto.

Figura A1.13

Um diagrama de estado para avançar além do espaço em branco e comentários em Java.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Estado

O **comportamento** de um objeto em determinado instante frequentemente depende do seu estado, ou seja, dos **valores de suas variáveis naquele instante**.

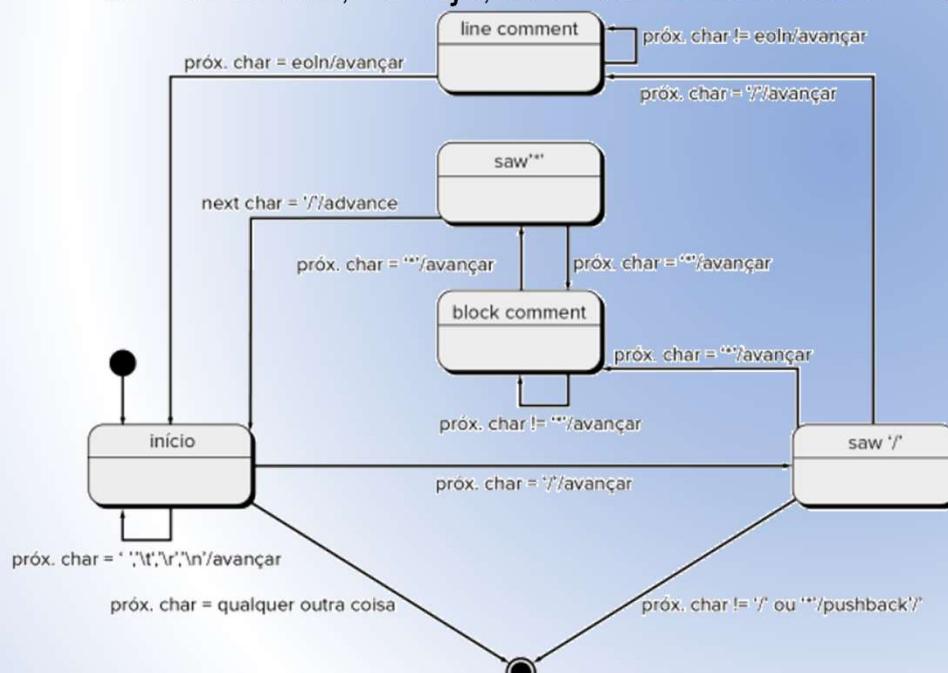


Figura A1.13

Um diagrama de estado para avançar além do espaço em branco e comentários em Java.

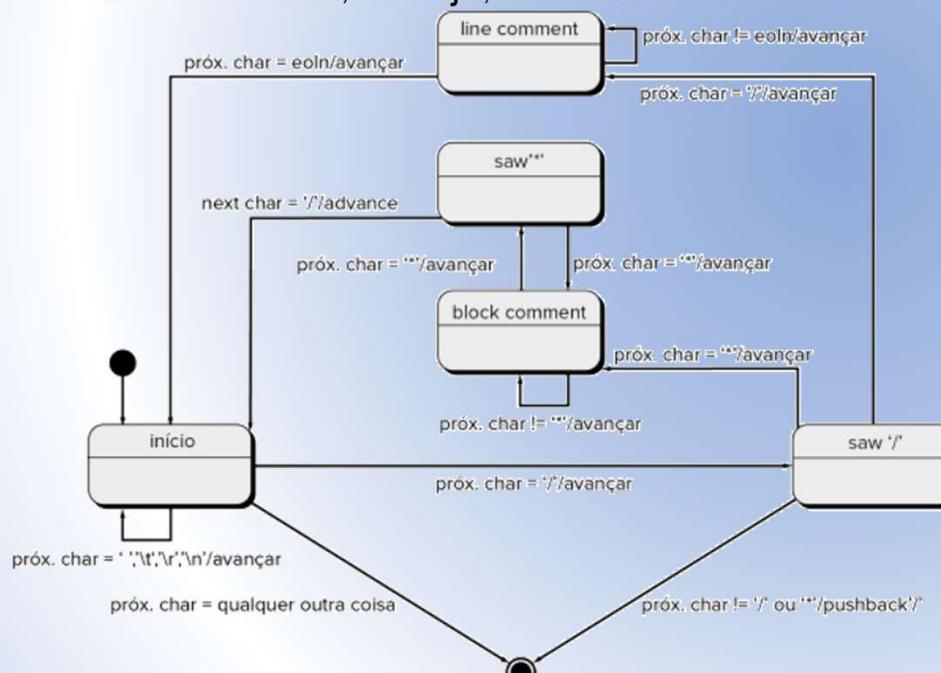
Considere o diagrama de estado para uma parte de um **compilador Java**. A **entrada** do compilador é um arquivo de **texto**, que pode ser considerado uma longa **sequência** (string) de **caracteres**. O compilador **lê os caracteres, um de cada vez**, e a partir deles determina a estrutura do programa. Uma **pequena parte** desse processo consiste em **ignorar caracteres “espaço em branco”** (p. ex., os caracteres espaço, tabulação, nova linha e return) e **caracteres dentro de um comentário**.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



UML - Diagrama de Estado

O **comportamento** de um objeto em determinado instante frequentemente depende do seu estado, ou seja, dos **valores de suas variáveis naquele instante**.



Suponha que o **compilador** delegue ao **WhiteSpaceAndCommentEliminator** a **tarefa de avançar sobre os caracteres espaço em branco e caracteres dentro de comentários**. A **tarefa** desse objeto é ler os **caracteres de entrada** até que todos os espaços em branco e os caracteres entre comentários tenham sido lidos.

PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



Modelos de Processos de Desenvolvimento de Software (Modelo em Cascata, Espiral e Prototipagem). Classificação de Requisitos de Software (funcionais e não funcionais). Técnicas de Levantamento de Requisitos. Projeto de arquitetura. Projeto e Implementação. Reuso de Software. Engenharia baseada em componentes. Engenharia de Software distribuído. Arquitetura orientada a serviço. Estudo de Viabilidade. Técnicas de documentação. Metodologias para desenvolvimento de sistemas.

BIBLIOGRAFIA

BIBLIOGRAFIA BÁSICA:

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

PRESSMAN, Roger; MAXIM, Bruce. Engenharia de Software. 8 ed. São Paulo: McGraw Hill Brasil, 2016.

SOMMERVILLE, Ian. Engenharia De Software. 10 ed. São Paulo: Pearson Brasil, 2019.

BIBLIOGRAFIA COMPLEMENTAR:

LARMAN, Craig. Utilizando UML e padrões. 3 ed. Porto Alegre: Bookman, 2007.

REZENDE, Denis Alcides. Engenharia de software e sistemas de informação. 3 ed. Rio de Janeiro: Brasport, 2005.

WASLAWICK Raul. Análise e Projeto de Sistemas de Informação Orientados a Objetos. 2 ed. Rio de Janeiro: Elsevier, 2010.



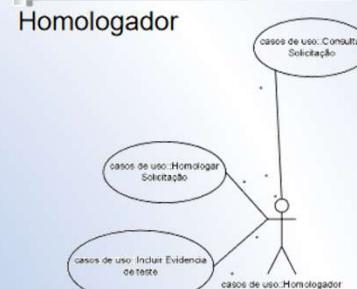


Participe

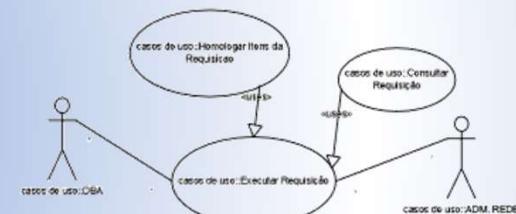
Habilidades



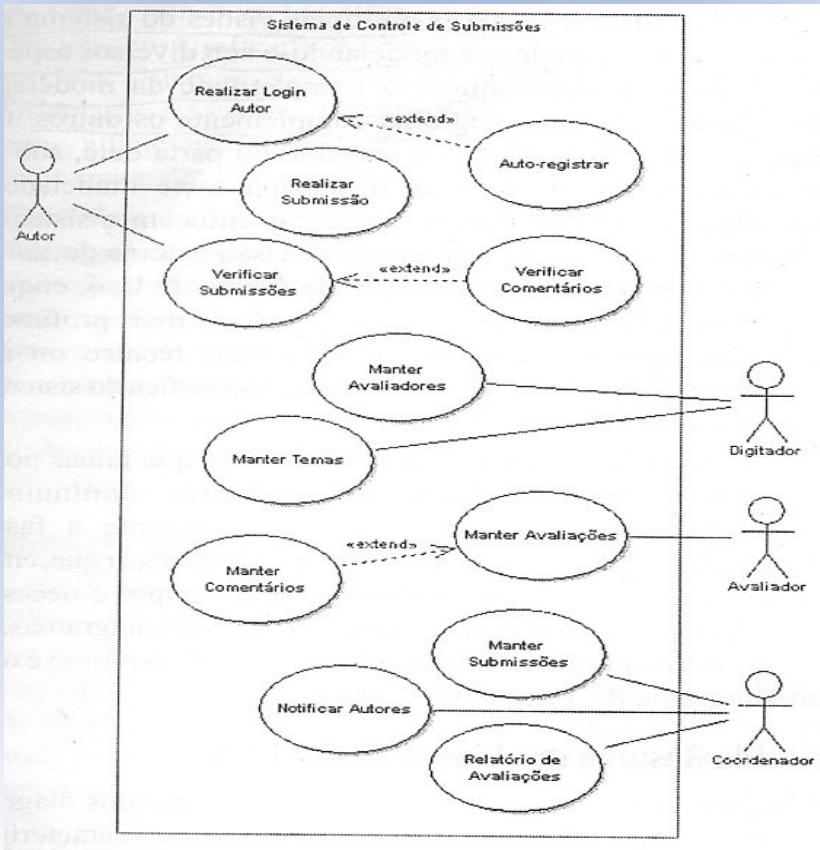
Identifique cada diagrama na UML



Homologador
Executor Sistemas e Rede



Identifique o Diagrama

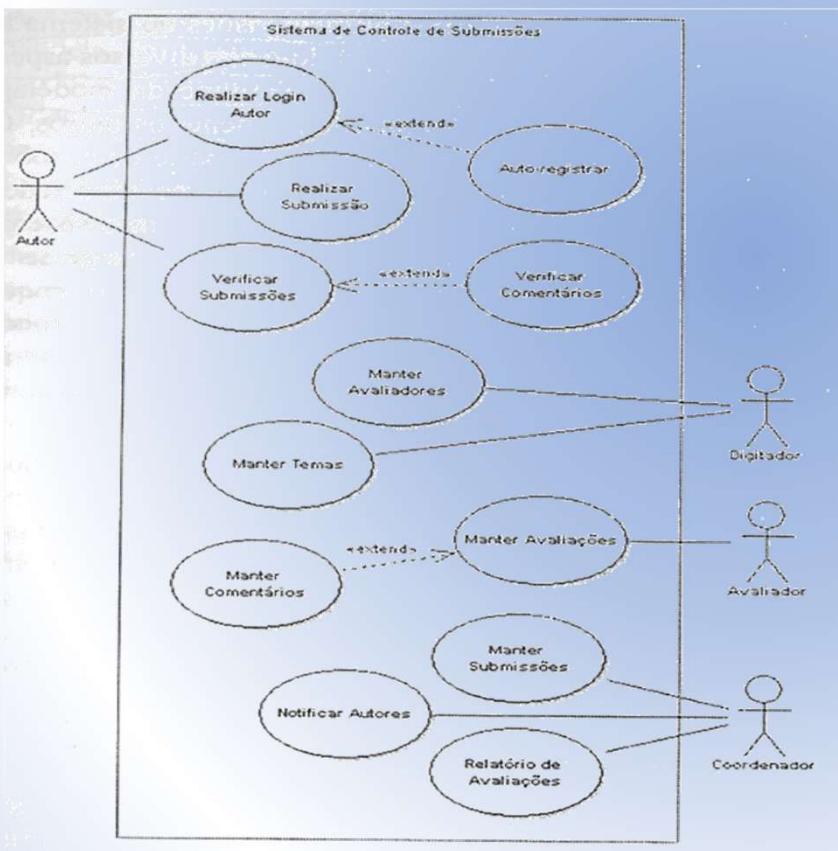


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



1. Diagrama de Caso de Uso:

Usado nas fases de levantamento e análise de requisitos do sistema.

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Descrição do Caso

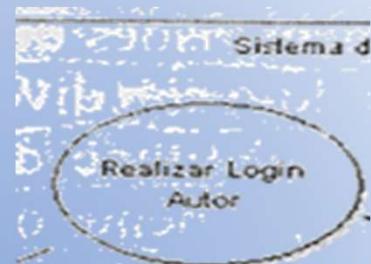


Tabela 3.1 – Documentação do Caso de Uso Realizar Submissão

Nome do Caso de Uso	Realizar Submissão
Caso de Uso Geral	
Ator Principal	Autor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um autor para submeter um trabalho ao congresso
Pré-Condições	O autor precisa estar logado
Pós-Condições	
Ações do Sistema	

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Descrição do Caso

Ações do Ator	Ações do Sistema
1. Solicitar opção de submissão de trabalhos	2. Selecionar temas aceitos pelo congresso
	3. Apresentar tela de submissão contendo os temas aceitos e os tipos de submissão (artigos, mini-cursos e palestras) válidos
4. Selecionar tema	5. Consultar tema selecionado
6. Selecionar tipo de submissão	
7. Informar dados da submissão	
8. Anexar arquivo com o trabalho a ser submetido	
9. Confirmar	
	10. Registrar submissão
	11. Apresentar mensagem de trabalho submetido com sucesso

Engenharia de Requisitos

Prof. Carla A. Lima Reis





Descrição do Caso

Restrições/Validações	submetido com sucesso 1. Todos os campos são obrigatórios 2. É obrigatório anexar o arquivo do trabalho submetido
-----------------------	---

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Descrição do Caso

Diagrama de ...

Tabela 3.1 – Documentação do Caso de Uso Realizar Submissão

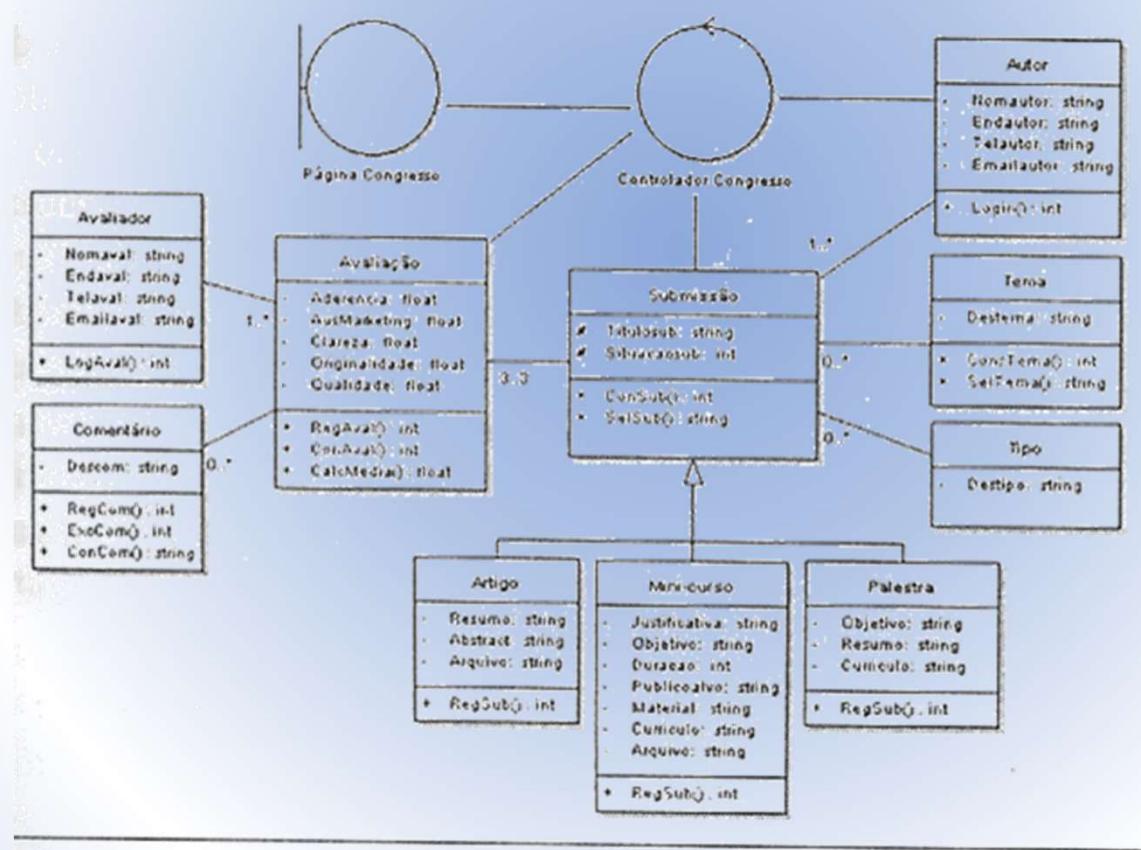
Nome do Caso de Uso	Realizar Submissão
Caso de Uso Geral	
Autor Principal	Autor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um autor para submeter um trabalho ao congresso
Pré-Condições	O autor precisa estar logado
Pós-Condições	
Ações do Ator	Ações do Sistema
1. Solicitar opção de submissão de trabalhos	
	2. Selecionar temas aceitos pelo congresso
	3. Apresentar tela de submissão contendo os temas aceitos e os tipos de submissão (artigos, mini-cursos e palestras) válidos
4. Selecionar tema	5. Consultar tema selecionado
6. Selecionar tipo de submissão	
7. Informar dados da submissão	
8. Anexar arquivo com o trabalho a ser submetido	
9. Confirmar	
	10. Registrar submissão
	11. Apresentar mensagem de trabalho submetido com sucesso
Restrições/Validações	1. Todos os campos são obrigatórios 2. É obrigatório anexar o arquivo do trabalho submetido

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama

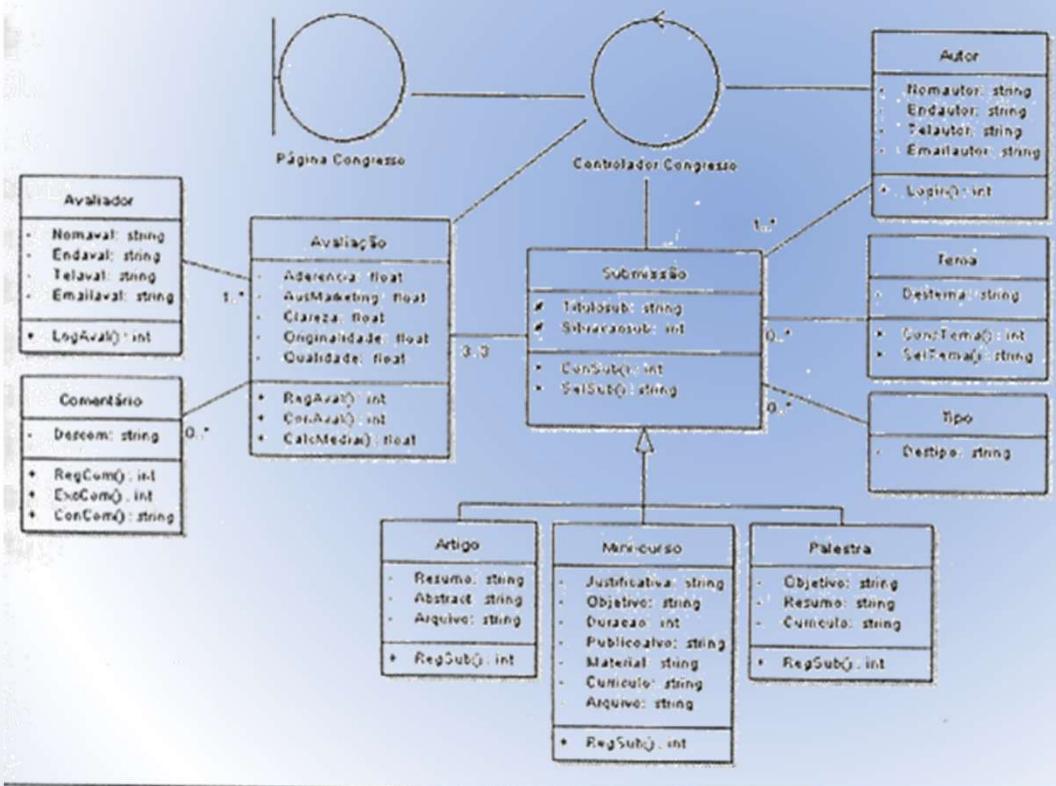


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



2. Diagrama de Classes:

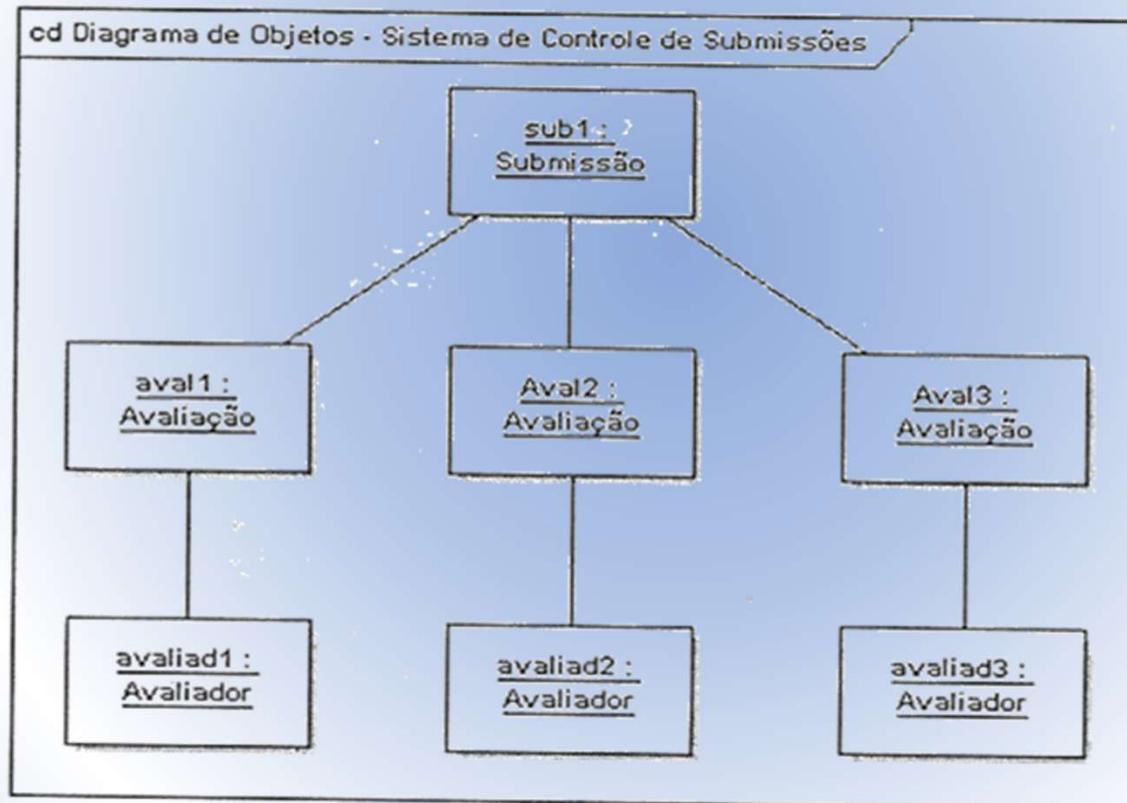
Define a estrutura das classes utilizadas pelo sistema, determinando os atributos e métodos.

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama

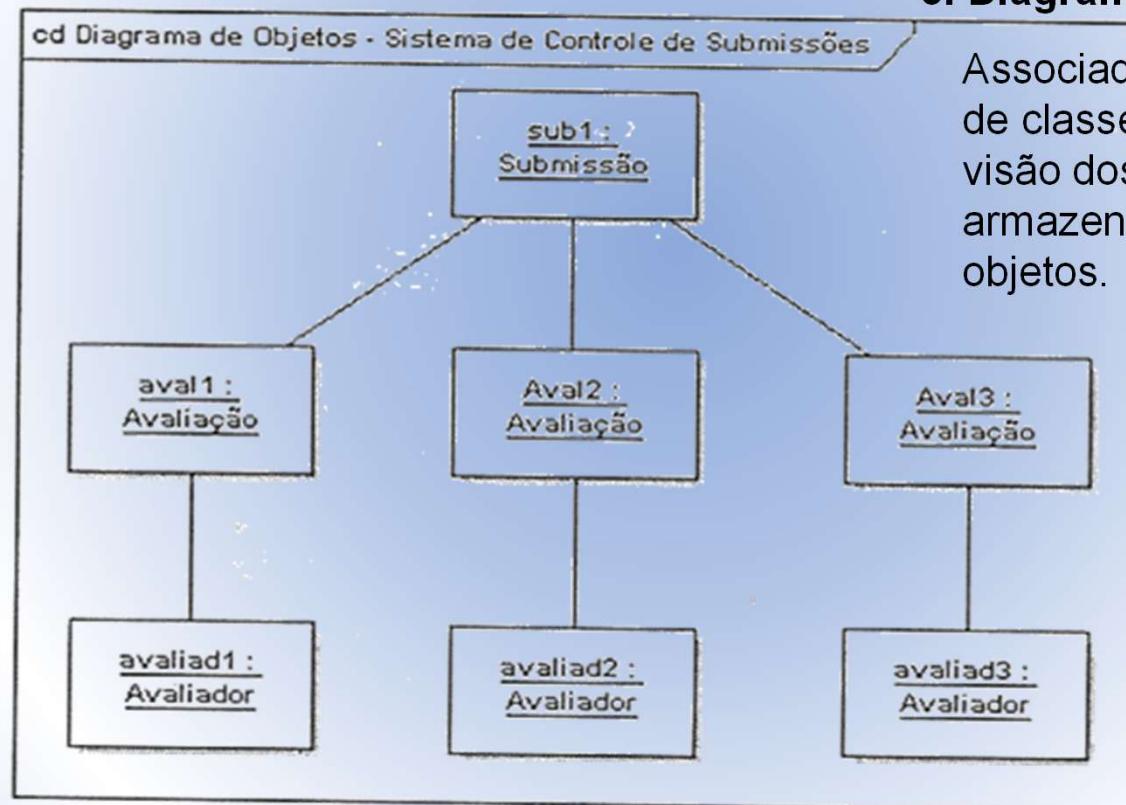


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



3. Diagrama de Objetos:

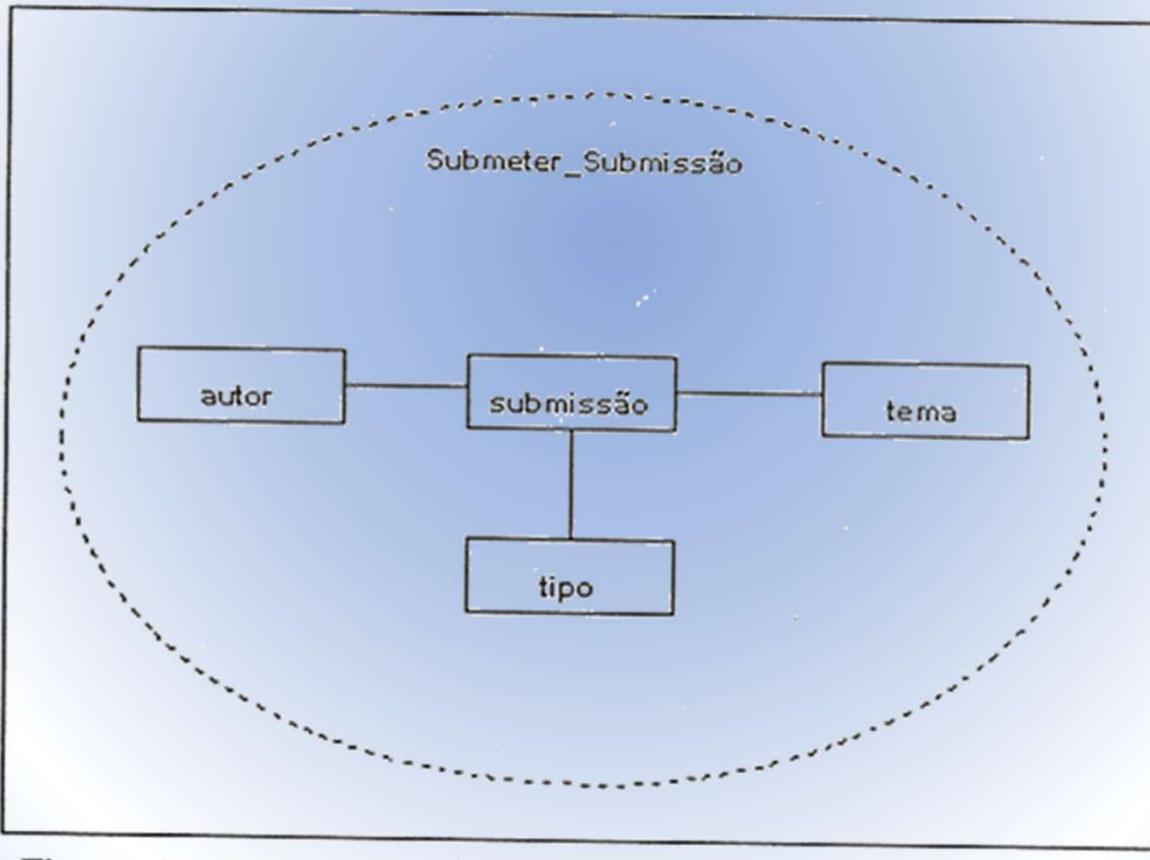
Associado ao diagrama de classes fornece a visão dos valores armazenados pelos objetos.

Engenharia de Requisitos

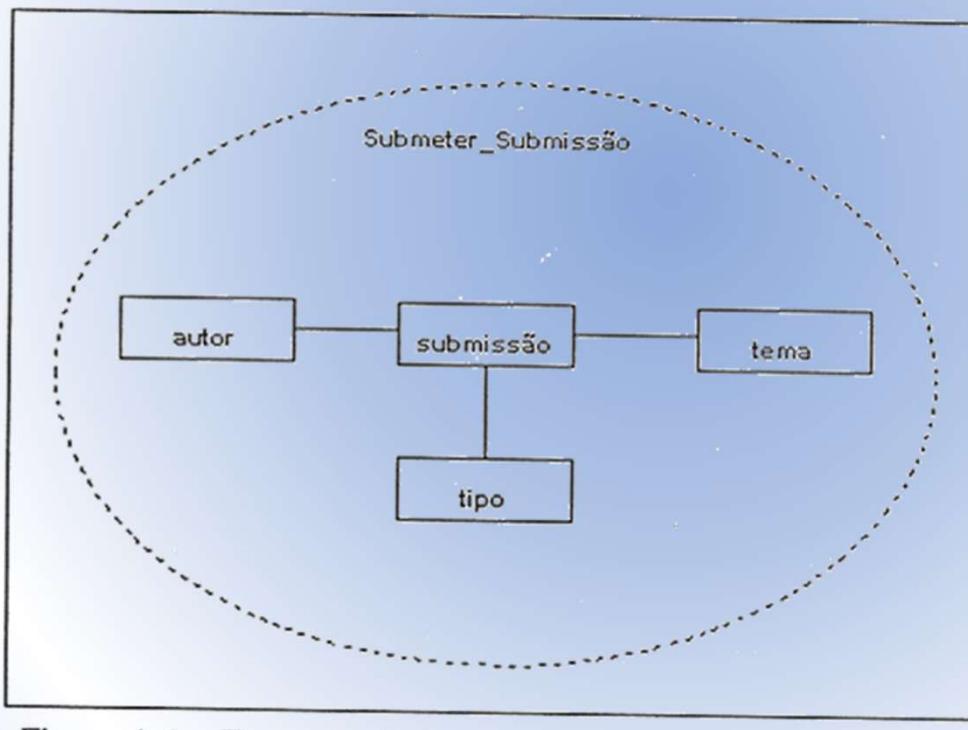
Prof. Carla A. Lima Reis



Identifique o Diagrama



Identifique o Diagrama

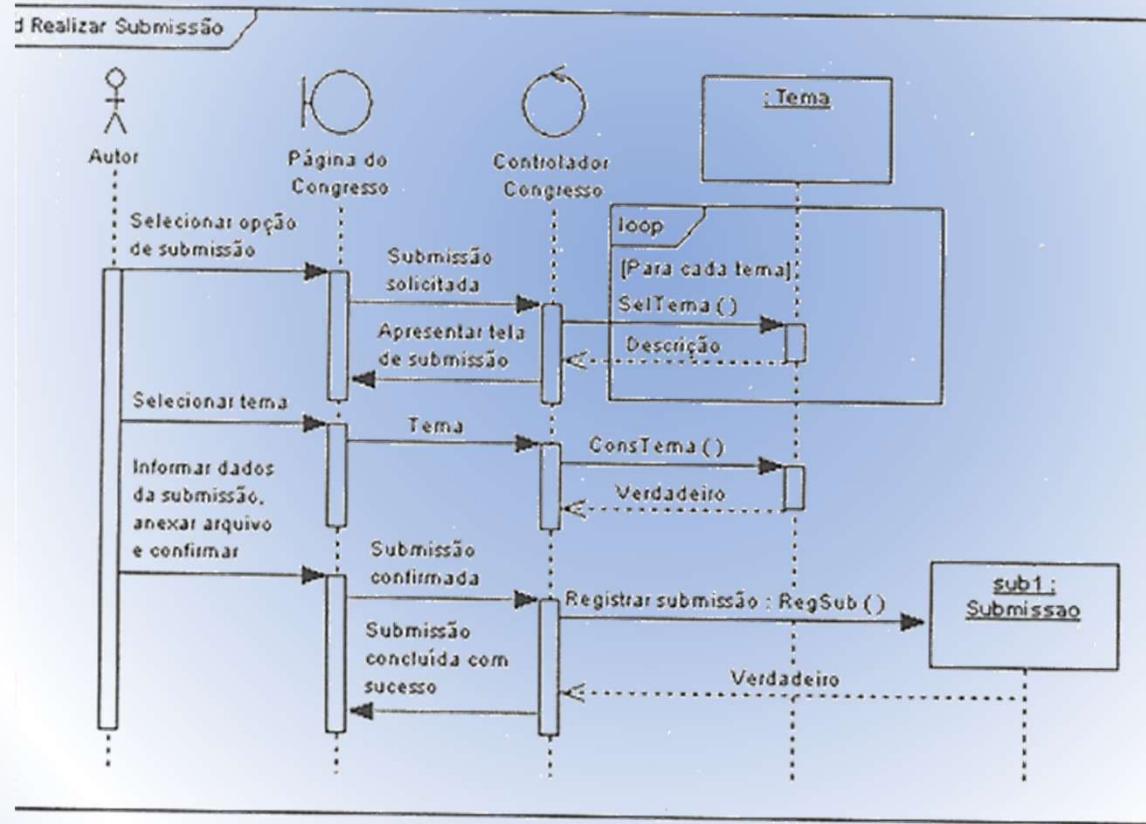


4. Diagrama de Estrutura Composta:

Usado para modelar colaborações, descreve uma visão de um conjunto de instâncias que cooperam entre si para executar uma função específica.



Identifique o Diagrama

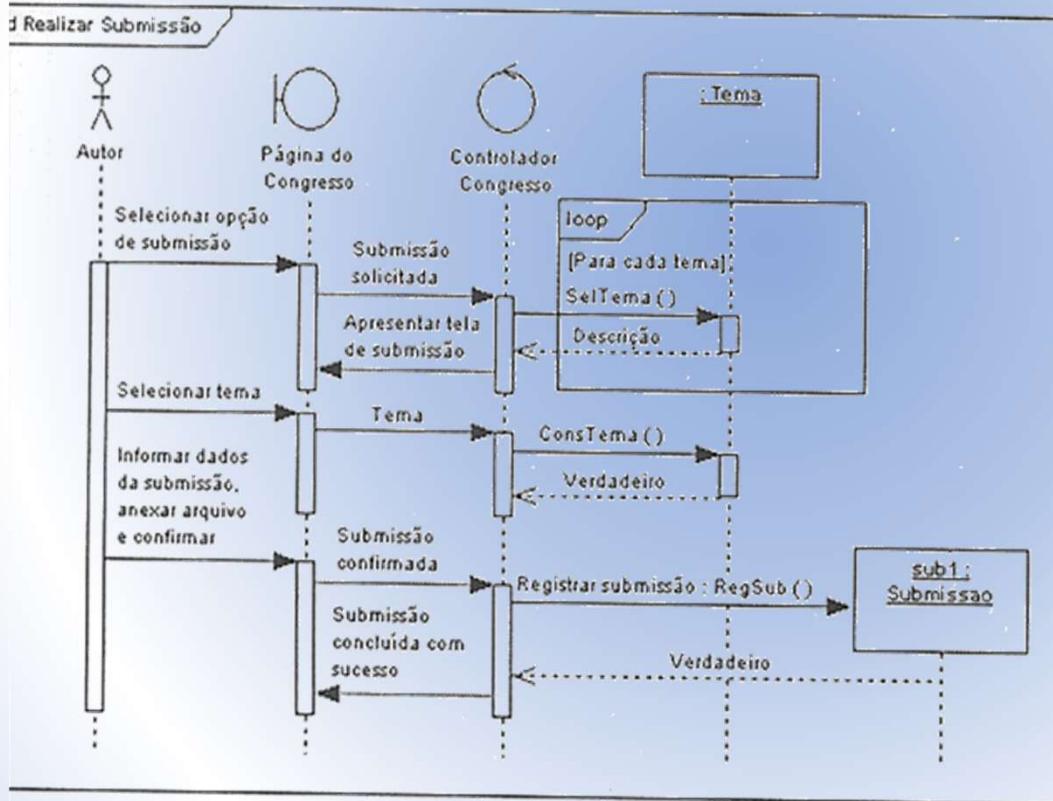


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



5. Diagrama de Sequencia:

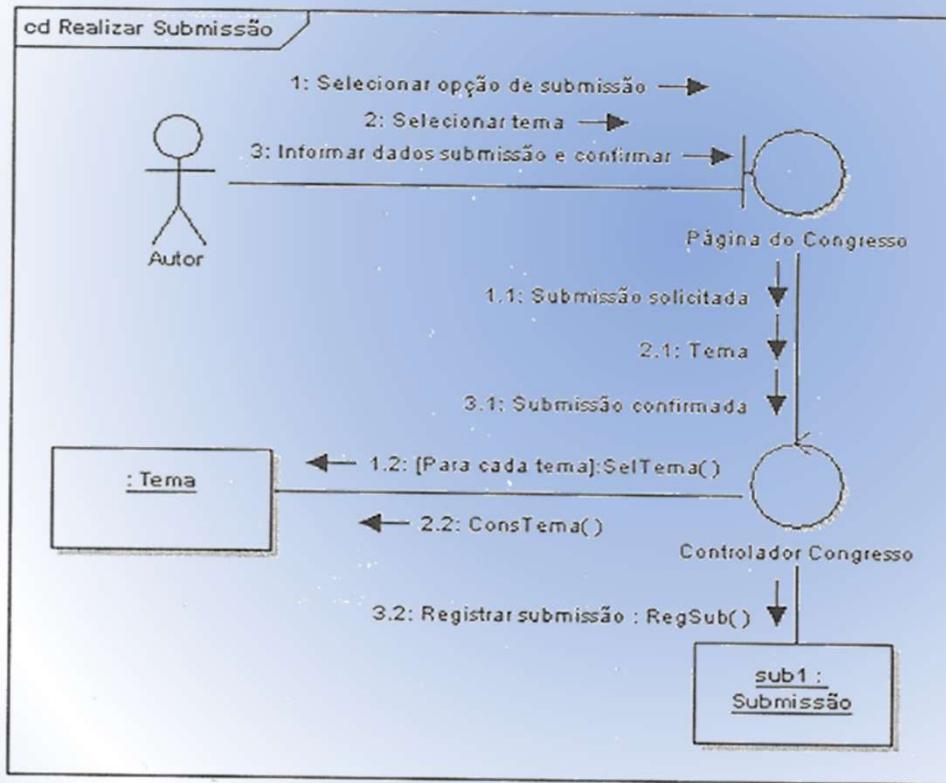
Preocupa-se com a ordem temporal em que as mensagens são trocados entre os objetos envolvidos..

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama

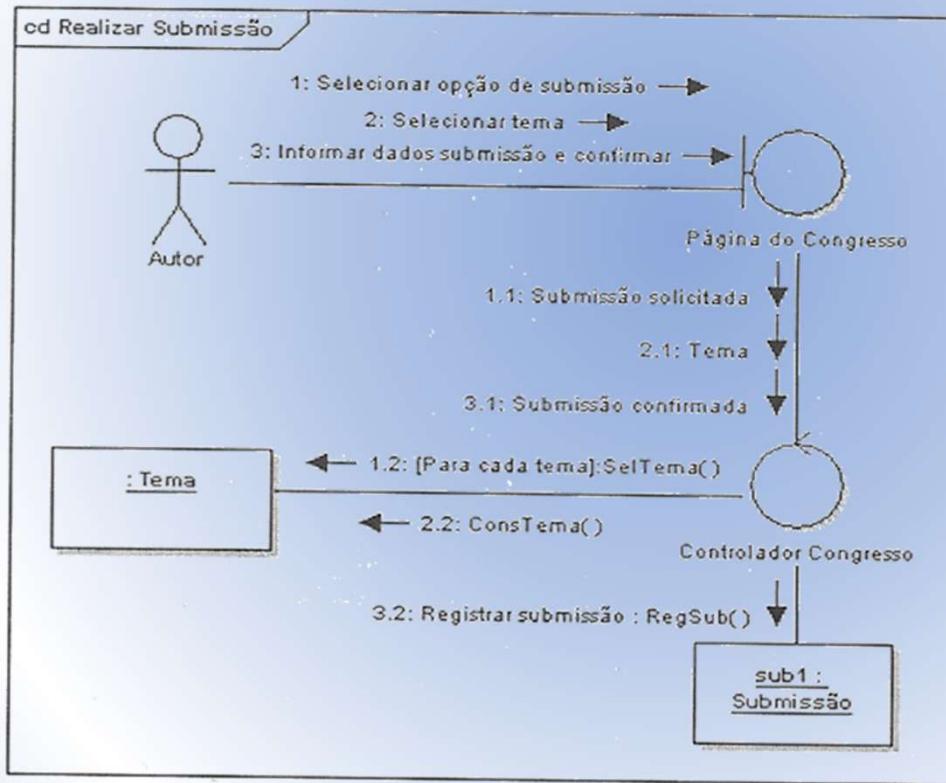


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



6. Diagrama de Comunicação:

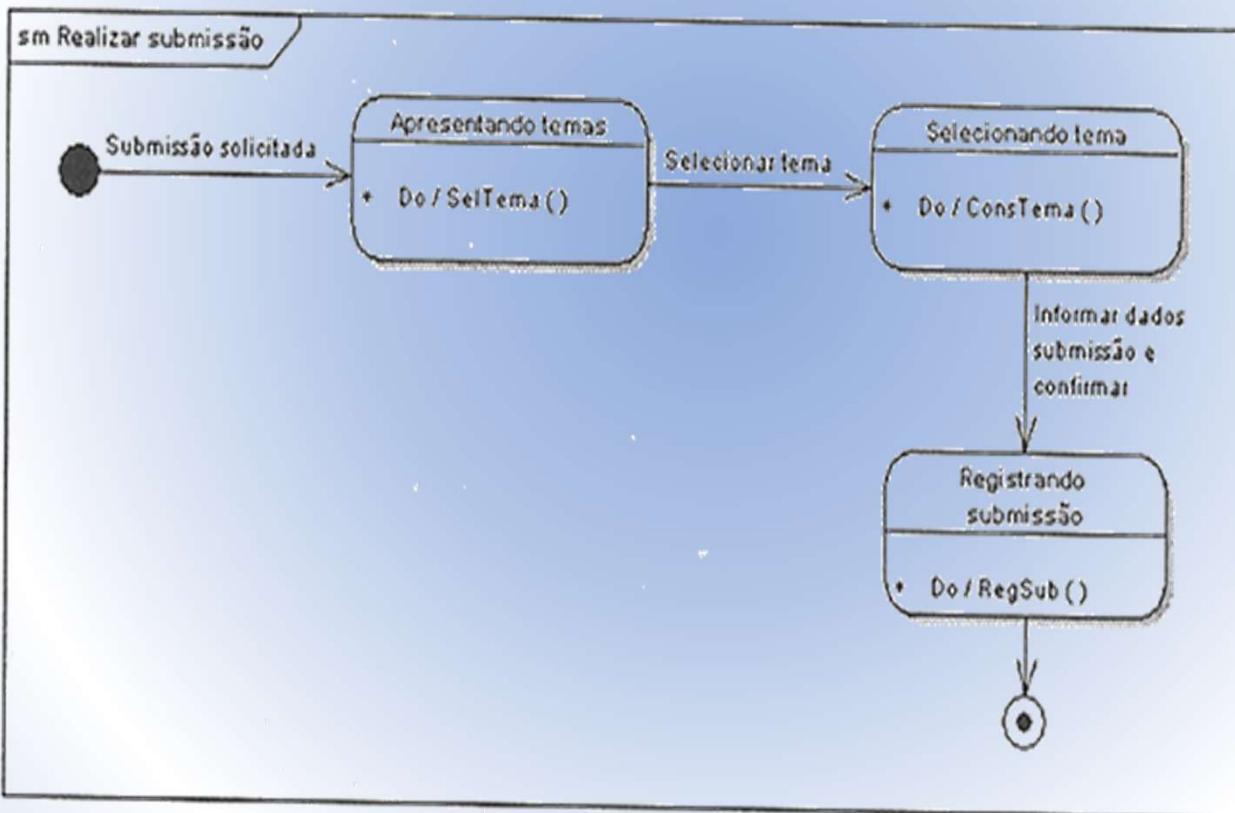
Completa o diagrama de sequencia e a sua diferença esta no fato de não se preocupar com a temporalidade do processo.

Engenharia de Requisitos

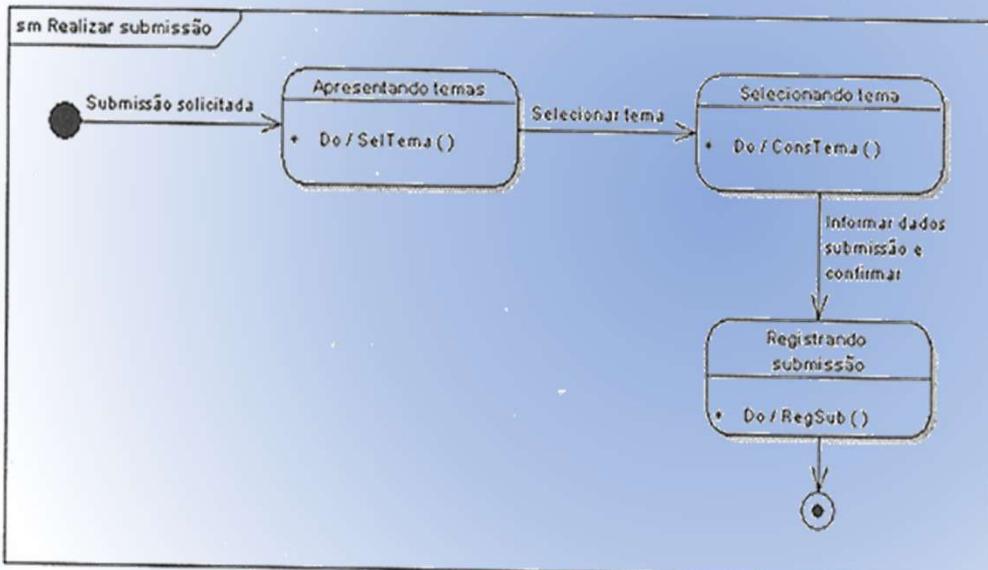
Prof. Carla A. Lima Reis



Identifique o Diagrama



Identifique o Diagrama

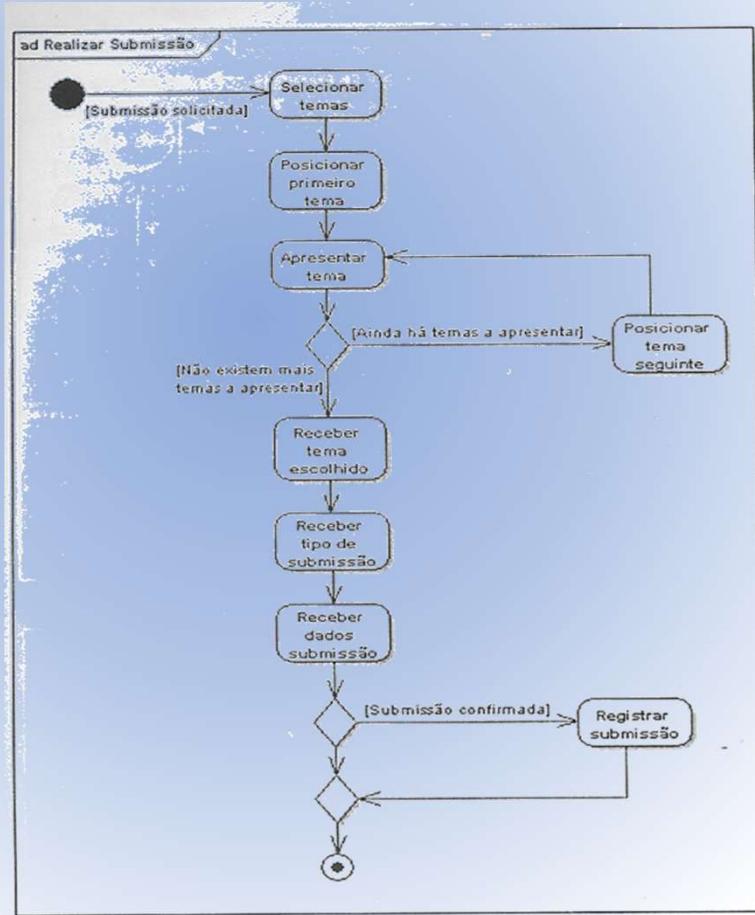


7. Diagrama de Maquina de Estados:

Este diagrama procura acompanhar as mudanças sofridas nos estados de uma instancia de uma classe.



Identifique o Diagrama

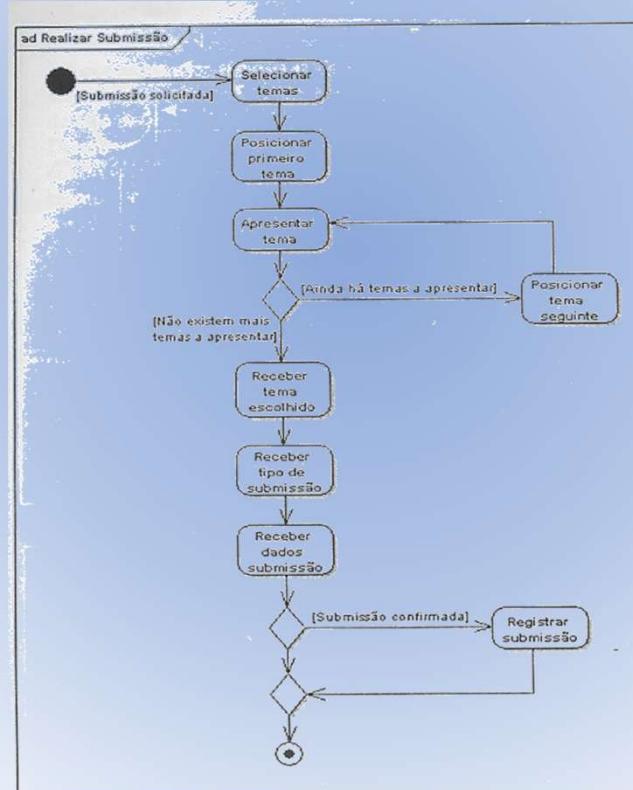


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



8. Diagrama de Atividade:

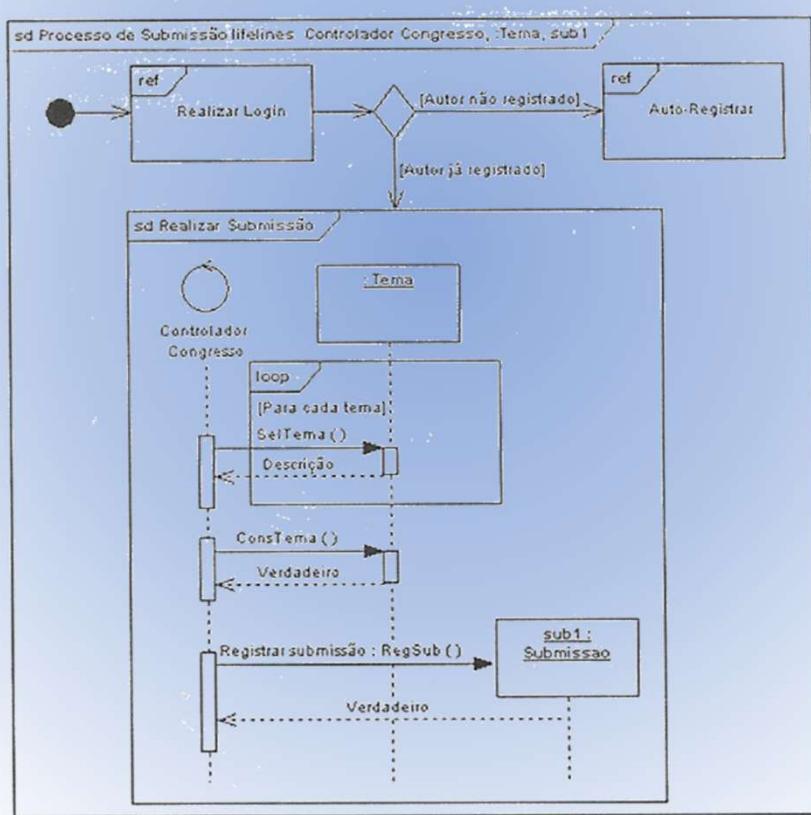
Preocupa-se em descrever os passos percorridos para a conclusão de uma atividade específica.

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama

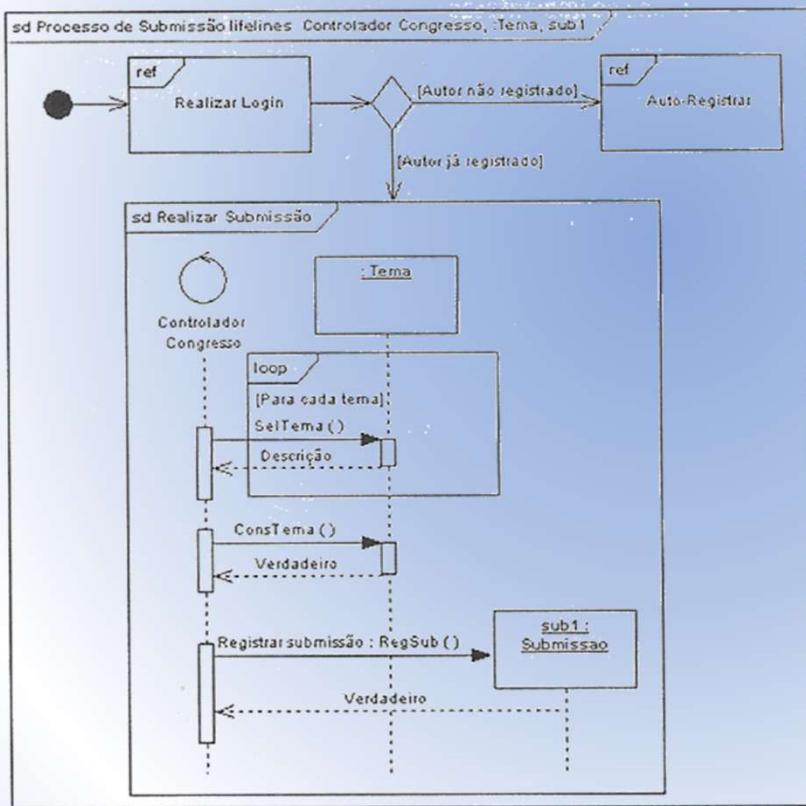


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



9. Diagrama de Interação Geral:

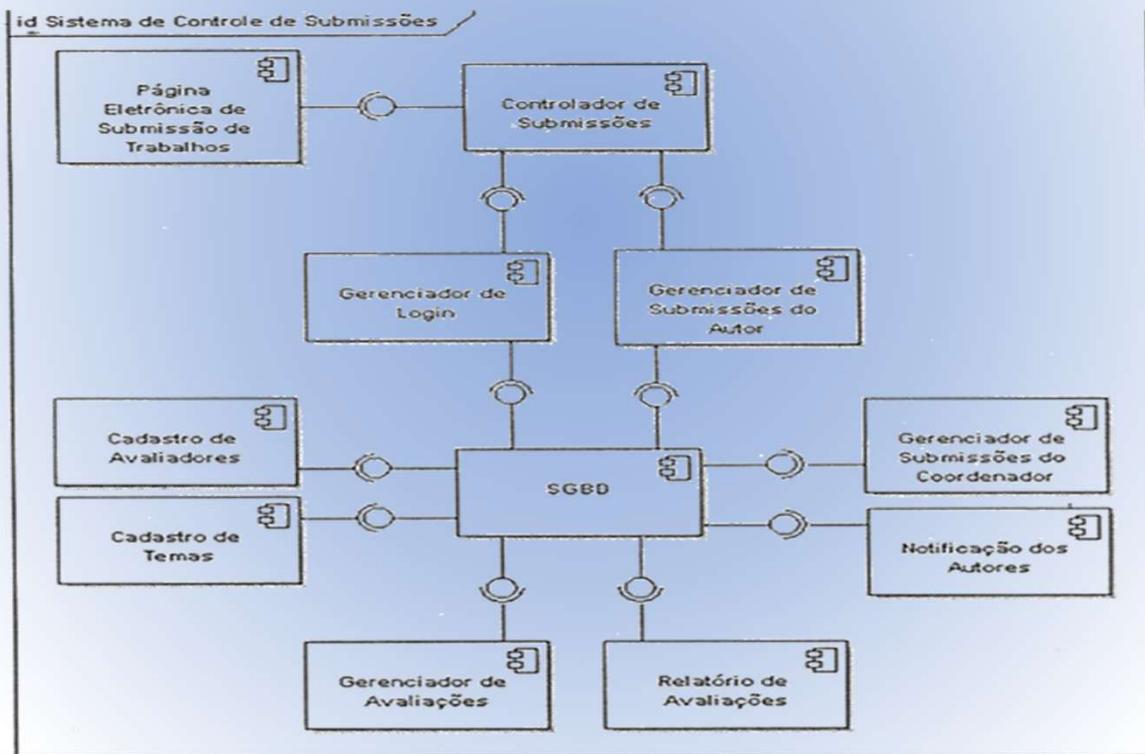
Fornece uma visão geral dentro de um sistema ou processo de negócio.

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama

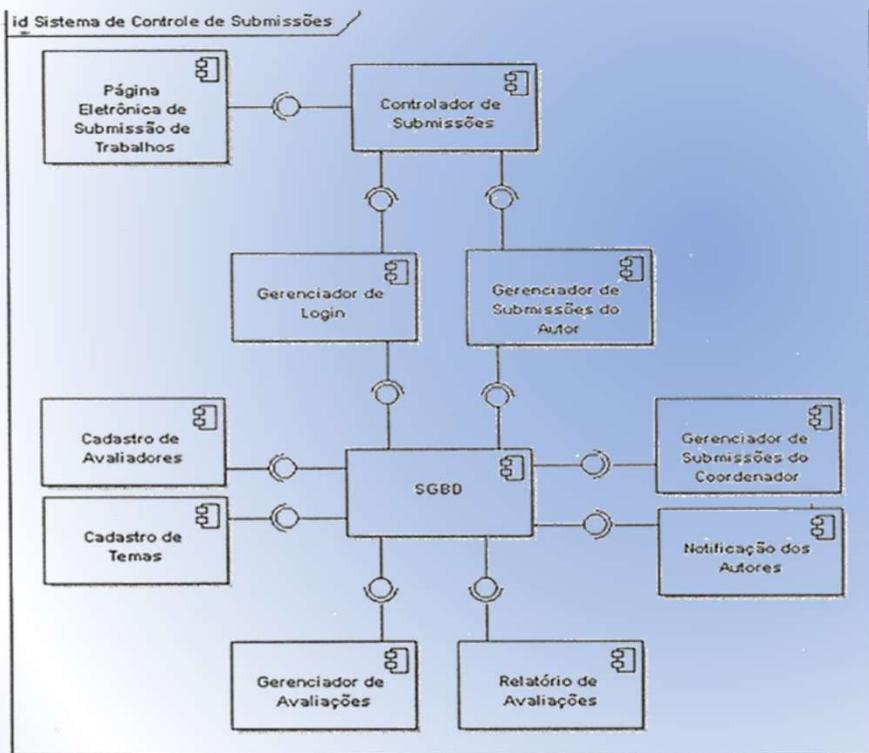


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



10. Diagrama de Componentes:

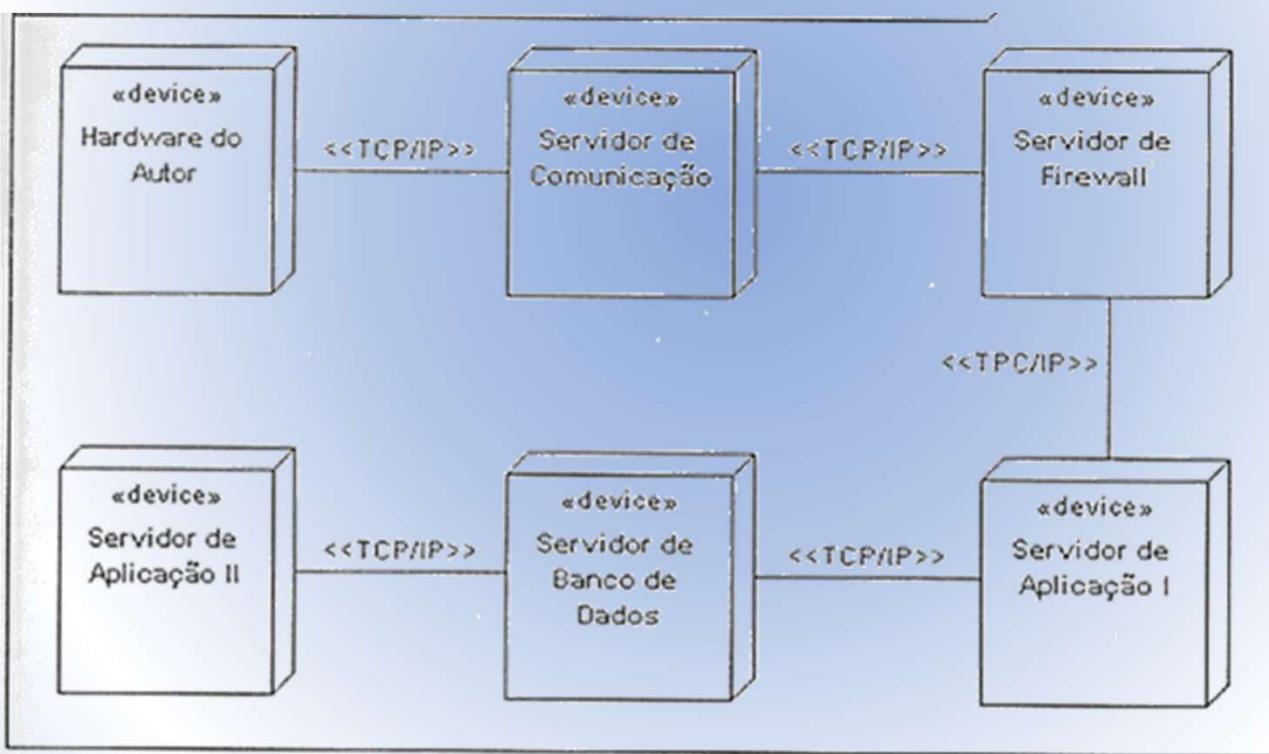
Esta associado a linguagem de programação e representa os componentes do sistema, tais como: código fonte, bibliotecas, formulários, arquivos de ajuda.

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama

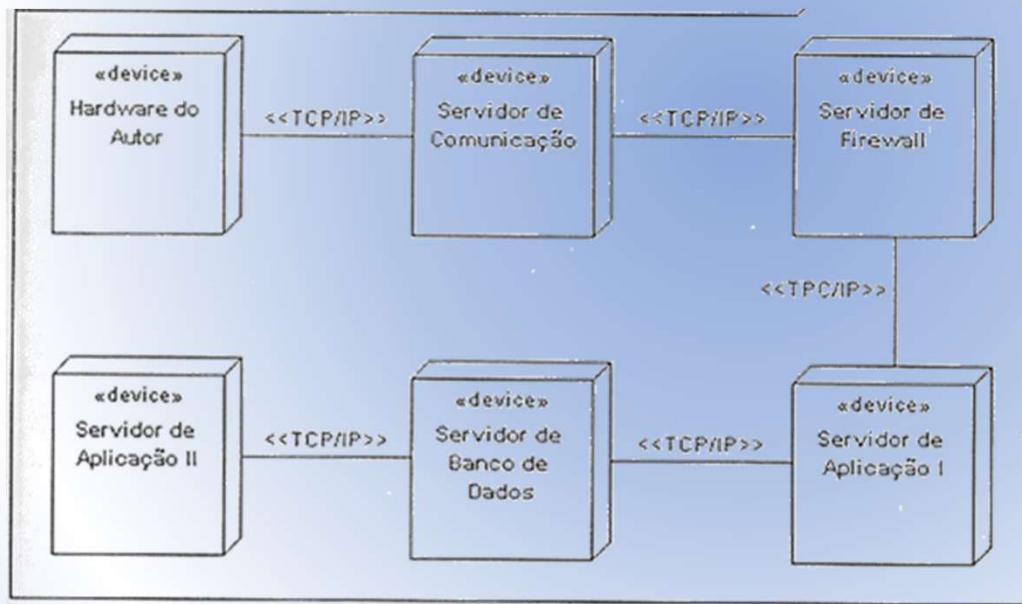


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



11. Diagrama de Implantação:

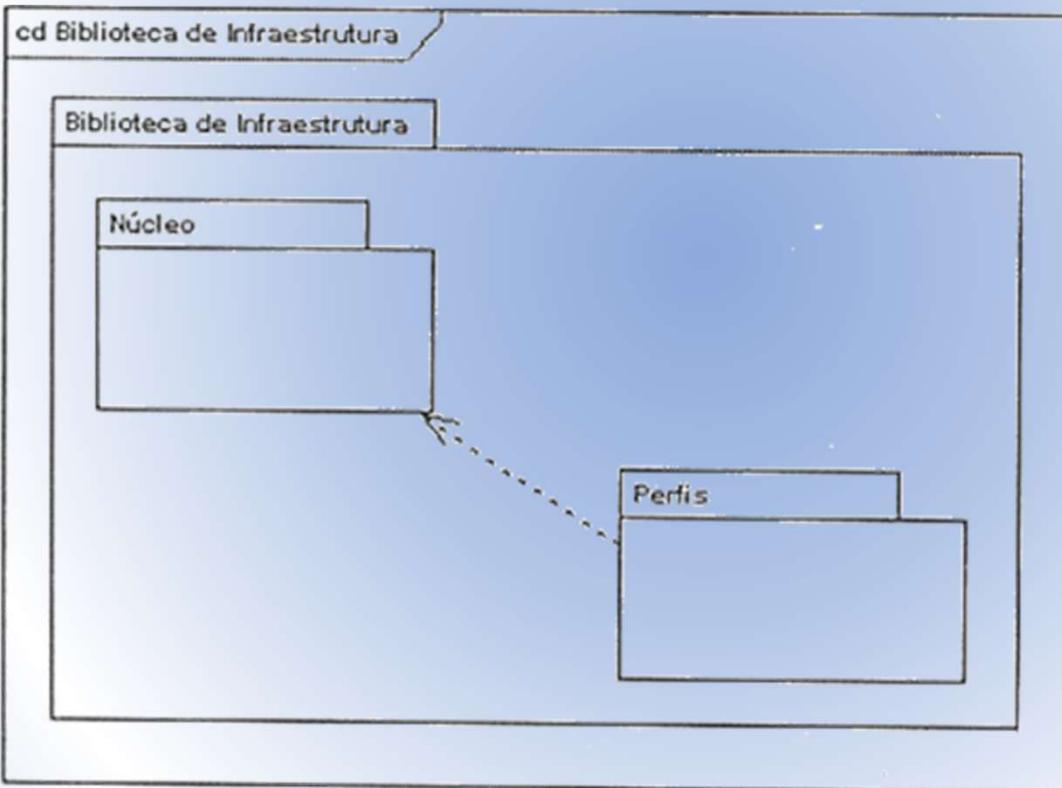
Determina a representação das características físicas, tais como: servidores, estações, topologias, protocolos de comunicações.

Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama

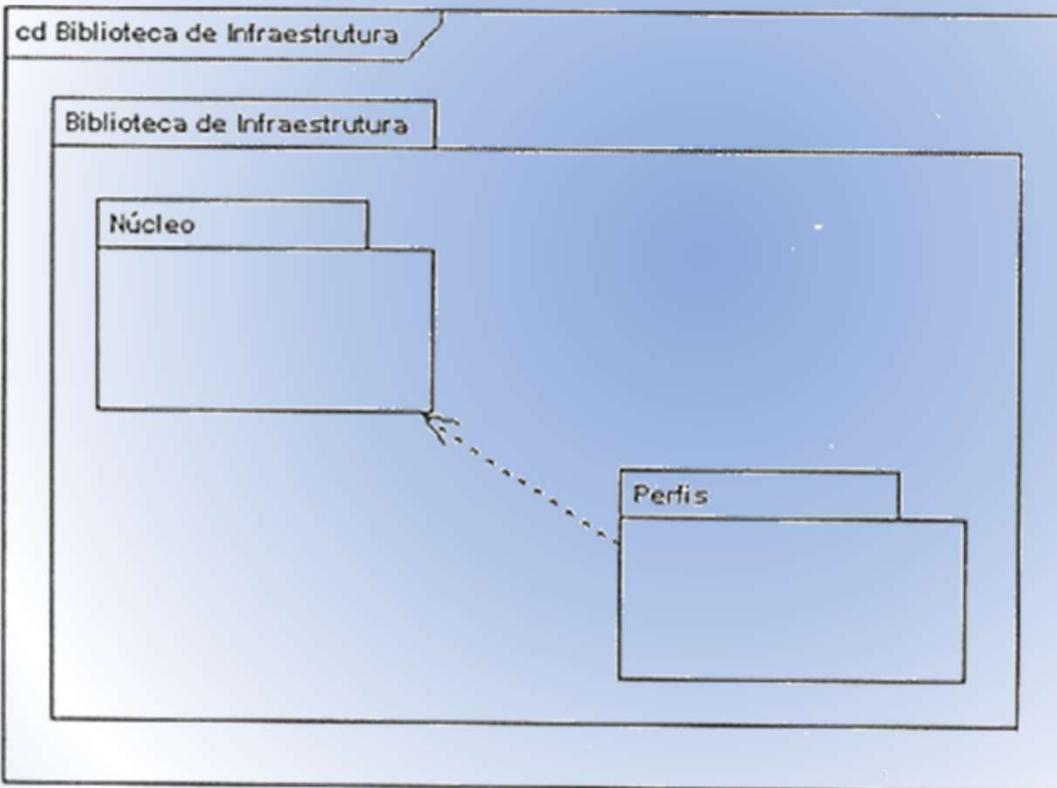


Engenharia de Requisitos

Prof. Carla A. Lima Reis



Identifique o Diagrama



12. Diagrama de Pacotes:

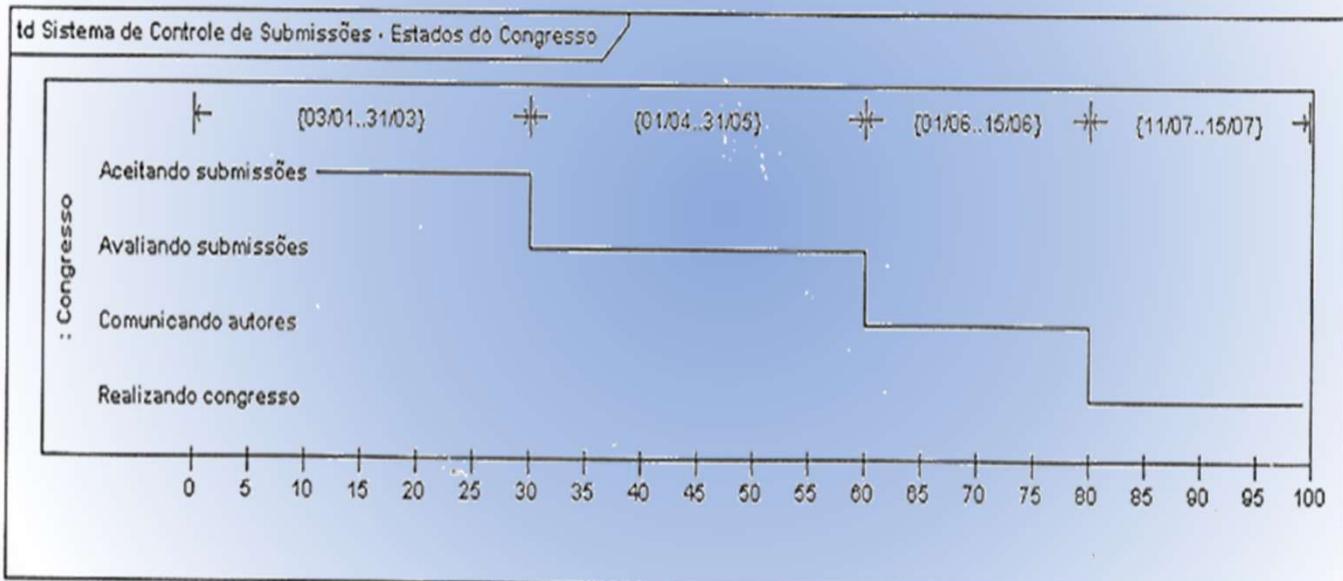
Representa os subsistemas ou sub-módulos englobados por um sistema.

Engenharia de Requisitos

Prof. Carla A. Lima Reis



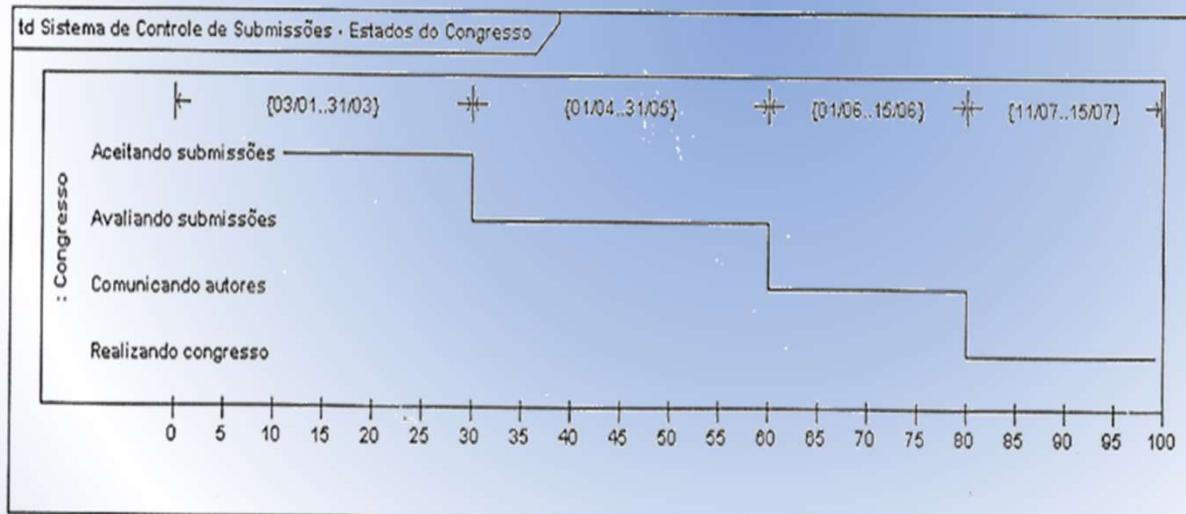
Identifique o Diagrama



Identifique o Diagrama

13. Diagrama de Tempo:

Utilizado para demonstrar a mudança no estado de um objeto no tempo em resposta a eventos externos.



UML





FERRAMENTAS



Word



Excel



Microsoft Visio é uma ferramenta de criação de Fluxogramas e Diagramas Profissionais

Microsoft Project é uma ferramenta de gestão de projetos (PMO) para simplificar negócios.



Visio



draw.io



dia



Project

P! Pesquise sobre ferramentas novas.





Ferramentas iniciais da Engenharia de Software:

The image shows a dual-screen setup. The bottom screen displays Microsoft Project version 2010, showing a Gantt chart for a project named 'Project1'. The top screen shows the Microsoft 365 Start screen with various application icons like Outlook, Word, PowerPoint, SharePoint, OneDrive, Excel, OneNote, Teams, Forms, and Visio.

Microsoft Project - Project1

File Edit View Insert Format Tools Project Report Collaborate Window Help

No Group

Type a question for help

Task Name Duration Start Finish Predecessors Resource Names Aug '21 08 Aug '21 15 Aug '21 22 Aug '21 29 Aug '21 05 Sep '21 12 Sep '21

Task Name	Duration	Start	Finish	Predecessors	Resource Names	Aug '21	08 Aug '21	15 Aug '21	22 Aug '21	29 Aug '21	05 Sep '21	12 Sep '21																																			
						M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S

Visio

Project



Microsoft Visio

Template Categories

- Getting Started
- Samples
- Business** (selected)
- Engineering
- Flowchart
- General
- Maps and Floor Plans
- Network
- Schedule
- Software and Database

Business

Featured Templates

- Brainstorming Diagram** (selected)
- Organization Chart
- PivotDiagram

Other Templates

- Audit Diagram
- Basic Flowchart
- Cause and Effect Diagram
- Charts and Graphs
- Cross Functional Flowchart
- Data Flow Diagram
- EPC Diagram
- Fault Tree Analysis Diagram
- ITIL Diagram
- Marketing Charts and Diagrams

Brainstorming Diagram Template

Create brainstorming diagrams (graphical representations of thought processes) for planning, problem solving, decision making, and brainstorming.

Measurement Units: US units Metric

Create

Processo de Negócio





Ferramentas iniciais da Engenharia de Software:

Microsoft Visio

File Edit View Insert Format Tools Data Shape Window Help

Type a question for h

Template Categories

- Getting Started
- Samples
- Business
- Engineering
- Flowchart**
- General
- Maps and Floor Plans
- Network
- Schedule
- Software and Database

Flowchart

Featured Templates

- Basic Flowchart**
- Cross Functional Flowchart
- Work Flow Diagram

Other Templates

- Data Flow Diagram
- IDEFO Diagram
- SDL Diagram

Basic Flowchart Template

Create flowcharts, top-down diagrams, information tracking diagrams, process planning diagrams, and structure prediction diagrams. Contains connectors and links.

Measurement Units: US units Metric

Create



Fluxo de Processos



 GOVERNO DO ESTADO
SÃO PAULO

Microsoft Visio

File Edit View Insert Format Tools Data Shape Window Help

Type a question for help

Template Categories

- Getting Started
- Samples
- Business
- Engineering
- Flowchart
- General
- Maps and Floor Plans
- Network**
- Schedule
- Software and Database

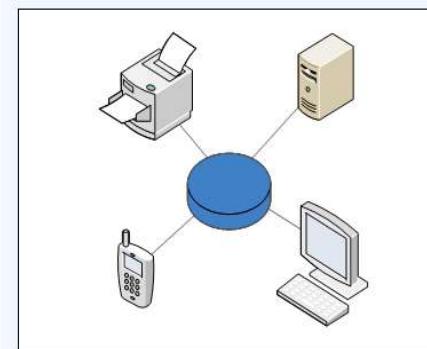
Network

Featured Templates

- Basic Network Diagram**
- Detailed Network Diagram
- Web Site Map

Other Templates

- Active Directory
- Conceptual Web Site
- LDAP Directory
- Rack Diagram



Basic Network Diagram Template

Create simple network designs and network architecture diagrams, using basic network and computer equipment shapes.

Measurement Units: US units Metric

Create

Redes



CPS Centro Paula Souza

**GOVERNO DO ESTADO
SÃO PAULO**

Microsoft Visio

Template Categories

- Getting Started
- Samples
- Business
- Engineering
- Flowchart
- General
- Maps and Floor Plans
- Network
- Schedule
- Software and Database**

Software and Database

Featured Templates

- Database Model Diagram**
- UML Model Diagram
- Windows XP User Interface

Other Templates

- COM and OLE
- Conceptual Web Site
- Data Flow Model Diagram
- Enterprise Application
- Express-G
- Jackson
- ORM Diagram
- Program Structure
- ROOM
- Web Site Map

Database Modeling Template

Document, design, generate*, and update* databases. Supports IDEF1X and relational notations. *Generate and update are features of the Visual Studio edition.

Measurement Units: US units Metric

Create



Base de Dados



CPS Centro Paula Souza

**GOVERNO DO ESTADO
SÃO PAULO**

Microsoft Visio

File Edit View Insert Format Tools Data Shape Window Help

Type a question for help

Template Categories

- Getting Started
- Samples
- Business
- Engineering**
- Flowchart
- General
- Maps and Floor Plans
- Network
- Schedule
- Software and Database

Engineering

All Templates

Basic Electrical

Circuits and Logic Fluid Power Industrial Control Systems Part and Assembly Drawing

Piping and Instrumentati... Process Flow Diagram Systems

Basic Electrical Template

Create schematic, one-line, and wiring diagrams and blueprints. Contains shapes for switches, relays, transmission paths, semiconductors, circuits, and tubes.

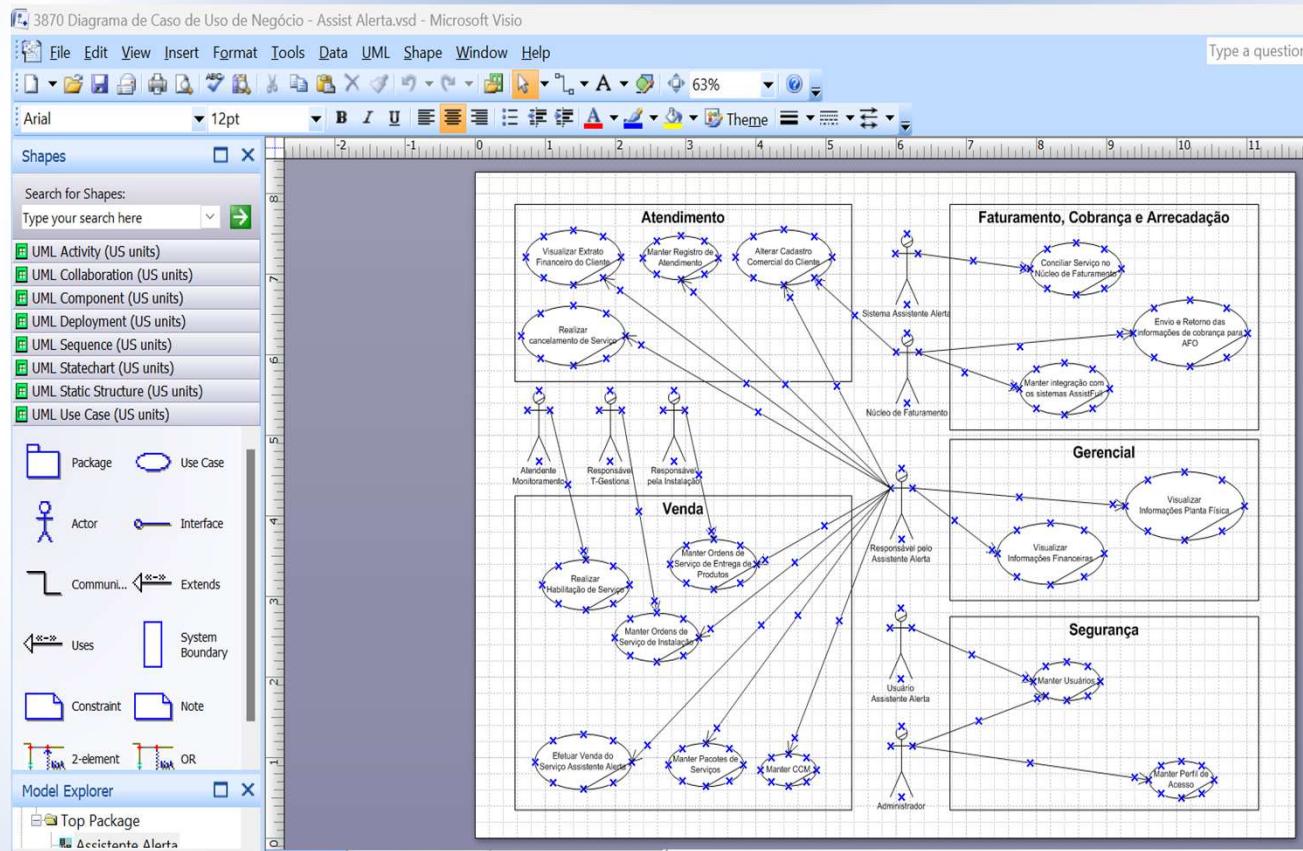
Measurement Units: US units Metric

Create

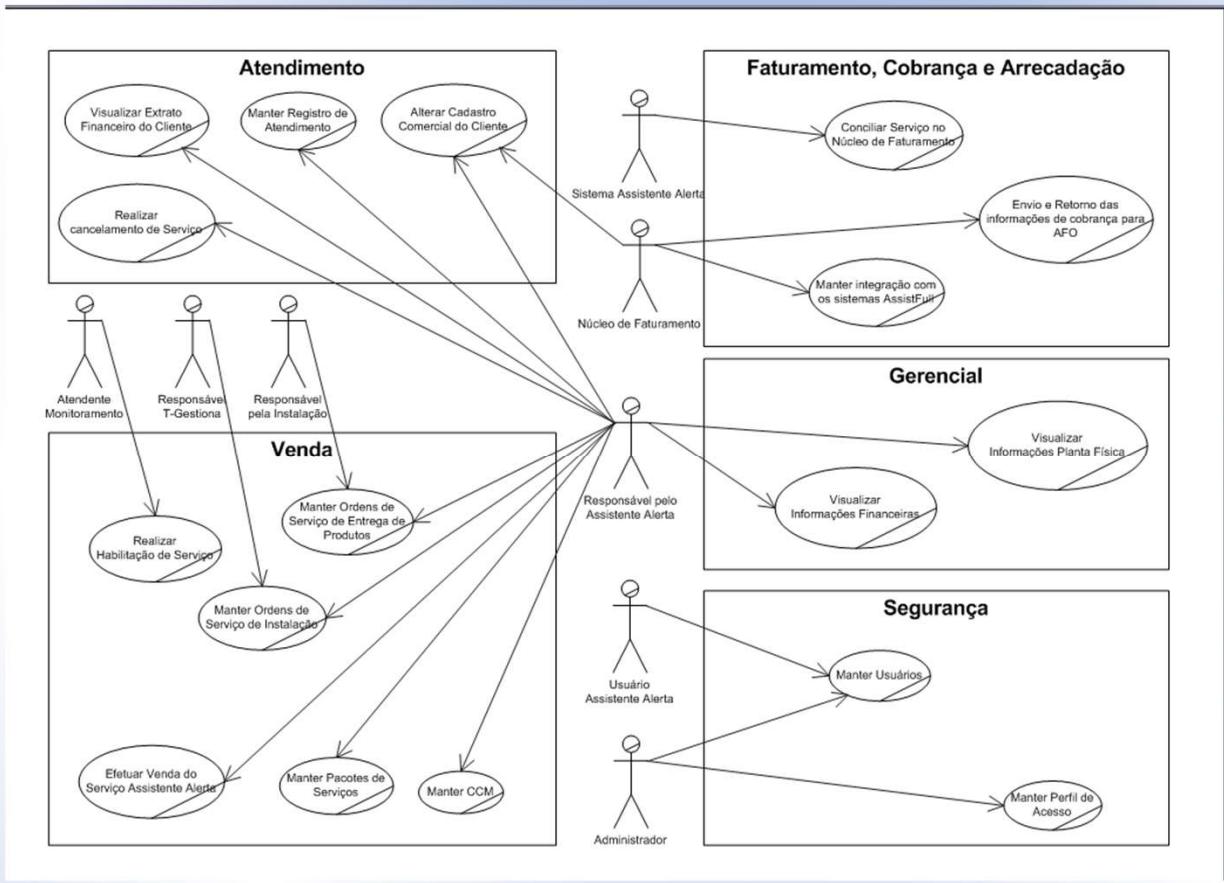
Engenharia



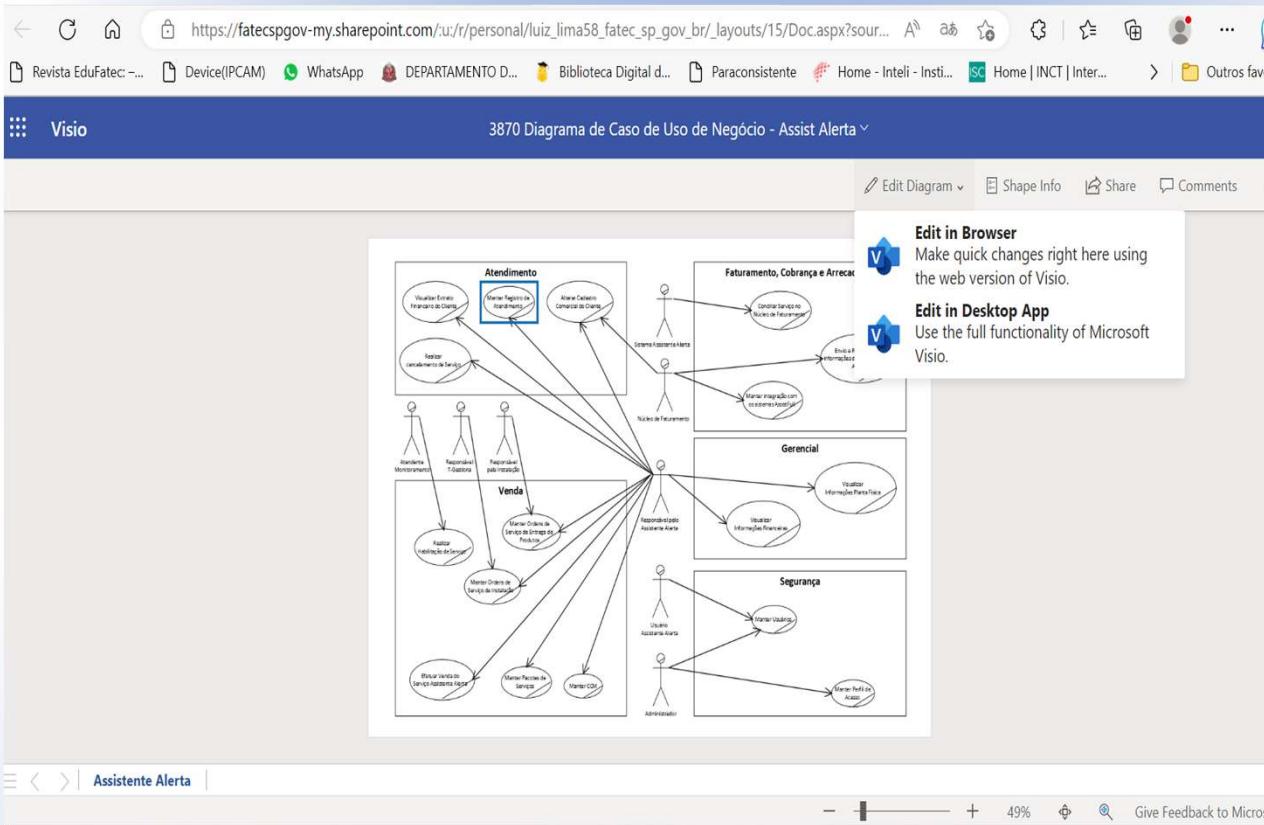
Ferramentas iniciais da Engenharia de Software:



Ferramentas iniciais da Engenharia de Software:

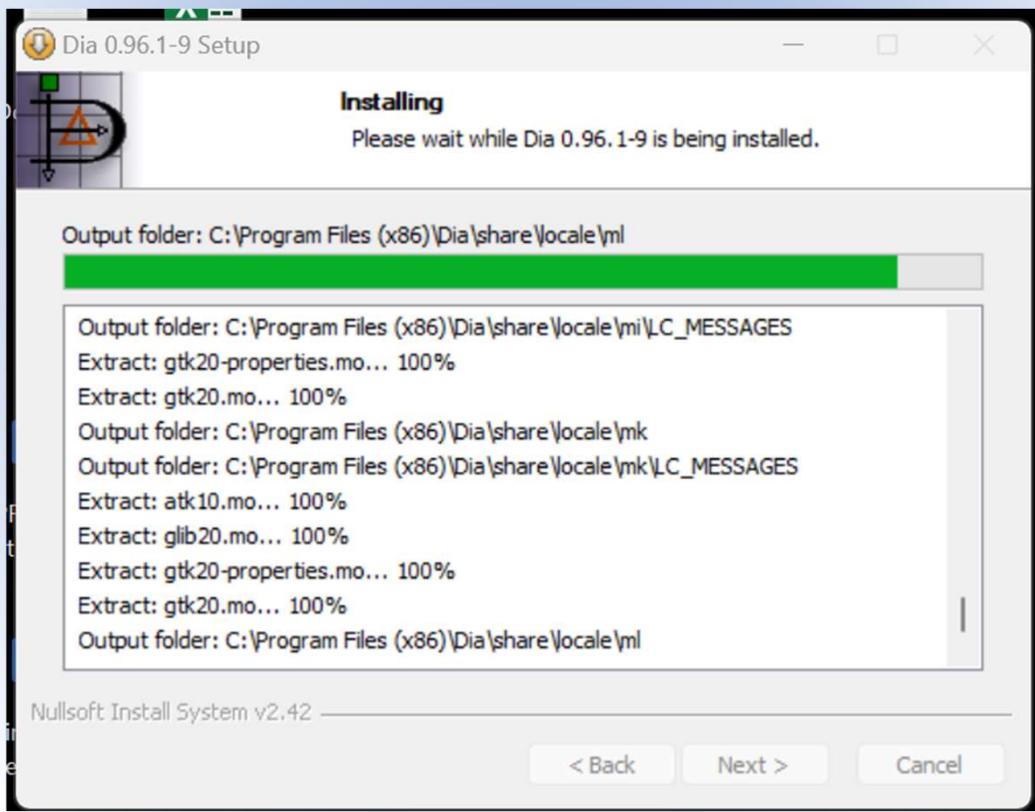


Ferramentas iniciais da Engenharia de Software:

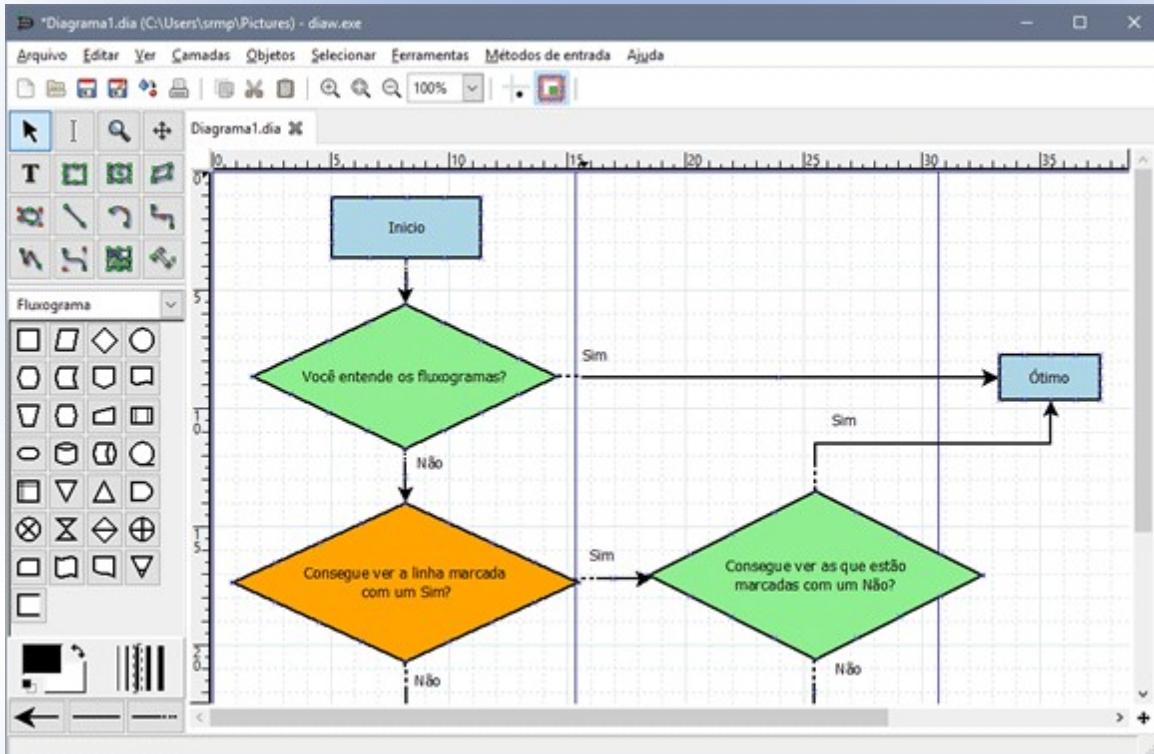




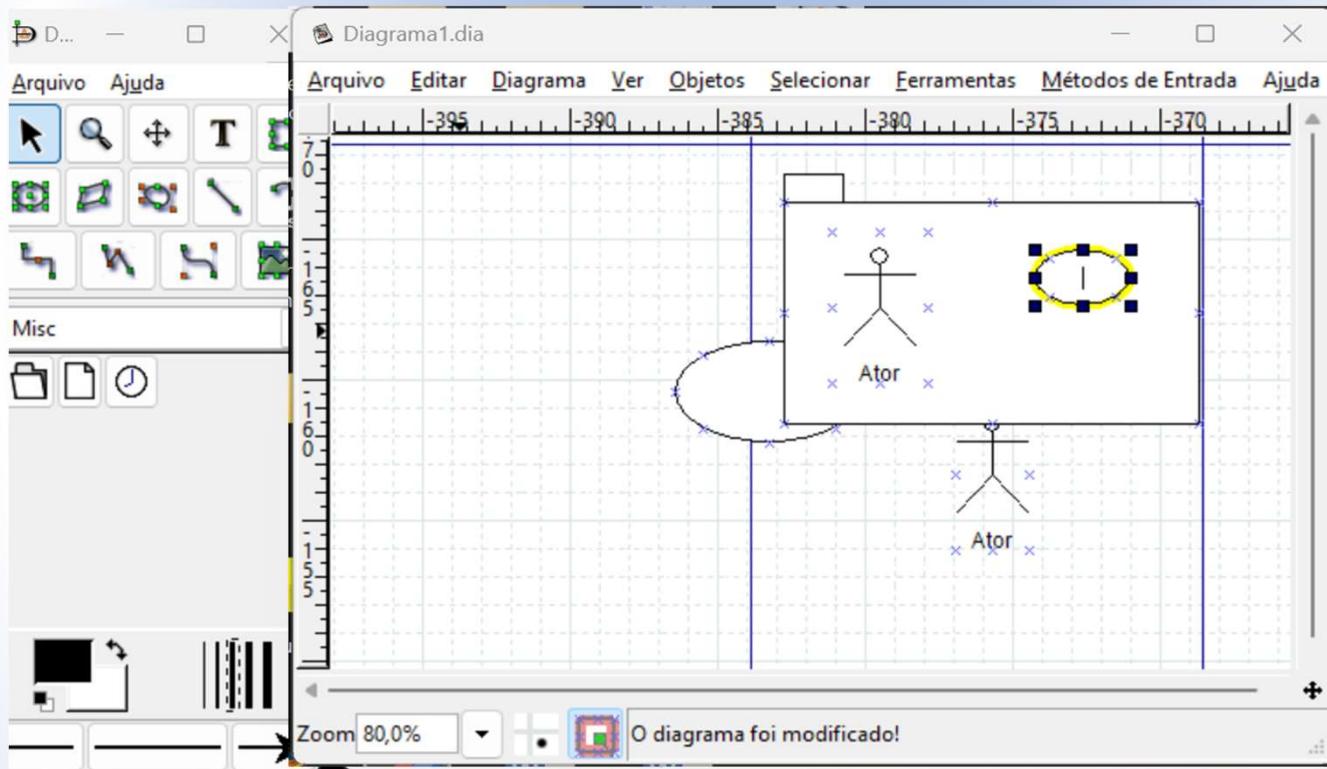
Ferramentas iniciais da Engenharia de Software:



Ferramentas iniciais da Engenharia de Software:

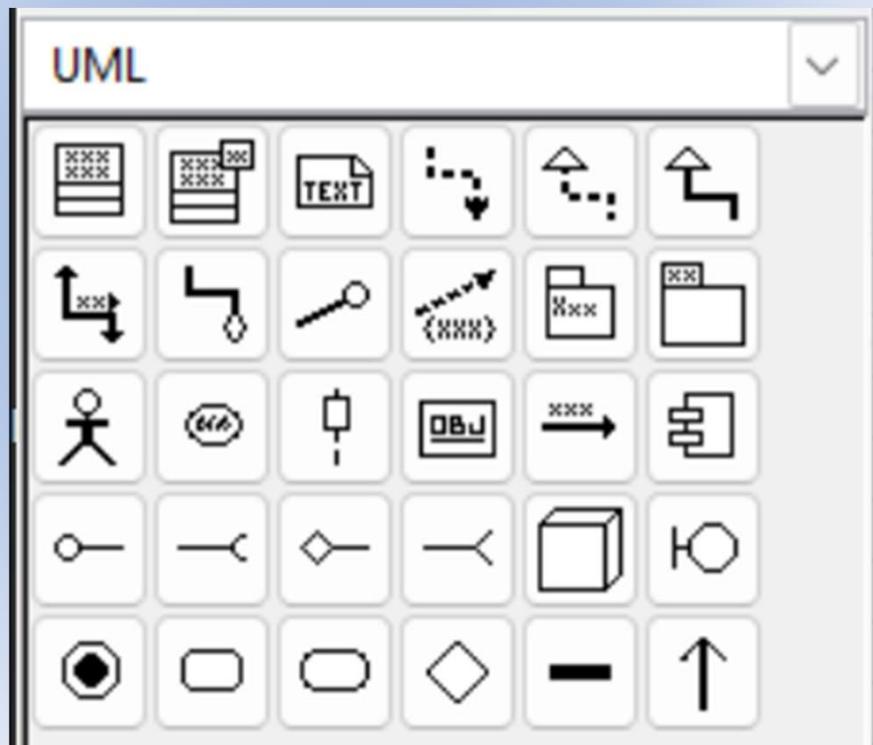
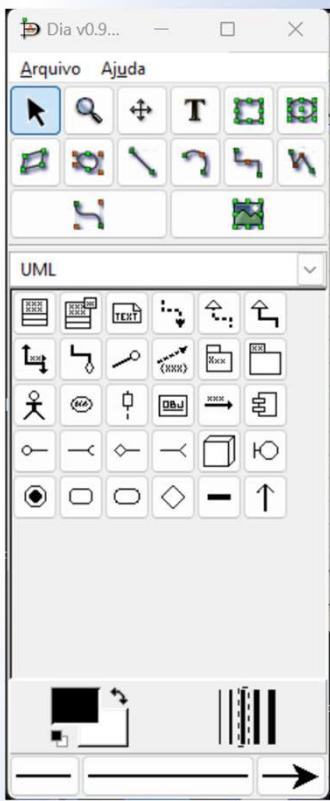


Ferramentas iniciais da Engenharia de Software:





Ferramentas iniciais da Engenharia de Software:



Ferramentas iniciais da Engenharia de Software:



diagrams.net

Security-first diagramming
for teams.

Bring your storage to our online tool, or go max privacy
with the desktop app.

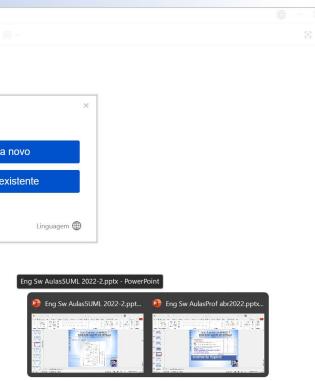
[Start](#) [Download](#)

No login or registration required.

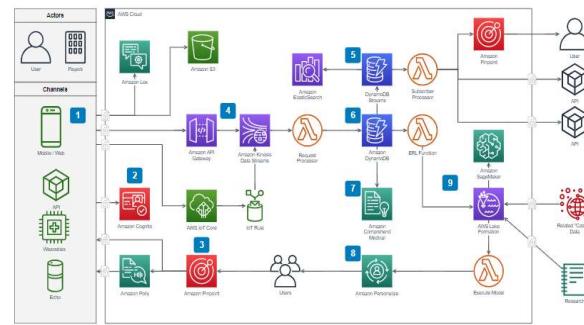


draw.io

Criar diagrama novo
Abrir diagrama existente



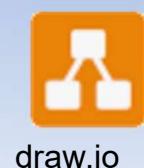
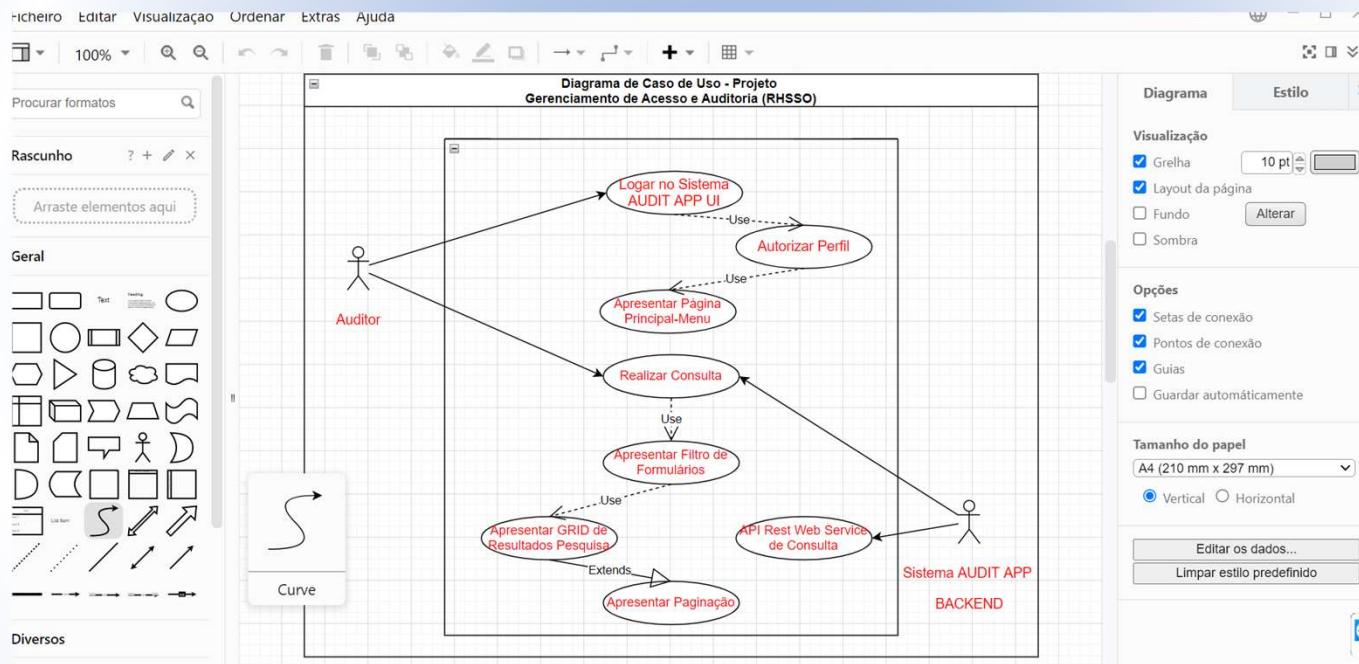
Eng Sw AulasSUMI_2022-2.pptx
Eng Sw AulasProf abr2022.pptx



A numbered diagram showing a flow from a User interacting with a Mobile App, through Amazon Cognito, to various AWS services including Lambda, S3, Kinesis, and Step Functions, eventually reaching an AI Model and a Researcher.

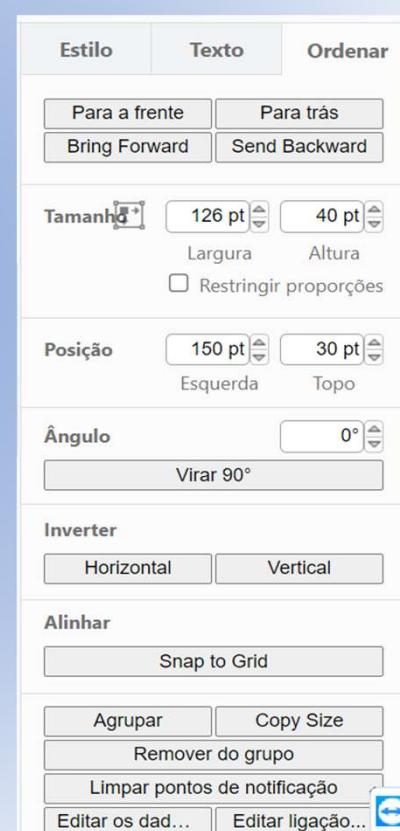
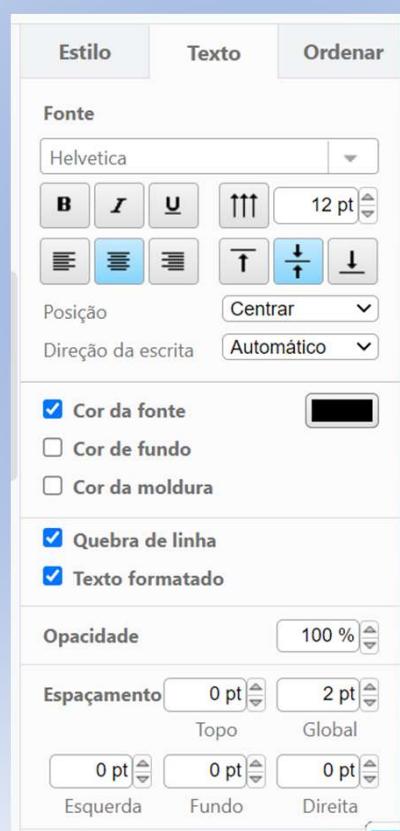
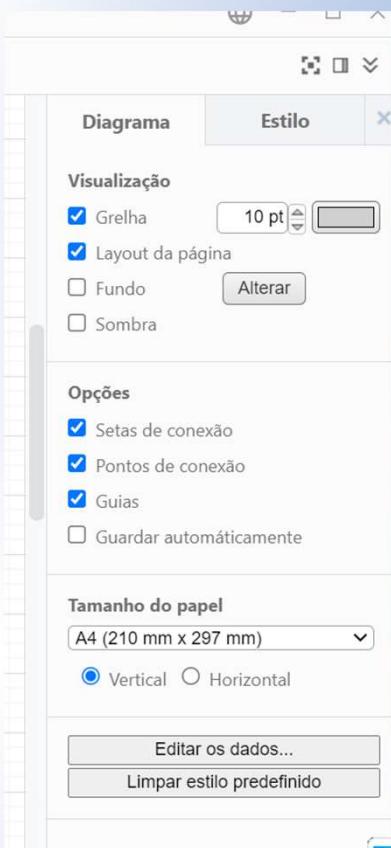


Ferramentas iniciais da Engenharia de Software:





Ferramentas iniciais da Engenharia de Software:



draw.io





***REQUISITOS!**

***VOCÊ SABERIA PEDIR, OUVIR, ENTENDER,
DESCREVER ?**



Como o cliente
explicou...



Como o líder de
projeto entendeu...



Como o analista
projetuou...



Como o programador
construiu...



Como o consultor de
negócios descreveu...



***REQUISITOS!**

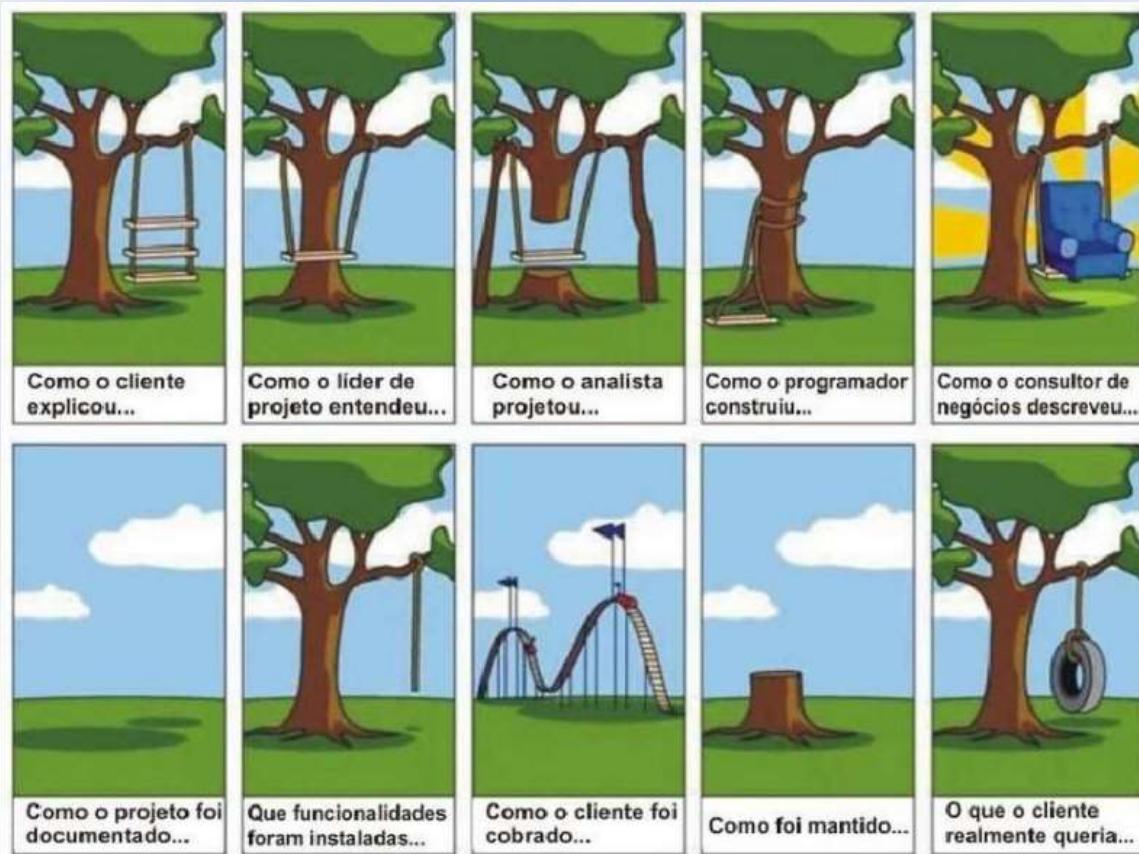
***VOCÊ SABERIA PEDIR, OUVIR, ENTENDER,
DESCREVER ?**





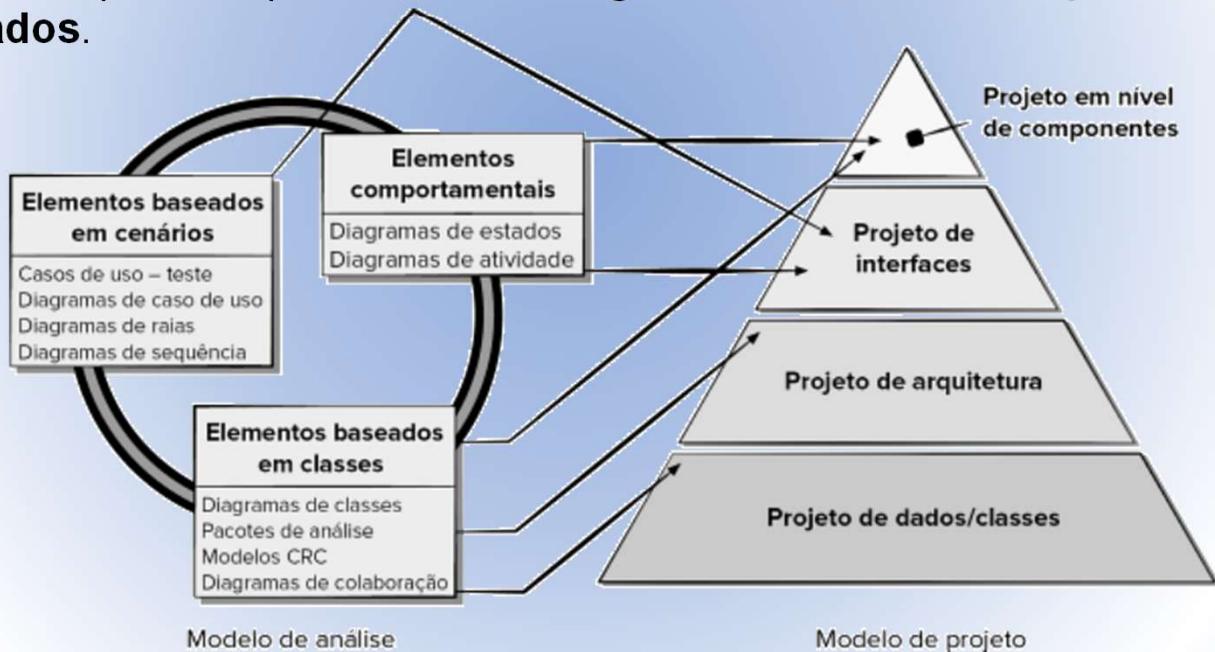
* REQUISITOS!

* VOCÊ SABERIA PEDIR, OUVIR, ENTENDER, DESCREVER ?



Projeto de Componentes, Arquitetura, Interfaces

Acertar o projeto de arquitetura antes de começar o desenvolvimento em si também é importante para **evitar cronogramas atrasados e orçamentos estourados**.



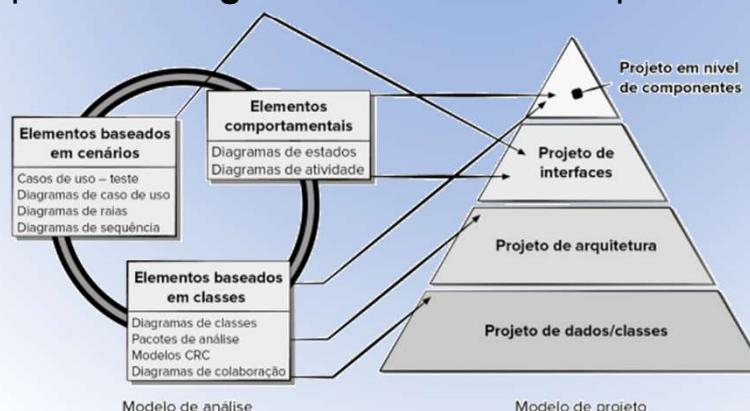
PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



Projeto

Acertar o projeto de arquitetura antes de começar o desenvolvimento em si também é importante para **evitar cronogramas atrasados e orçamentos estourados**.

A importância do projeto de software pode ser definida em uma única palavra – qualidade. **Projeto é a etapa em que a qualidade é incorporada à engenharia de software**. O projeto nos **fornecerá representações** do software que podem ser avaliadas em termos de qualidade. Projeto é a **única maneira** pela qual podemos **transformar** precisamente os **requisitos** dos envolvidos em um **produto** ou sistema de **software finalizado**. O projeto de software **serves como base** para todas as atividades de apoio e da **engenharia de software** que se seguem.

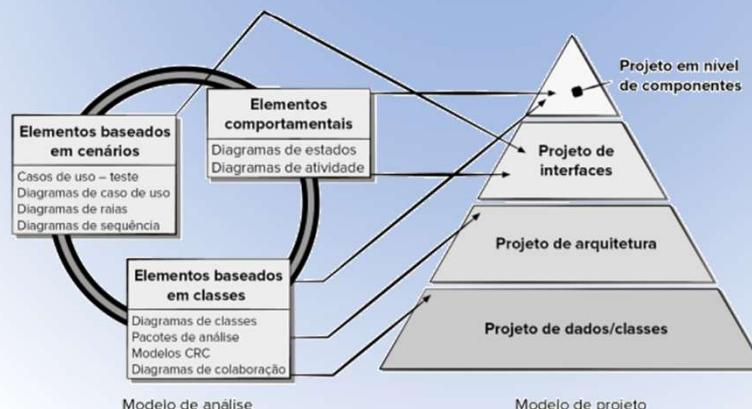


PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



Projeto de Interface

O projeto de **interfaces** descreve como o software se **comunica** com sistemas que **operam em conjunto** e com as **pessoas** que o utilizam. Uma interface **implica fluxo de informações** (p. ex., dados e/ou controle) e um tipo de **comportamento específico**. Consequentemente, **modelos comportamentais** e de **cenários de uso** **fornecem** grande parte das **informações necessárias** para o projeto de interfaces.

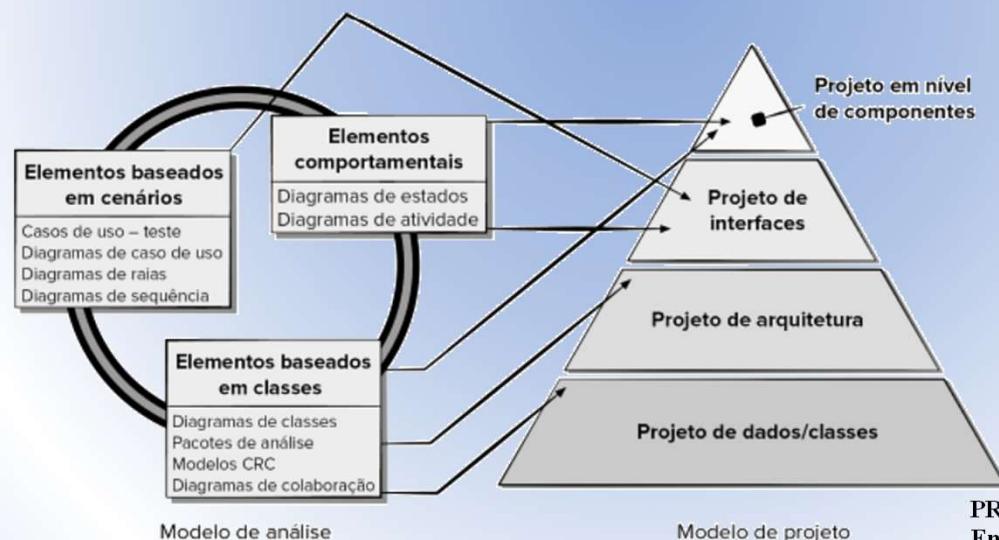


PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



Projeto de Componentes

O projeto de **componentes** transforma elementos estruturais da **arquitetura** de software em **uma descrição procedural** dos **componentes** de software. As informações obtidas dos modelos **baseados em classes** e dos **modelos comportamentais** servem como base para o projeto de componentes.



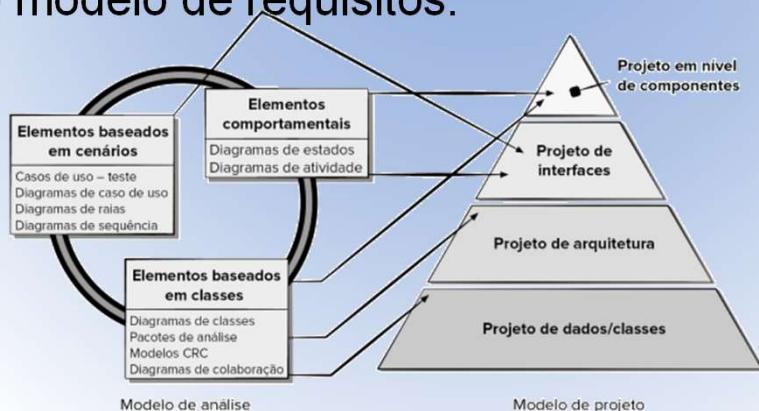
PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.





Projeto de Arquitetura

O projeto de **arquitetura** define as **relações entre** os principais **elementos estruturais** do software, o **estilo arquitetural** e os **padrões** de projeto que podem ser usados para **satisfazer** os **requisitos** definidos para o sistema e as **restrições** que afetam o modo como a arquitetura pode ser implementada [Sha15]. A **representação** do projeto de arquitetura – a **organização da solução técnica** de um sistema baseado em computador – é derivada do modelo de requisitos.

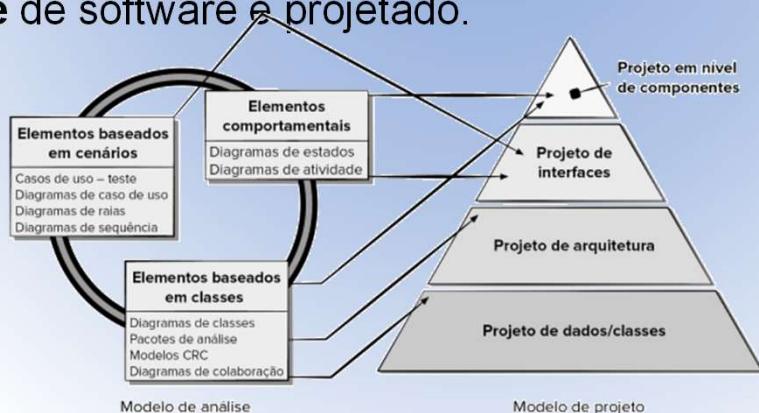


PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.



Projeto de Dados/Classes

O projeto de **dados/classes** transforma os **modelos** de classes em **realizações** de classes de projeto e nas estruturas de dados dos requisitos necessárias para implementar o software. Os objetos e as relações definidos no **modelo CRC** (classe-responsabilidade-colaborador) e no conteúdo detalhado dos dados representados por atributos de classes e outra notação fornecem a base para a realização do projeto de dados. **Parte** do projeto de classes **pode ocorrer com** o projeto da **arquitetura de software**. O projeto de classe mais detalhado **ocorre** à medida que cada **componente** de software é projetado.



PRESSMAN, Roger; MAXIM, Bruce.
Engenharia de Software. 8 ed. São Paulo:
McGraw Hill Brasil, 2016.

Projeto de Dados/Classes

Obs: quarta seção na parte inferior da caixa de classe pode ser usada para listar (**CRC**) as responsabilidades da classe.

Colaborações. responsabilidades de **duas formas**: (1) uma classe pode usar suas **próprias operações** para manipular seus próprios atributos, cumprindo, portanto, determinada responsabilidade; ou (2) uma classe pode **colaborar** com **outras classes**. As colaborações são identificadas determinando se uma classe pode ou não cumprir cada responsabilidade por si só. Caso não possa, ela precisa interagir com outra classe. PRESSMAN, Roger; MAXIM, Bruce. Engenharia de Software - 8 ed. São Paulo:

Classes. Deve seguir diretrizes básicas para a identificação de classes e objetos.

Responsabilidades.

Diretrizes básicas para a identificação das responsabilidades (atributos e operações)



Modelos de Processos de Desenvolvimento de Software (Modelo em Cascata, Espiral e Prototipagem). Classificação de Requisitos de Software (funcionais e não funcionais). Técnicas de Levantamento de Requisitos. Projeto de arquitetura. Projeto e Implementação. **Reúso de Software**. Engenharia baseada em componentes. Engenharia de Software distribuído. Arquitetura orientada a serviço. Estudo de Viabilidade. Técnicas de documentação. Metodologias para desenvolvimento de sistemas.

BIBLIOGRAFIA

BIBLIOGRAFIA BÁSICA:

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

PRESSMAN, Roger; MAXIM, Bruce. Engenharia de Software. 8 ed. São Paulo: McGraw Hill Brasil, 2016.

SOMMERVILLE, Ian. Engenharia De Software. 10 ed. São Paulo: Pearson Brasil, 2019.

BIBLIOGRAFIA COMPLEMENTAR:

LARMAN, Craig. Utilizando UML e padrões. 3 ed. Porto Alegre: Bookman, 2007.

REZENDE, Denis Alcides. Engenharia de software e sistemas de informação. 3 ed. Rio de Janeiro: Brasport, 2005.

WASLAWICK Raul. Análise e Projeto de Sistemas de Informação Orientados a Objetos. 2 ed. Rio de Janeiro: Elsevier, 2010.



APRESENTAÇÃO DO PROFESSOR

Luiz Lima
Professor



 aula.prof.luiz@gmail.com

APDADOS
Associação dos Profissionais de Dados da Província de Deus

[INÍCIO](#) [QUEM SOMOS](#) [NOTÍCIAS](#) [VIOLÂNCIA](#) [EVENTOS](#) [PARECERES](#) [SERVIÇOS](#) [CONTATO](#)

MEMBROS POR ESTADO

Pesquisa por nome do membro... Filtro de Cidade... Filtro Avançado...
Escolha o estado brasileiro, em seguida a cidade, e digite o acima o nome do profissional para validação.

Voltar para SP APDADOS Membros de São Paulo

Dado Alves Presidente CNPPD 2024 V CONGRESSO NACIONAL DOS PROFISSIONAIS DE PRIVACIDADE DE DADOS
Luiz Lima Diretor

- Técnico em **Eletrônica** (1989)
- Graduação: Tecnologia em **Processamento de Dados** (2001).
- Especialização: Formação em **Educação à Distância** (2011).
- Mestrado: **Engenharia de Produção** - Sistema Especialista AITOD baseado na Lógica Paraconsistente Anotada Evidencial ET (2018).
- Doutorado: **Engenharia de Produção** - Automação Computerizada no Diagnósticos Precoce de Câncer de Pênis feito por meio da Rede Neural Artificial Paraconsistentes (2022).

Obrigado!



IA/LP/Metaverso/LGPD.