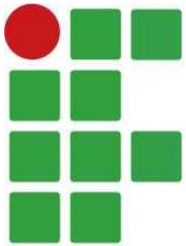
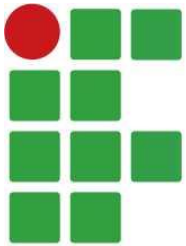


INSTITUTO FEDERAL
Rio Grande do Sul
Campus Feliz

UML -
Unified Modeling Language

Profa. Dra. Ana Paula Lemke



INSTITUTO FEDERAL
Rio Grande do Sul
Campus Feliz

Modelos

Modelos

- Descrevem e abstraem aspectos essenciais do sistema.
- Modelos visam:
  - Testar uma entidade física antes de lhe dar forma
    - Ex.: modelos de aviões testados em túneis de vento.
  - Comunicação com clientes (Ex.: plantas baixas)
  - Visualização (Ex.: maquetes)
  - Redução da complexidade pela decomposição do sistema (do mundo real ou do software) em pedaços compreensíveis.

Por que modelar o sistema?

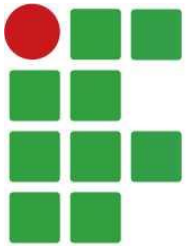
- Aceitação pelo usuário:
  - Ausência de um modelo visível ao usuário faz com que ele dê conformidade a soluções incompletas ou mesmo erradas;
  - Facilita a interação com o usuário.

- Ciclo de vida muito comprido:
  - O usuário modifica suas necessidades em função da dinâmica do mundo real.
  - As pessoas envolvidas podem não permanecer até o fim do projeto.

Por que modelar o sistema?

- Documentação:
  - Documentos textuais e narrativos cansam e desestimulam;
  - Indispensável para a manutenção do sistema.

- Confiabilidade pelo rigorismo e consistência entre as visões do sistema.

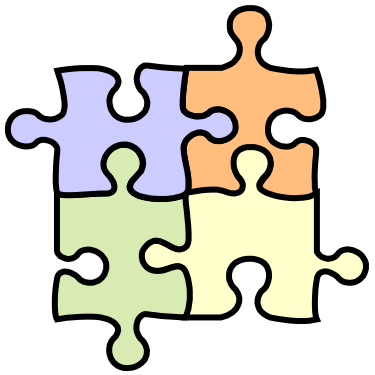


INSTITUTO FEDERAL
Rio Grande do Sul
Campus Feliz

Visão Geral da UML

Contexto

- Década de 90:
  - Mercado fragmentado.
  - Falta de padronização.
  - Resistência da indústria e usuários em investir em OO.



Histórico

- Metodologias anteriores à UML:
  - Booch
  - OMT
  - OOSE/Objectory
  - Outras metodologias...
- Cada metodologia possuía:
  - Notação própria (símbolos)
  - Processo
  - Ferramentas (CASE)
- A UML é a união do que há de melhor nas três metodologias.
  - A UML foi aprovada pela OMG em 1997.

Visão Geral

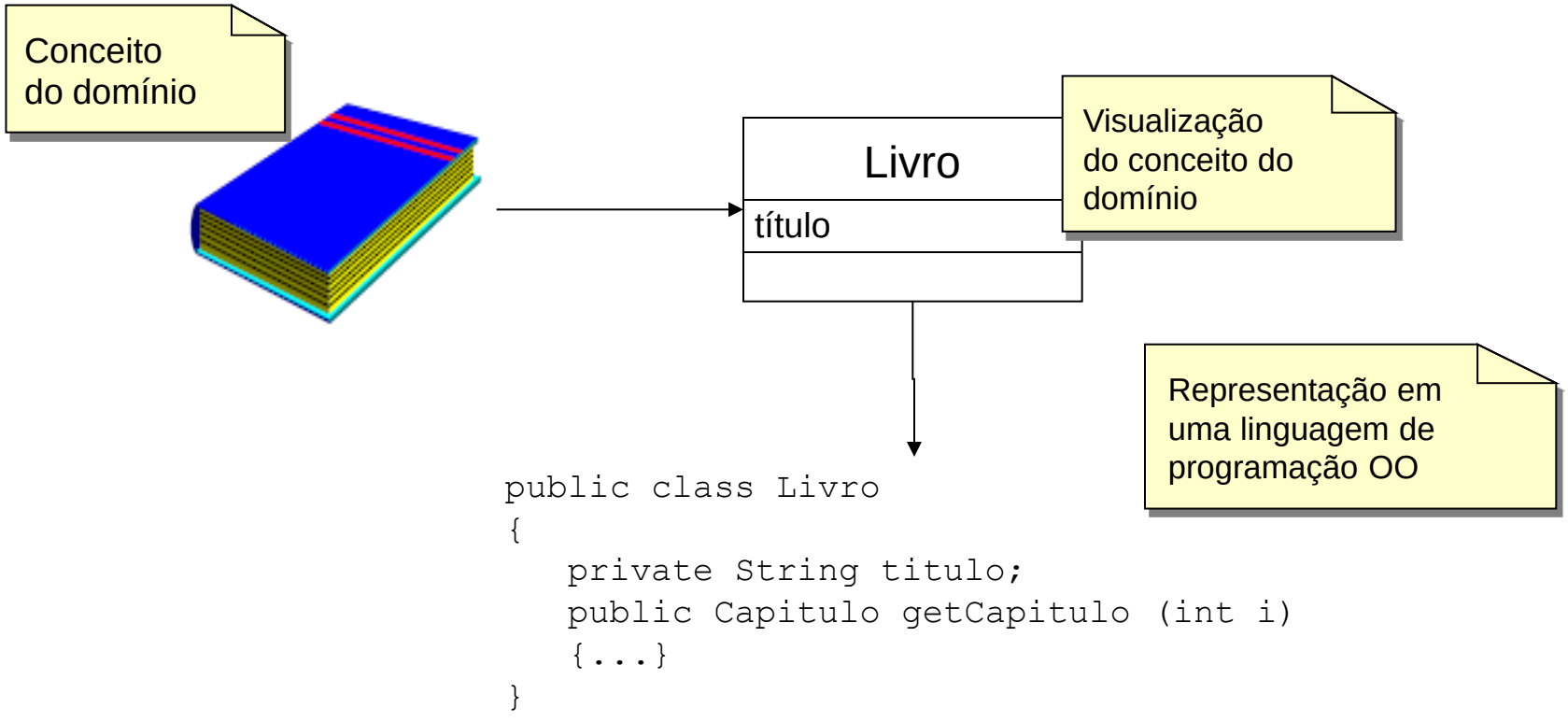
- Tentativa de padronizar a modelagem OO.
- Visa a criação de modelos com qualidade.
- A **UML** é uma linguagem destinada a **visualizar**, **especificar**, **construir** e **documentar** os artefatos de um sistema complexo de software.

- **Artefatos**: diagramas, documentos e código-fonte.

Objetivos da UML

- Descrever modelos de sistema - do mundo real e de software - baseados em conceitos de objetos.
  - Fornecer uma linguagem de modelagem OO visual fácil, pronta para uso, permitindo amplas facilidades de modelagem.
  - Fornecer mecanismos de extensibilidade e especialização de conceitos de base.
  - Independência de processos e linguagens de programação.
    - Abranger todo o ciclo de vida.
    - Diferentes tecnologias de implementação.
  - Integrar melhores práticas em desenvolvimento OO de sistemas.

Objetivos da UML



Adaptado do livro “Utilizando UML e padrões", de Craig Larman, 2001

Vantagens da UML

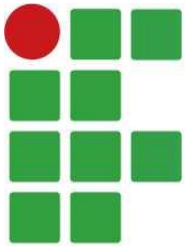
- Padronização:
  - Impulso no desenvolvimento e adoção de ferramentas para desenvolvimento OO de software
  - Intercambialidade, interoperabilidade

- Bom compromisso entre conceituação e flexibilidade:
  - Ampla variedade notacional
  - Facilidade de extensão/personalização
  - Facilidade de evolução (conceitual, tecnológica)

Principais Verdades sobre UML

- **UML não é proprietário**.
  - A UML foi originalmente concebida pela *Rational Software*, mas agora a UML é propriedade da OMG, e é aberta para todos.
- **UML não é um processo ou método.**
  - A UML encoraja o uso de técnicas OO e ciclos de vida iterativos.
- **UML não é difícil.**
  - A UML é grande, mas você não precisa usar ou entendê-la toda.
  - Ivar Jacobson disse que “20% da UML resolve cerca de 80% dos problemas do dia-a-dia”.
- **UML não é perda de tempo.**
  - Se a UML for corretamente usada, ela pode ajudar a diminuir o tempo total de desenvolvimento, o custo total do projeto e os custos com comunicação. UML melhora a compreensão, a produtividade e a qualidade do sistema desenvolvido.

Adaptado do livro “UML 2.0 for Dummies”, de Michael Jesse Chonoles e James A. Schardt, 2003.

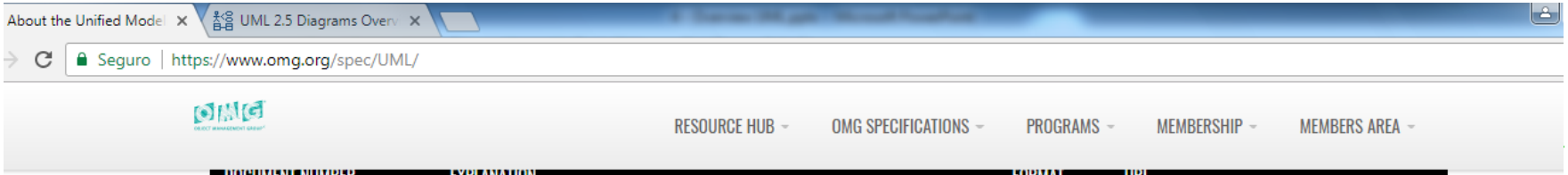


INSTITUTO FEDERAL
Rio Grande do Sul
Campus Feliz

Overview dos diagramas da UML 2.5.1

Versão dos livros utilizados na confecção do material: 2.3
Versão estável atual (dezembro de 2017): 2.5.1

Fonte de consulta oficial: http://www.omg.org/spec/UML/
Boa fonte de consulta: http://www.uml-diagrams.org/



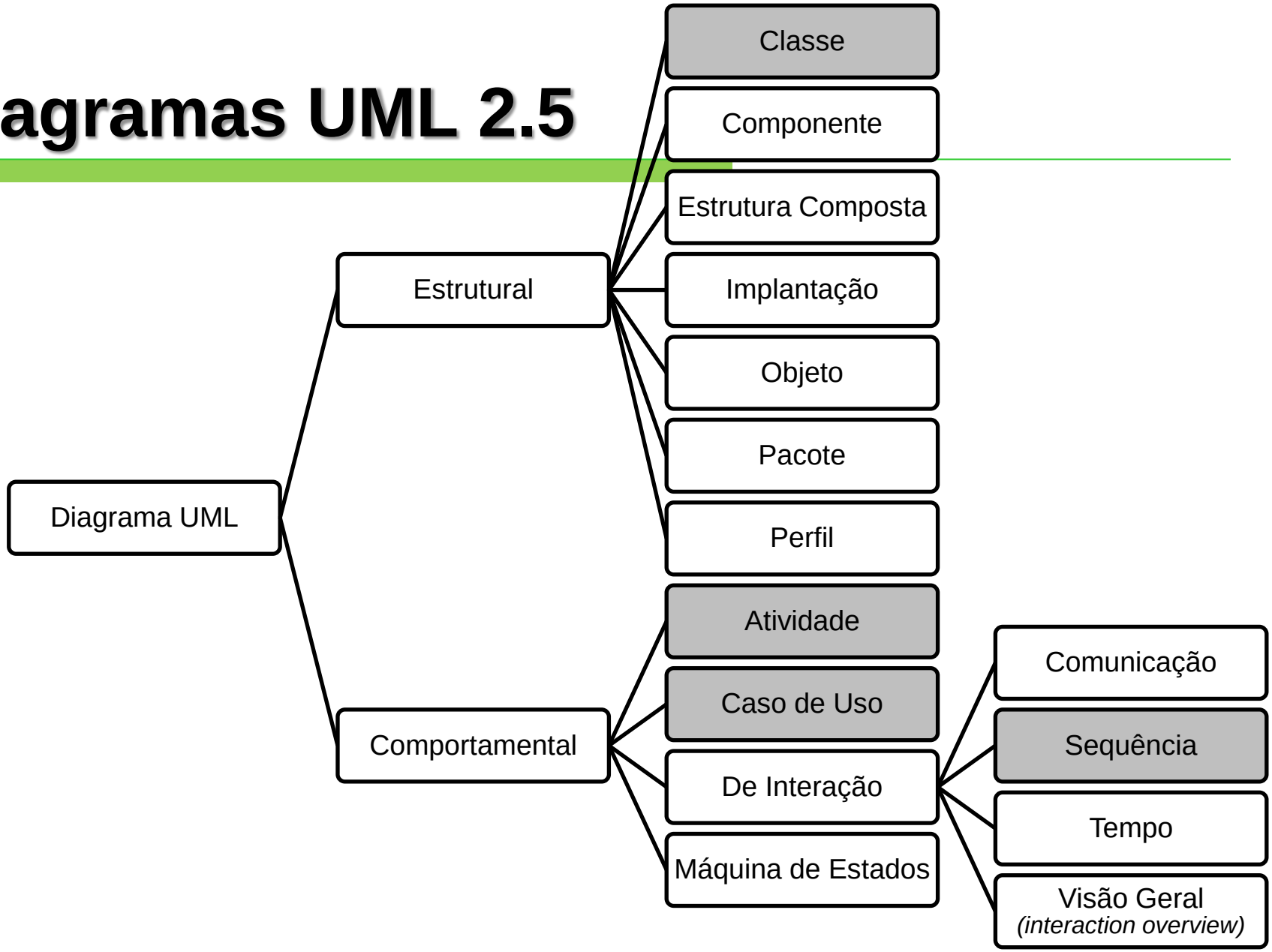
DOCUMENT NUMBER	EXPLANATION	FORMAT	URL
formal/17-12-06	Specification changebar	PDF	UML/2.5.1/PDF/changebar

HISTORY

FORMAL VERSIONS

VERSION	ADOPTION DATE	URL
2.5.1	December 2017	https://www.omg.org/spec/UML/2.5.1
2.4.1	July 2011	https://www.omg.org/spec/UML/2.4.1
2.3	May 2010	https://www.omg.org/spec/UML/2.3
2.2	January 2009	https://www.omg.org/spec/UML/2.2
2.1.2	October 2007	https://www.omg.org/spec/UML/2.1.2
2.0	July 2005	https://www.omg.org/spec/UML/2.0
1.5	March 2003	https://www.omg.org/spec/UML/1.5
1.4	September 2001	https://www.omg.org/spec/UML/1.4
1.3	February 2000	https://www.omg.org/spec/UML/1.3
1.2	July 1999	https://www.omg.org/spec/UML/1.2
1.1	December 1997	https://www.omg.org/spec/UML/1.1

Diagramas UML 2.5



Síntese dos diagramas UML 2.X

- Estruturais:
  - Mostram as características do sistema que não mudam como tempo.
  - **Diagramas**: Classe, Objeto, Pacote, Estrutura composta, Componentes, Implantação e Perfil.
- Comportamentais:
  - Mostram como o sistema responde as requisições ou vai se modificando ao longo do tempo.
  - **Diagramas**: Caso de Uso, Atividade e Máquina de Estados..
- De Interação:
  - São também diagramas comportamentais. Mostram a troca de mensagens em uma colaboração (um grupo de objetos cooperantes), para atingir um objetivo.
  - **Diagramas**: Sequência, Comunicação, Visão Geral e Tempo.

Diagrama de Casos de Uso

- Mostra um conjunto de atores e casos de uso (funcionalidades do sistema).
- Atores ajudam a identificar os tipos de usuários do sistema e podem, inclusive, ser outros sistemas.

Diagrama de Atividades

- Descreve uma **sequência** de ações.
- Um diagrama de atividades associado a um Caso de Uso descreve as atividades realizadas pelo Ator e pelo Sistema.

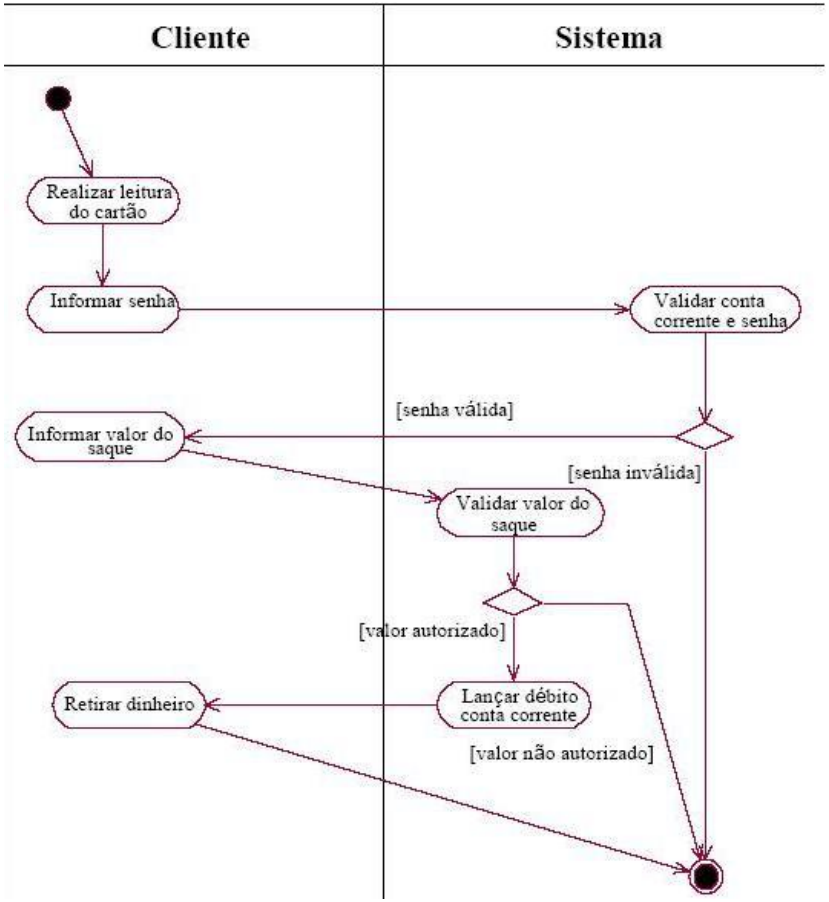


Diagrama de Classes

- O **Diagrama de Classes** é um esquema, um padrão ou um modelo que descreve muitas instâncias de objetos.
- Mostra a estrutura de classes, seus relacionamentos, atributos e métodos.

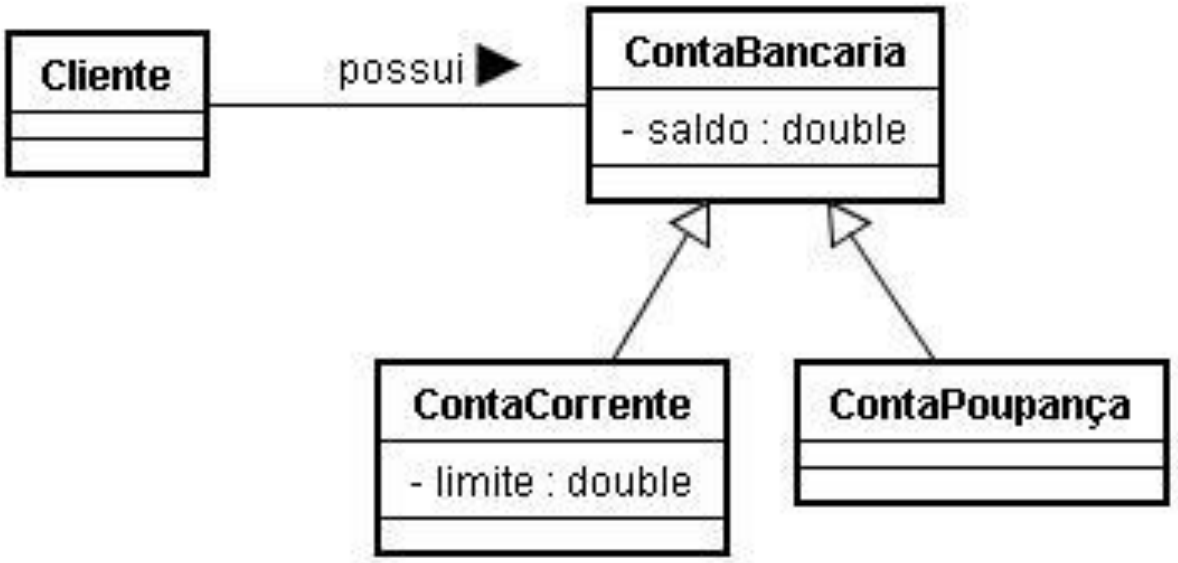


Diagrama de Objetos

- Mostra um conjunto de objetos e seus relacionamentos.

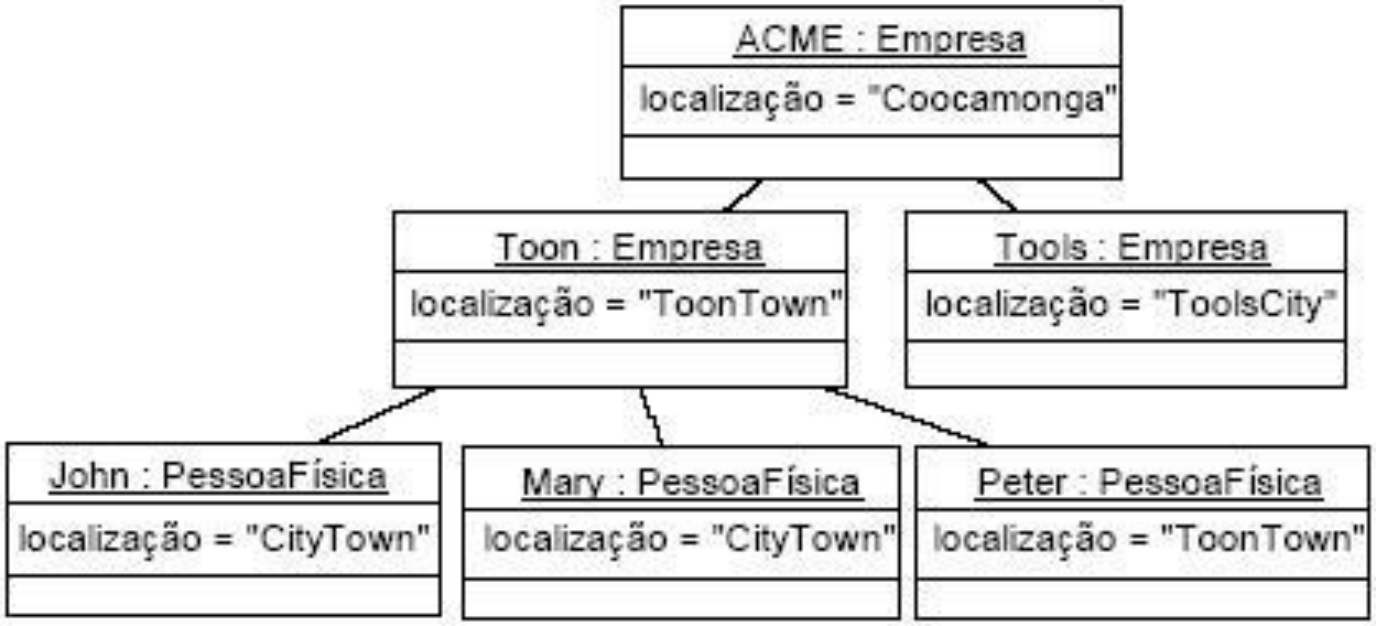


Diagrama de Estrutura Composta

- Foi criado para representar visualmente as partes de classes, componentes ou colaborações, incluindo os pontos de interação, usados para acessar as *features* do sistema.
  - Modela um conjunto de instâncias que cooperam entre si para executar uma função específica.

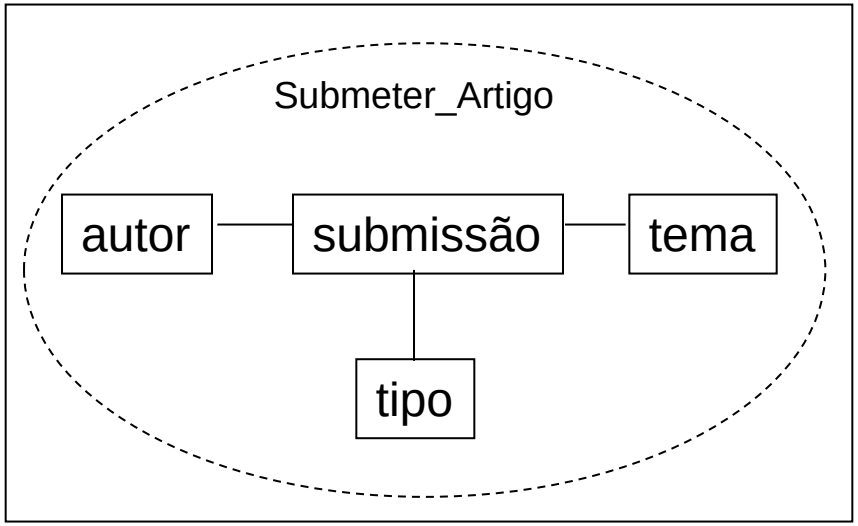


Diagrama de Sequência

- Diagrama de interação que enfatiza o ordenamento das mensagens trocadas entre os objetos.

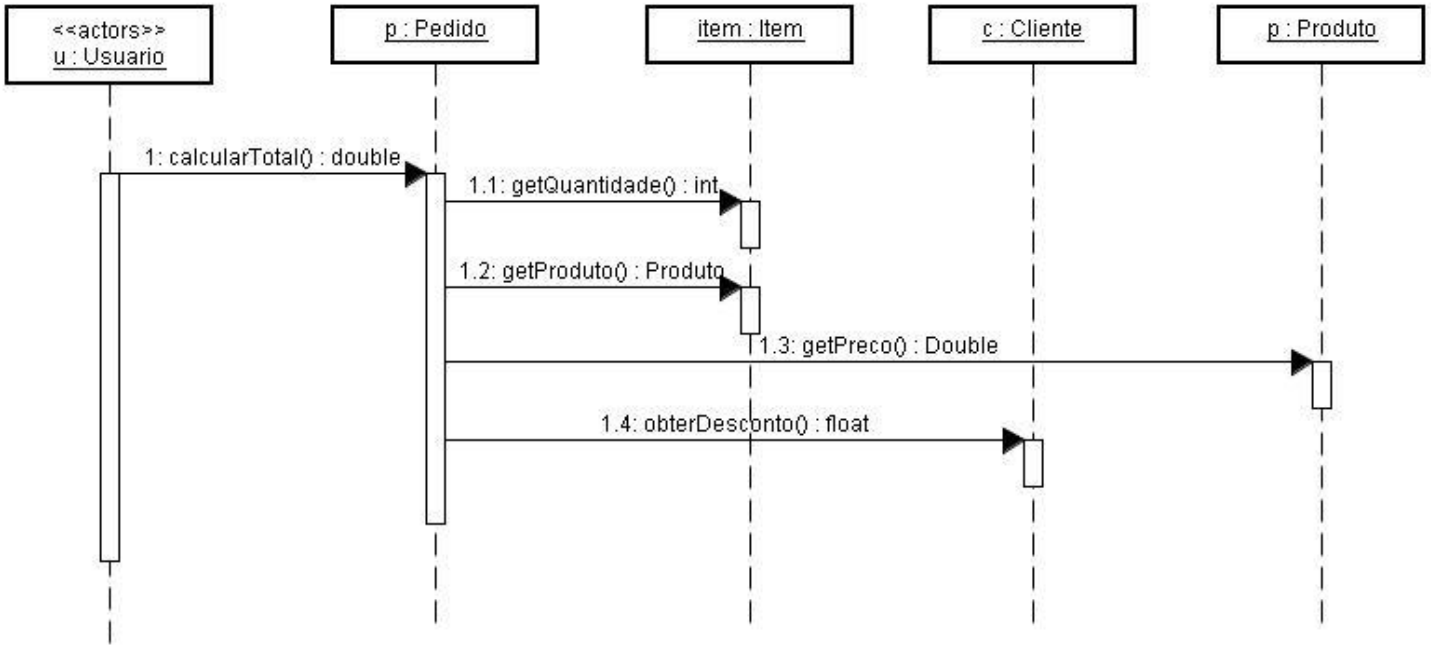


Diagrama de Comunicação

- Diagrama de interação que enfatiza a organização estrutural dos objetos que trocam mensagens.

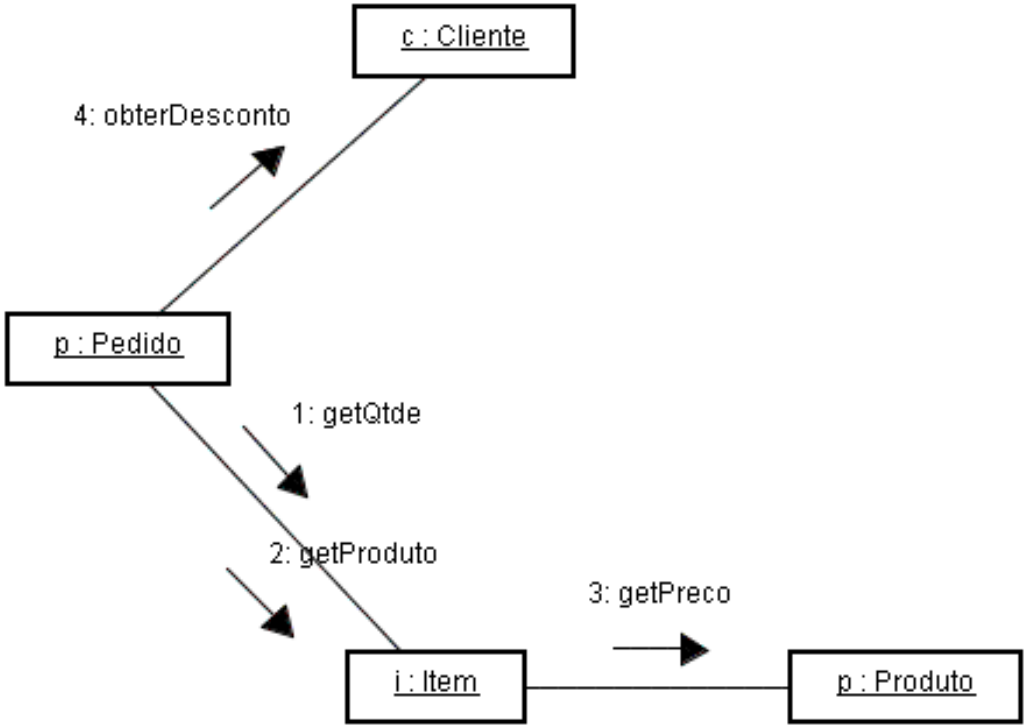
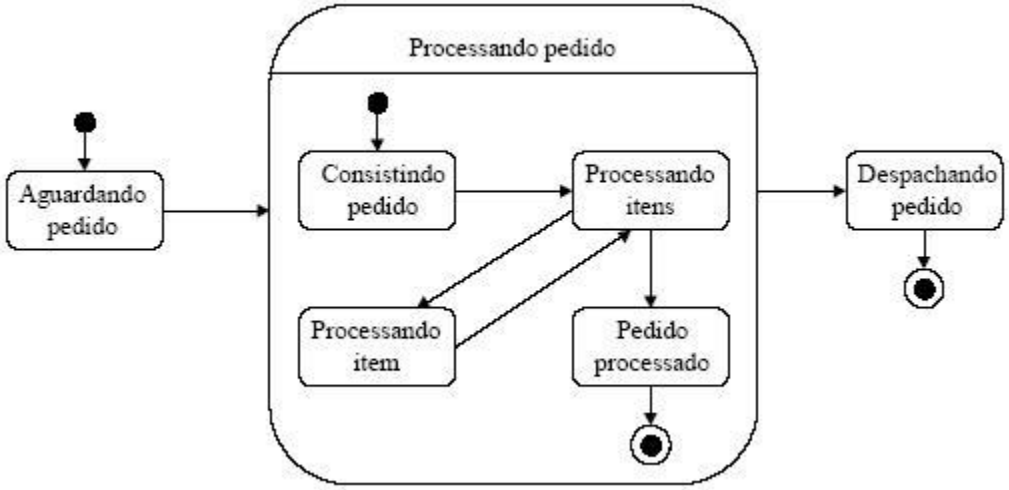


Diagrama de Máquina de Estados

- Mostra como estímulos externos causam mudanças no objeto ao longo de seu tempo de vida.
- Particularmente útil na construção de sistemas reativos.



- **Estado**: condição ou situação existente na vida de um objeto.

Diagrama Visão Geral

- Pode ser considerado como um diagrama de atividade onde as ações são substituídas por pequenos diagramas de seqüência, comunicação ou tempo.
- Foram definidos para visualizar o fluxo geral de controle, logo, eles não mostram em detalhes as mensagens.

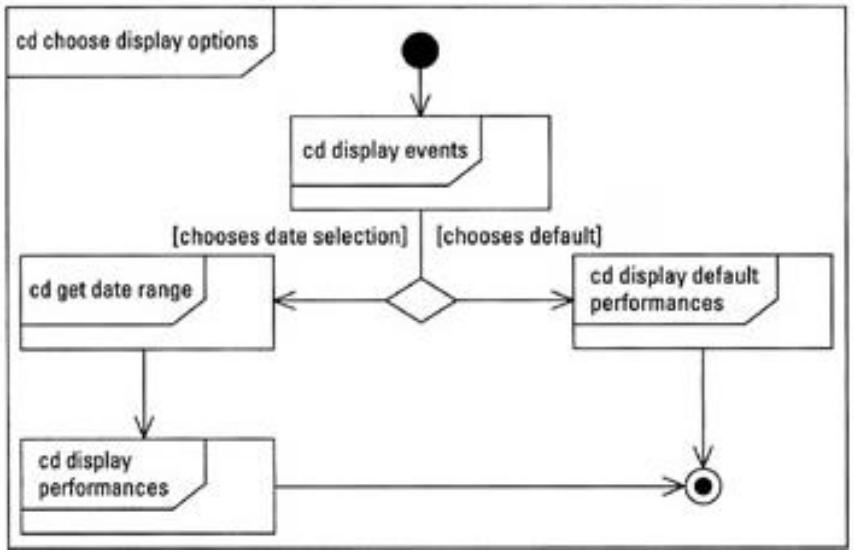


Diagrama de Pacotes

- Mostra subsistemas ou módulos englobados por um sistema de forma a determinar as partes que o compõe.
- Foca em como os elementos são agrupados e nas dependências decorrentes desses agrupamentos.

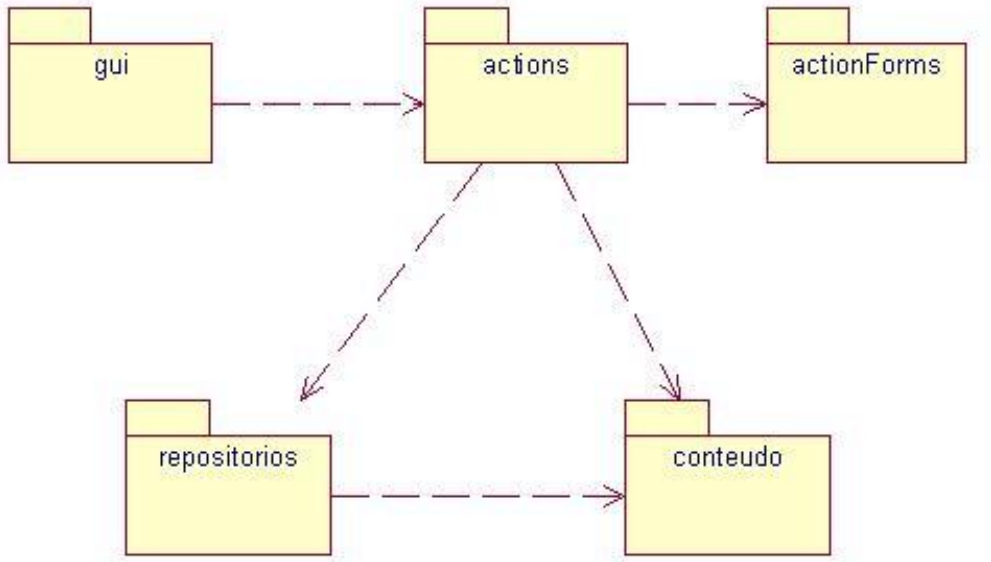


Diagrama de Componentes

- Diagramas de componentes mostram uma visão estática da implementação de um sistema.
- São compostos por **componentes**, **interfaces** e **relações entre componentes.**

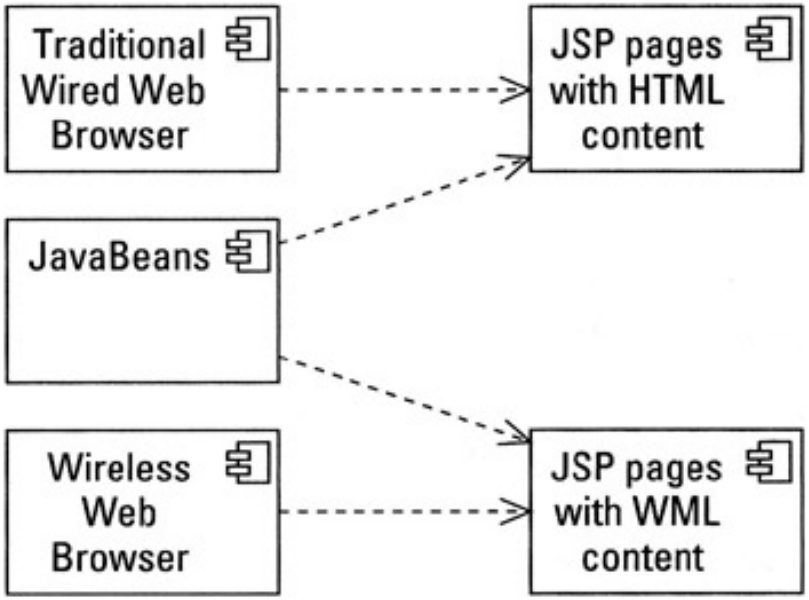


Diagrama de Implantação

- Um diagrama de implantação provê uma visão da relação física entre componentes de *software* e de *hardware*.
- Cada nodo de um diagrama de implantação tipicamente representa um tipo de hardware, com um PC, um servidor, um drive de CD.

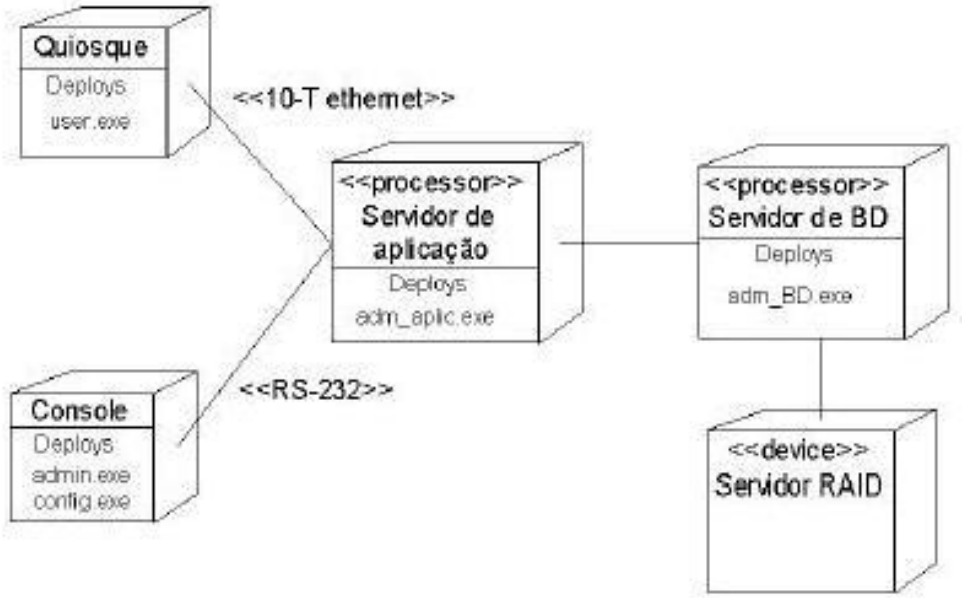


Diagrama de Tempo

- Descreve a mudança no estado ou condição de um objeto de uma classe durante um tempo.
- Foi projetado para especificar as restrições de tempo relacionadas ao envio e recebimento de mensagens durante uma interação.

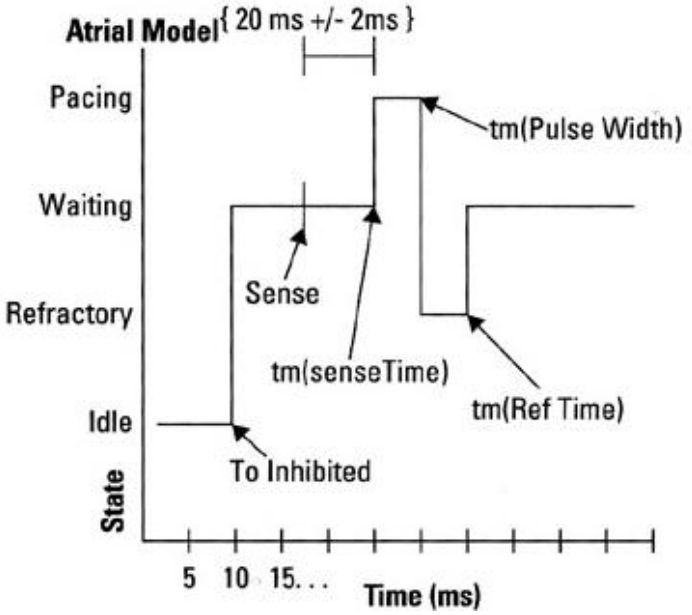
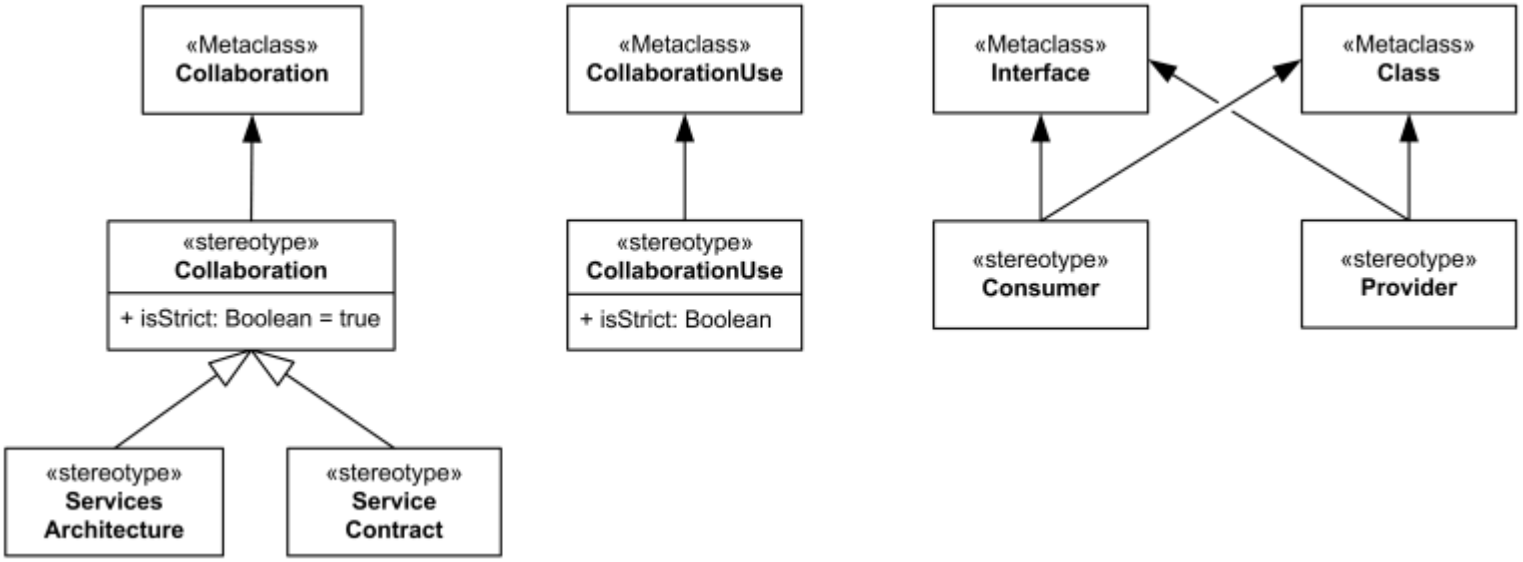


Diagrama de Perfil

- Descreve mecanismos de extensão para a UML através da customização de estereótipos, valores atribuídos e restrições.



Modelando com UML

- Principais perguntas em um sistema e como respondê-las usando diagramas UML:
  - **Quem usa o sistema?**
    - Mostre os **atores** em seus **diagramas de casos de uso**.
  - **O sistema é constituído de quê?**
    - Desenhe **diagramas de classes** para mostrar a estrutura lógica do sistema e **diagramas de componentes** para mostrar a estrutura física.
  - **Em que lugar estão situados os componentes do sistema?**
    - Indique seus planos de onde os componentes irão estar localizados e "rodar" nos **diagramas de implantação**.

Adaptado do livro "UML 2.0 for Dummies", de Michael Jesse Chonoles e James A. Schardt, 2003.

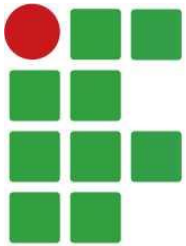
Modelando com UML [cont]

- **Quais e quando importante eventos ocorrem no sistema?**
  - Mostre quais os eventos que os objetos reagem com **diagramas de máquinas de estados e de tempo**.
- **Como o sistema começa a trabalhar?**
  - Mostre as partes do sistema em **diagramas de estruturas compostas** e use **diagramas de comunicação** para mostrar as interações no nível de abstração necessário para projeto e implementação.

Adaptado do livro “UML 2.0 for Dummies”, de Michael Jesse Chonoles e James A. Schardt, 2003.

Ferramentas de modelagem UML

- Modelar sistemas utilizando a notação UML é mais fácil quando se usa uma ferramenta de modelagem UML, que serve para:
  - Desenhar diagramas UML
  - Desenhar a notação UML corretamente
  - Organizar os diagramas em pacotes
  - Procurar por elementos específicos nos diagramas
  - Fazer engenharia reversa
  - Fazer relatórios de modelos
  - Gerar código



INSTITUTO FEDERAL
Rio Grande do Sul
Campus Feliz

UML

.:. Termos e conceitos .:.

Definições Básicas

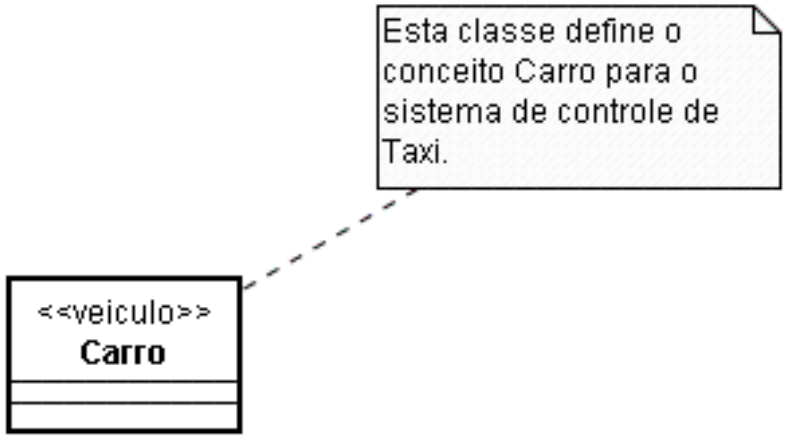
- **Sistema**: coleção de subsistemas organizados para a realização de um objetivo e descritos por um conjunto de modelos.
- **Subsistema**: representa uma partição dos elementos de um sistema maior em partes independentes.
- **Modelo**: são abstrações semanticamente fechadas de um sistema, representando uma simplificação autoconsistente e completa da realidade.
- **Visão**: abrange um subconjunto de itens que pertencem a um modelo, cujo foco esta voltado para um único aspecto do sistema.
- **Diagramas**: apresentação gráfica de um conjunto de elementos. Cada diagrama tem um diferente propósito e expressa um importante aspecto ou mecanismo do sistema.

Adornos

- Elementos gráficos ou textuais que são adicionados à notação básica de elementos UML.
- Usados para visualizar detalhes a respeito da especificação de um elemento.
- São adicionadas através da colocação de texto ou símbolo gráfico perto do elemento a ser detalhado.
- Exemplos:
  - Papéis e cardinalidades de uma associação;
  - Notas;
  - Etc.

Nota

- Símbolo gráfico para representação de restrições ou de comentários anexados a um elemento ou a uma coleção de elementos.
- Usos: requisitos, observações, explicações, comentários, código para operações, entre outros.

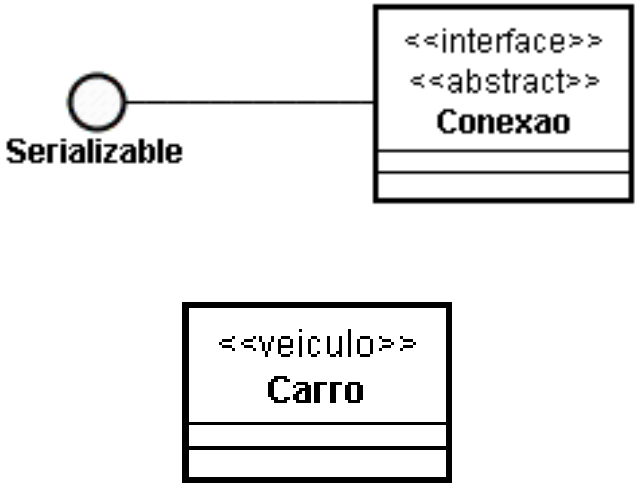


Mecanismo de Extensibilidade

- Usado para estender a linguagem UML de “forma controlada”.
- Permite adaptação fácil a novos contextos (e.g. modelagem de aplicações de um determinado domínio), reaproveitando um núcleo de primitivas e construtores consolidados:
  - Modelagem na WEB
  - *Workflow*
  - Sistemas geográficos

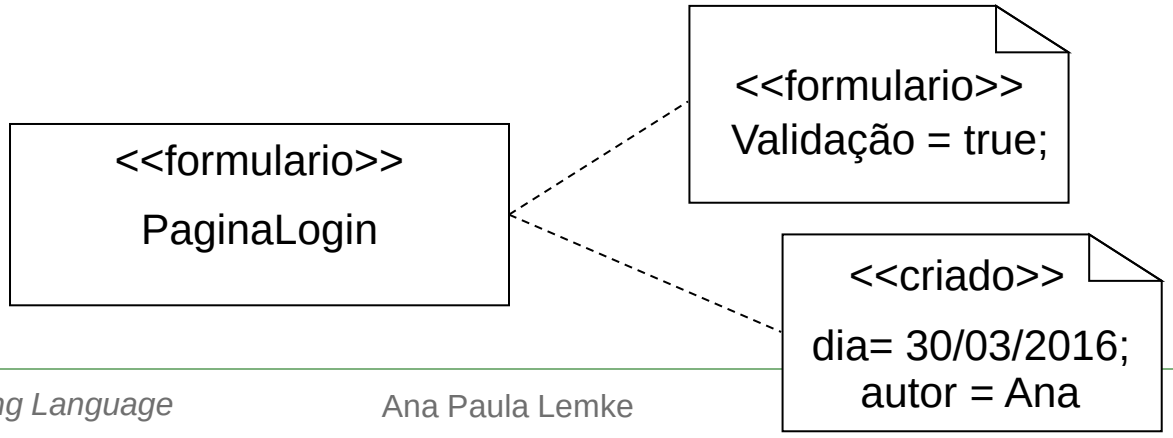
Extensibilidade - Estereótipos

- **Estereótipo**: é uma extensão do vocabulário de UML, permitindo a criação de novos tipos de blocos de construção semelhante aos existentes, mas específicos a determinado problema.
- Estereótipos estendem a semântica, mas não a estrutura de tipos e classes pré-existentes.
  - Um tipo particular de associação.
  - Um tipo particular de classe (atributos próprios implícitos, restrições sobre tipos de associações aos quais pode estar ligado, etc.).



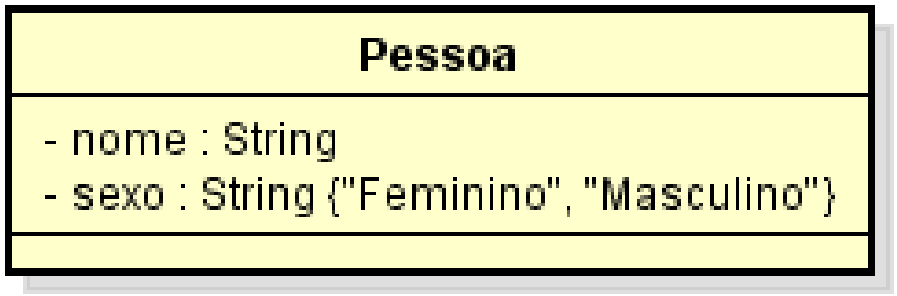
Extensibilidade – Valores atribuídos

- **Valor Atribuído**: é uma extensão da propriedade de um elemento da UML, permitindo a criação de novas informações na especificação deste elemento.
- **Não confundir valor atribuído com atributo**!
  - Os atributos definidos no modelo de projeto existem no sistema. Os valores atribuídos existem apenas no modelo de projeto. Eles afetam a geração do código, mas normalmente não são vistos nele.
- Uso comum: especificação de propriedades relevantes para geração de código ou gerência de configuração.



Extensibilidade - Restrições

- **Restrição**: é uma extensão da semântica de um elemento da UML, permitindo adicionar novas regras ou modificar as existentes.
- Representação gráfica: **{<restrição>}**



Tarefas

- Lista de exercícios de fixação
- Trabalho de pesquisa
  - Descrição da notação de diferentes diagramas.

- Quer saber mais sobre UML?
  - Leia a parte introdutória do livro "**UML: Guia do Usuário"**

Bibliografia

- Sbrocoo, J.S. **UML 2.3**: Teoria e Prática. São Paulo: Érica, 2011.
- Lima, A. **UML 2.3**: do requisito à solução. São Paulo: Érica, 2011.
- Fowler, Martin. **UML Essencial**. Porto Alegre: Bookman Companhia, 2005, 3ª edição.
- Booch, G.; Rumbaugh, J., Jacobson, I. **UML - GUIA DO USUÁRIO**. Rio de Janeiro: Campus, 2005, 2ª edição.
- Miles, Russ, Hamilton, Kim. "**Learning UML 2.0"**. O'Reilly Media, 2006, 1ª edição.
- http://www.uml-diagrams.org/

Bibliografia complementar utilizada para confecção desse material:

- Larman, Craig. **Utilizando UML e padrões**. Porto Alegre: Bookman, 2007, 3ª edição.
- Chonoles, M. and Schardt J. "**UML 2.0 for Dummies**", 2003.