ENGENHARIA DE REQUISITOS

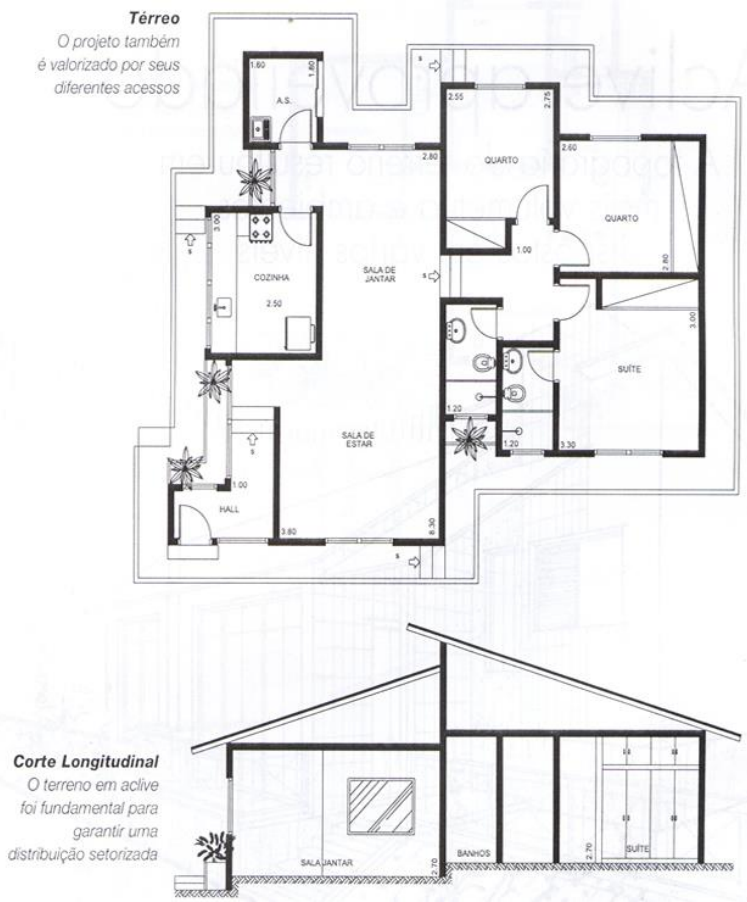
Unified Modeling Language – UML

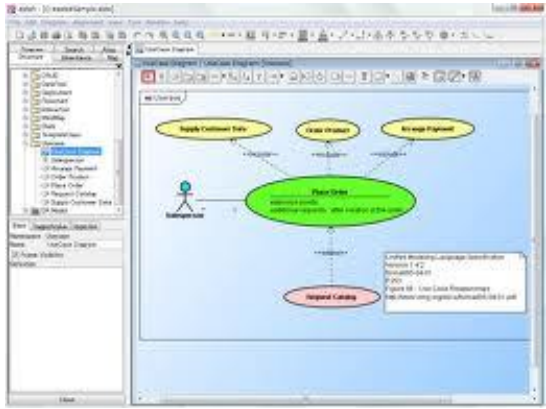
Introdução

Universidade de Passo Fundo – UPF
Prof. Jeangrei Veiga
jeangrei@upf.br

Introdução

- **Softwares**, assim como qualquer desenvolvimento de alta complexidade **precisam ser planejados** antes de serem





A modelagem no planejamento de um software

- O desenvolvimento de um **software sem o planejamento** adequado pode representar um **grande risco** para a **qualidade** do produto gerado, para o cumprimento de **prazos**, para o **custo** do desenvolvimento, em fim, para o **êxito** do desenvolvimento.

A modelagem no planejamento de um software

- A **modelagem** constitui um **auxílio ao tratamento da complexidade** do software que está sendo planejado. Proporciona um **caminho** entre o estabelecimento de um problema e o código de software que constitui a **solução**.
- A modelagem é parte central de todas as atividades que levam à implantação de um bom sistema.

O que é um modelo?

- **Modelos são abstrações** que retratam a essência de um problema.
  - Engenheiros, arquitetos, e demais profissionais montam modelos antes de criar algo. Em desenvolvimento de software, não é diferente. Podemos criar modelos que representam o software antes de implementá-lo.

O que é um modelo?

- A modelagem é uma maneira de pensar sobre problemas e soluções utilizando modelos simplificados que representem tanto os requisitos do que se está querendo desenvolver, como as especificações para a solução proposta e como deve ser implementada.

Objetivos da Modelagem

- Na modelagem de softwares orientados a objetos objetiva-se obter uma **abstração de todos os objetos que irão compor o sistema**, com seus dados e comportamentos.

Objetivos da Modelagem

- Também busca-se ter uma boa **especificação do software**, tornando-o mais fácil de implementar e modificar, passível de ter partes reutilizadas. Dessa forma, a modelagem deve representar o sistema de uma maneira eficaz e compreensível.

- A modelagem também representa um importante recurso para a **documentação** no desenvolvimento de um software.

Modelagem na Análise e Projeto

- Para o planejamento de um software é necessário especial atenção em duas etapas do processo de desenvolvimento: **Análise e Projeto.**
- Estas etapas são voltadas a entender e planejar o que será implementado.

Modelagem na Análise e Projeto

- Na etapa de análise a modelagem representa um esforço para o domínio do problema.

- Na etapa de projeto, a modelagem é focada em representar a solução computacional.

Requisitos para uma Modelagem Completa

- Uma modelagem completa deve ser útil principalmente em duas situações:
- **Codificação:** deve proporcionar a um programador gerar código correto a partir da especificação de projeto, mesmo não tendo participado da análise e projeto do sistema. Deve ter todas as informações necessárias para orientar a codificação, de forma que fique coerente com a especificação de requisitos.

Requisitos para uma Modelagem Completa

- **Manutenção:** Deve permitir a quem que for fazer uma manutenção no sistema, entender o que foi feito, mesmo não tendo participado da etapa de implementação, sem que este precise analisar os códigos para obter as informações necessárias para a manutenção.

Modelagem Estrutural

- Representa a **estrutura** de um conjunto de elementos que compõem um software orientado a objetos e seus relacionamentos.
- Pode ser construída com diagramas para diferentes níveis de abstração, como por exemplo:

- **Diagrama de pacotes:** permite uma modelagem em um nível mais alto de abstração, dando uma visão mais geral da estrutura do sistema.

- **Diagrama de classes:** permite uma modelagem em um baixo nível de abstração, detalhando as estruturas para os objetos, suas características, comportamentos e seus relacionamentos.

Modelagem Dinâmica

- Representa as **funcionalidades** de um conjunto de elementos que compõem um software orientado a objetos e como interagem entre eles, os estados que podem assumir, entre outras funcionalidades do software.
- Pode ser construída com diagramas para diferentes níveis de abstração, como por exemplo:

- **Diagrama de Estado:** permite representar estados que os objetos podem assumir, suas transições e eventos do sistema que geram tais mudanças.

- **Diagrama de Casos de Uso:** permite demonstrar as funções executadas com o software e quem as executa.

UML
Unified Modelling Language

UML - Unified Modeling Language

É uma linguagem de diagramação ou notação para especificar, visualizar e documentar modelos de sistemas de software Orientados à Objeto.

UML é uma linguagem de modelagem e não uma metodologia

É voltada para o desenho de software Orientado à Objeto e tem um uso limitado para outros paradigmas de programação.

É controlada pelo Grupo de Gerenciamento de Objeto (Object Management Group - OMG) e é um padrão da indústria para descrever graficamente software.

Uso de UML

- A UML é usada no desenvolvimento dos mais diversos tipos de sistemas
- Abrange sempre qualquer característica de um sistema em um de seus diagramas
- É aplicada em diferentes fases do desenvolvimento de um sistema
  - Desde a especificação da análise de requisitos até a finalização com a fase de testes

Usos da UML

A UML pode ser utilizada para:

- **Mostrar a fronteira de um sistema e suas principais funções** usando Casos de Uso e Atores
- **Ilustrar realizações de Casos de Uso** com Diagramas de Interações
- **Representar a estrutura estática de um sistema** usando Diagramas de Classes e pacotes
- **Modelar o comportamento de objetos** com Diagramas de Transições de Estado
- **Revelar a arquitetura da implementação física** com Diagramas de Componentes e Distribuição
- **Estender sua funcionalidade** com Estereótipos

Visão Geral da UML

- Elementos de Modelagem
- Relacionamentos
- Mecanismos de Extensibilidade
- Diagramas

Elementos de Modelagem

Elementos estruturais

classes, interfaces, colaboração, casos de uso, classes ativas, componentes, nós

Elementos de comportamento

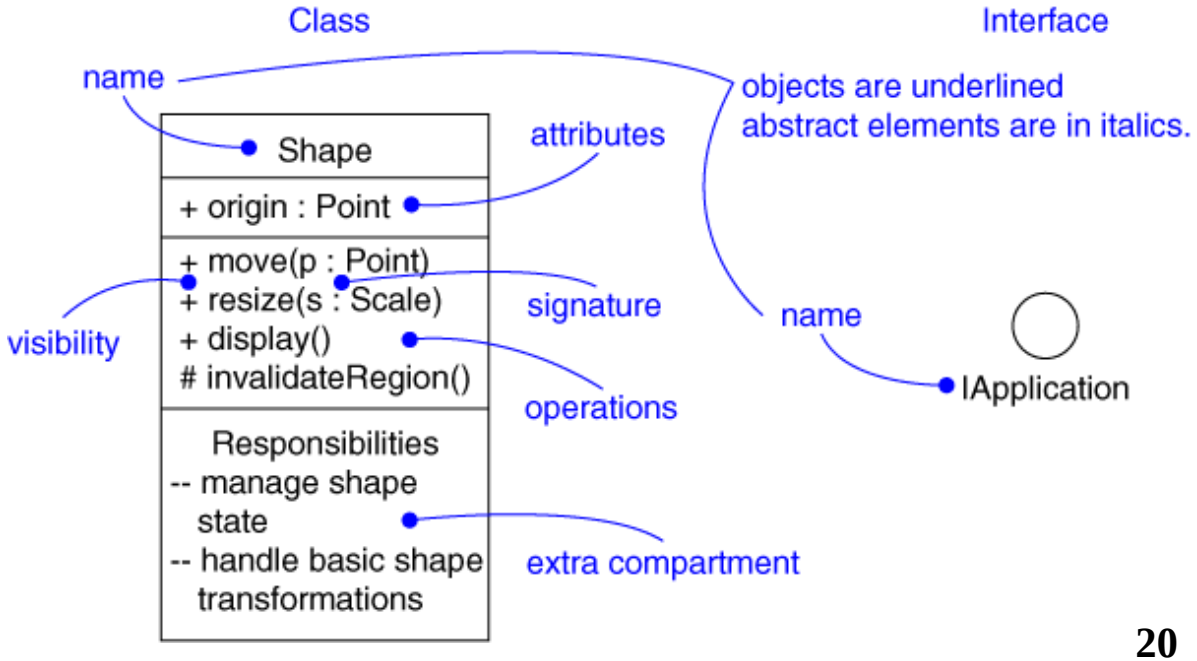
interação, máquinas de estado

Elementos de agrupamento

package (pacote), subsistemas

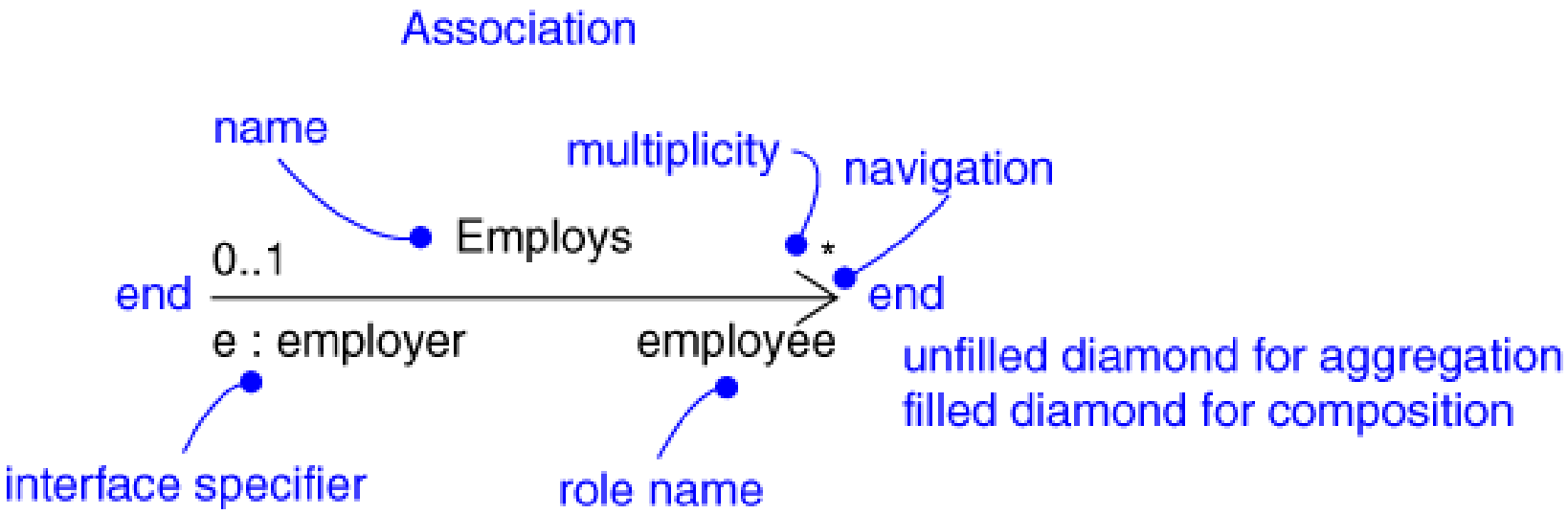
Outros elementos

notas (comentários)



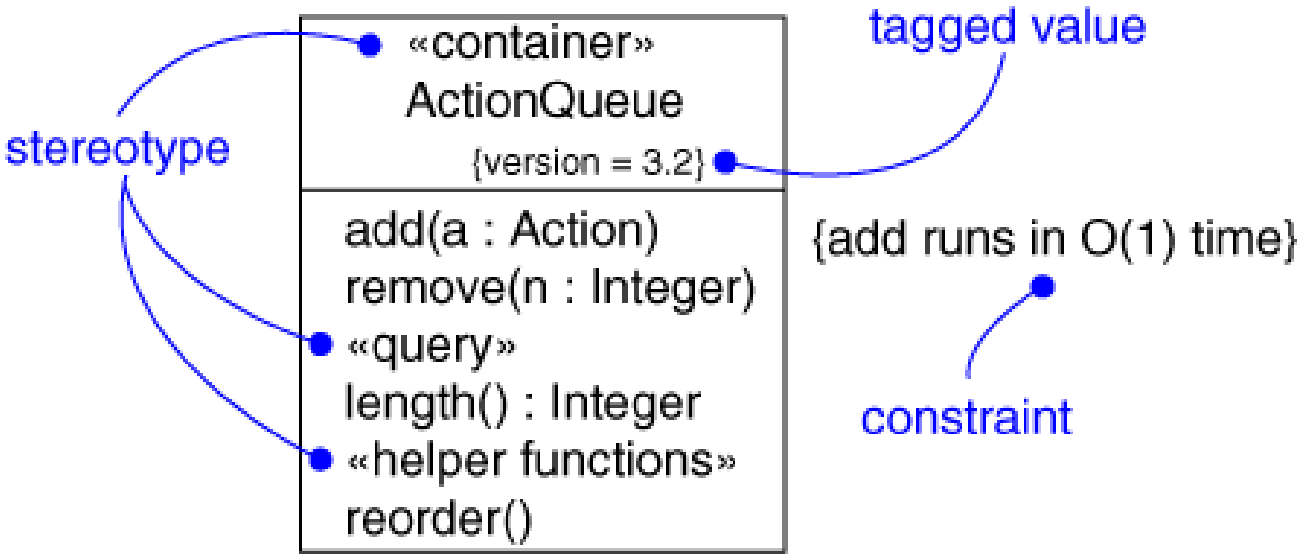
Relacionamentos

Dependência
Associação
Generalização
Realização

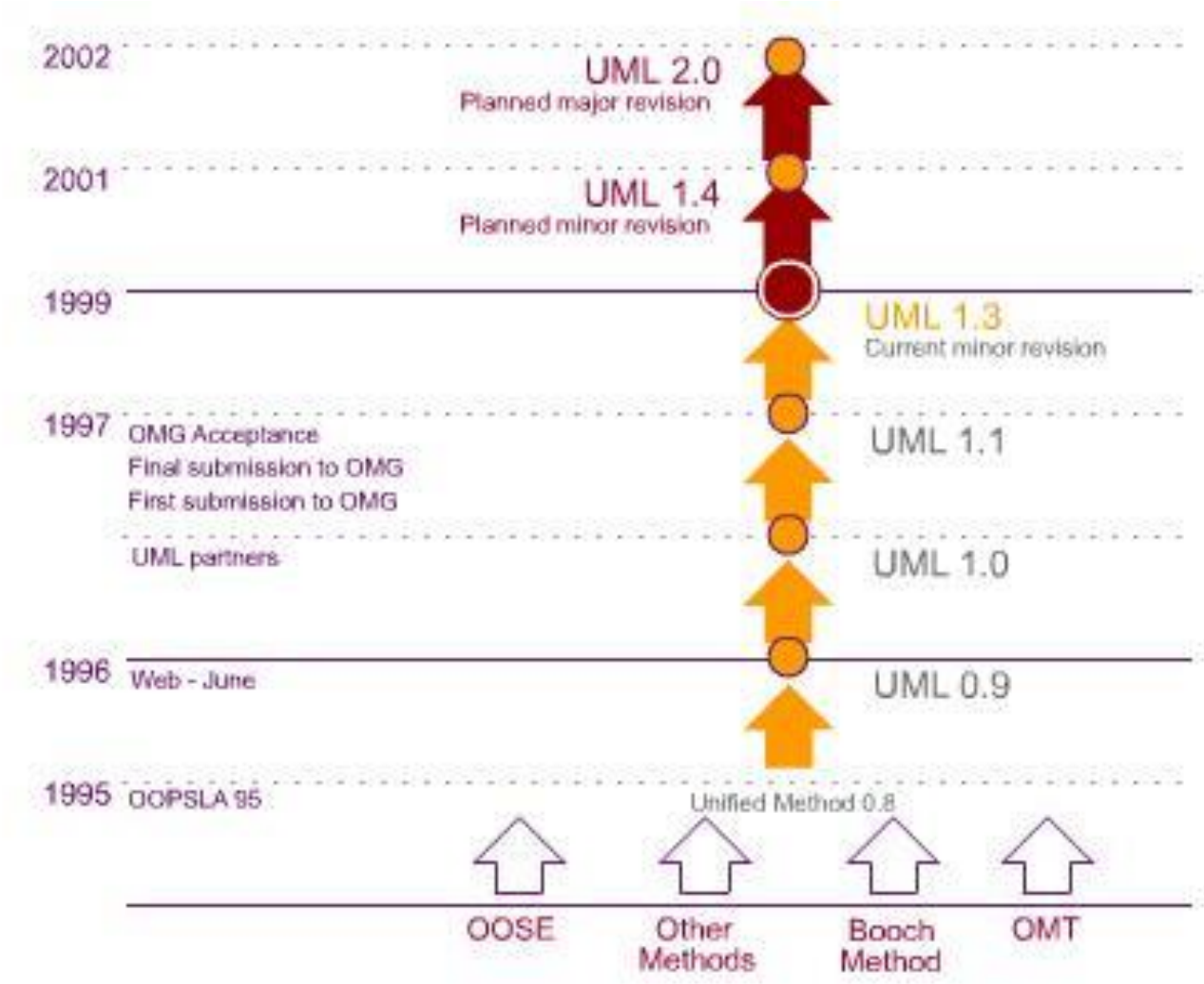


Mecanismos de Extensibilidade

Estereótipos
Valores marcados (Tagged value)
Restrições (Constraint)



UML – Evolução



UML – Versão atual

Unified Modeling Language™ (UML®)

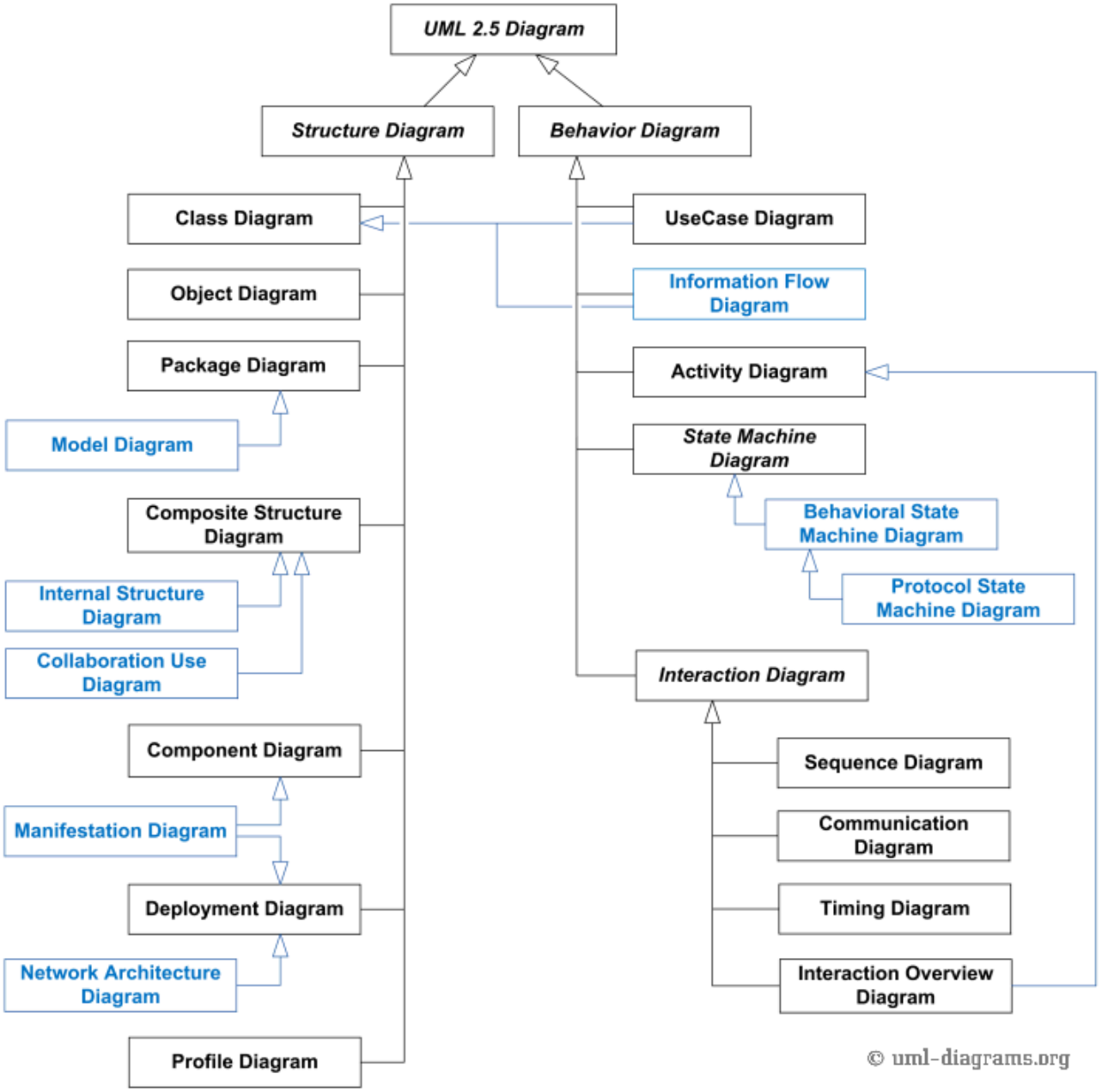
NOTE: There are no XSD files associated with Version 2.0

The current version is found at http://www.omg.org/spec/UML/Current

OMG Formal Versions Of UML®

Version	Release Date	URL
2.5	June 2015	http://www.omg.org/spec/UML/2.5
2.4.1	August 2011	http://www.omg.org/spec/UML/2.4.1
2.4	March 2011	http://www.omg.org/spec/UML/2.4
2.3	May 2010	http://www.omg.org/spec/UML/2.3
2.2	February 2009	http://www.omg.org/spec/UML/2.2
2.1.2	November 2007	http://www.omg.org/spec/UML/2.1.2
2.1.1	August 2007	http://www.omg.org/spec/UML/2.1.1

UML



UML 2.5 Diagrams Overview.
Note, items in blue are not part of official taxonomy of UML 2.5 diagrams.

UML – Diagramas UML 2.0

Modelagem Estrutural

Modelagem Dinâmica

Ponto de vista	Foco específico	Diagramas
Modelagem estrutural de sistema		Diagrama de classes Diagrama de objetos Diagrama de pacotes Diagrama de estrutura composta Diagrama de componentes Diagrama de utilização
Modelagem estrutural de classe		Diagrama de classes Diagrama de objetos
Modelagem dinâmica de sistema	Funcionalidades do sistema	Diagrama de casos de uso Diagrama de visão geral de interação Diagrama de atividades[1]
		Diagrama de máquina de estados[2]
	Interação de objetos	Diagrama de seqüência Diagrama de comunicação Diagrama de temporização[3]
Modelagem dinâmica de classe	Modelagem de estados	Diagrama de máquina de estados
	Algoritmos de métodos	Diagrama de atividades Diagrama de máquina de estados

Quadro A.1: Os diagramas de UML e os quatro pontos de vista essenciais

UML – Diagramas UML 2.0

Vamos trabalhar...

Modelagem Estrutural

Modelagem Dinâmica

Ponto de vista	Foco específico	Diagramas
Modelagem estrutural de sistema		Diagrama de classes Diagrama de objetos Diagrama de pacotes Diagrama de estrutura composta Diagrama de componentes Diagrama de utilização
Modelagem estrutural de classe		Diagrama de classes Diagrama de objetos
Modelagem dinâmica de sistema	Funcionalidades do sistema	Diagrama de casos de uso Diagrama de visão geral de interação Diagrama de atividades[1]
		Diagrama de máquina de estados[2]
	Interação de objetos	Diagrama de seqüência Diagrama de comunicação Diagrama de temporização[3]
Modelagem dinâmica de classe	Modelagem de estados	Diagrama de máquina de estados
	Algoritmos de métodos	Diagrama de atividades Diagrama de máquina de estados

Quadro A.1: Os diagramas de UML e os quatro pontos de vista essenciais

Diagramas

- Um diagrama é uma apresentação gráfica de uma coleção de elementos de modelo, freqüentemente mostrado como um gráfico conectado de arcos (relacionamentos) e vértices (outros elementos do modelo)

- A modelagem de um sistema com o uso de UML se dá por meio da notação definida pelos seus vários tipos de diagramas

Diagramas

Diagrama de Caso de Uso: mostra atores (pessoas ou outros usuários do sistema), casos de uso (os cenários onde eles usam o sistema), e seus relacionamentos

Diagrama de Classe: mostra classes e os relacionamentos entre elas

Diagrama de Seqüência: mostra objetos e uma seqüência das chamadas do método feitas para outros objetos.

Diagrama de Colaboração: mostra objetos e seus relacionamentos, colocando ênfase nos objetos que participam na troca de mensagens

Diagrama de Estado: mostra estados, mudanças de estado e eventos num objeto ou uma parte do sistema

Diagrama de Atividade: mostra atividades e as mudanças de uma atividade para outra com os eventos ocorridos em alguma parte do sistema

Diagrama de Componente: mostra os componentes de programação de alto nível (como Java Beans).

Diagrama de Distribuição: mostra as instâncias dos componentes e seus relacionamentos.

Diagramas UML

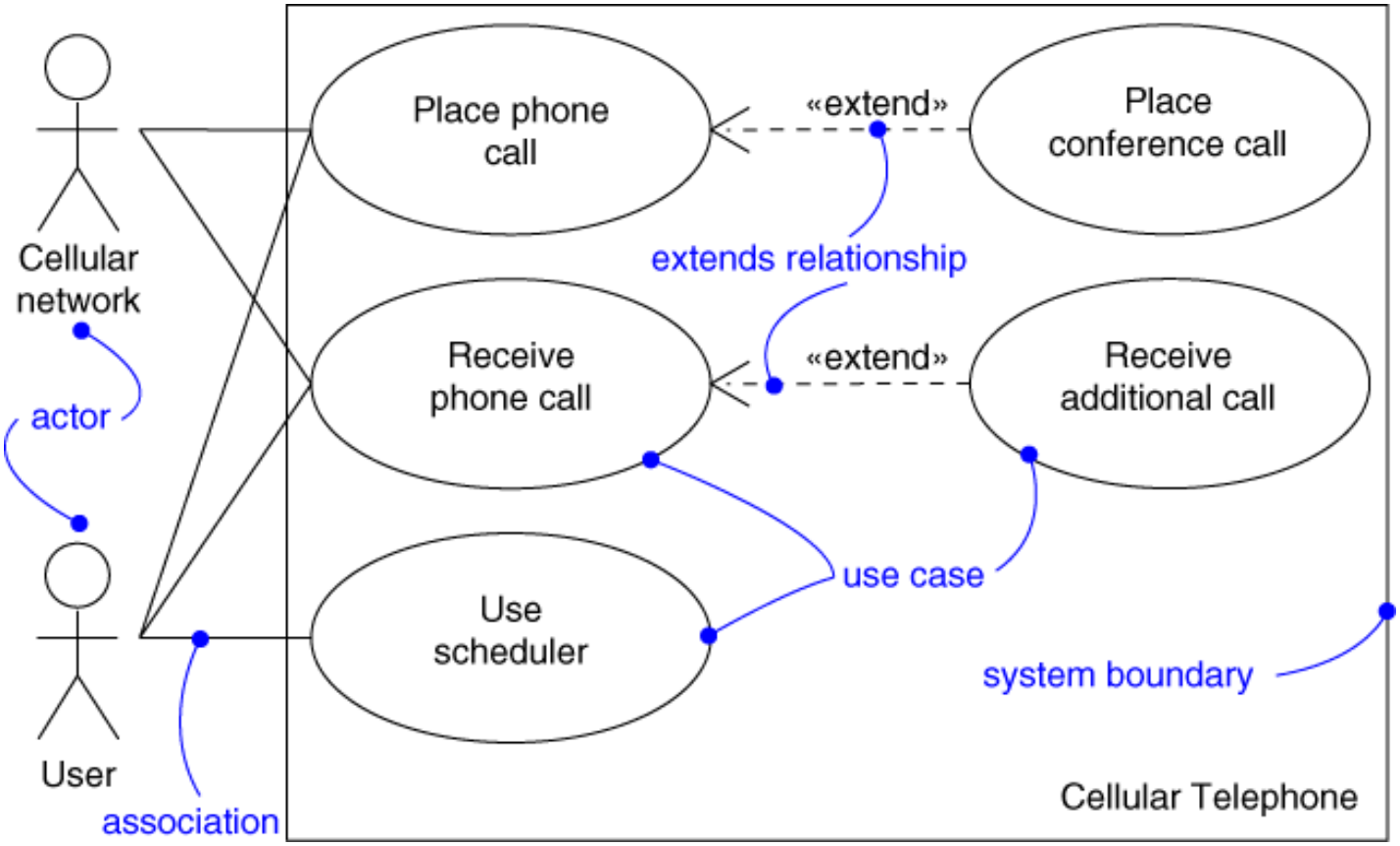
- Cada tipo de diagrama captura uma perspectiva diferente do sistema

- O conjunto de diagramas, juntamente com a documentação de suporte formam os elementos primários de modelagem de um sistema

- Para dar início a construção de qualquer diagrama da UML, primeiro deve-se conhecer a sua notação ou seja
  - A forma de representação
  - Sua semântica

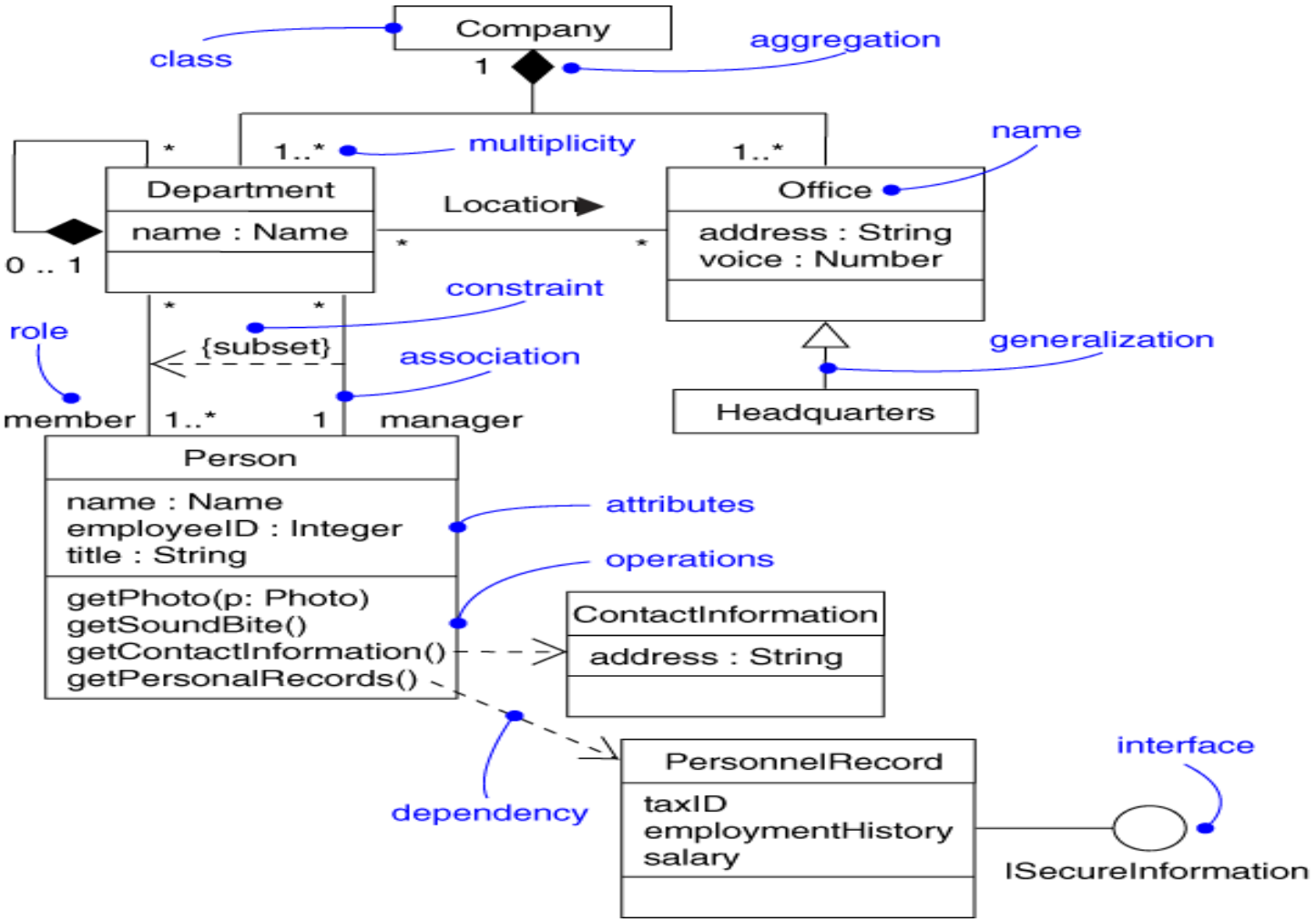
Exemplo de Diagrama de Casos de Uso

Captura a funcionalidade do sistema conforme vista pelos usuários



Exemplo de Diagrama de Classes

Captura classes e os relacionamentos entre elas



Exemplo de Diagrama de Atividades

Captura comportamento dinâmico (orientado por atividades)