

Programação Orientada a Objetos

Prof. Hugo Marcondes
hugo.marcondes@ifsc.edu.br

Aula 03

Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina

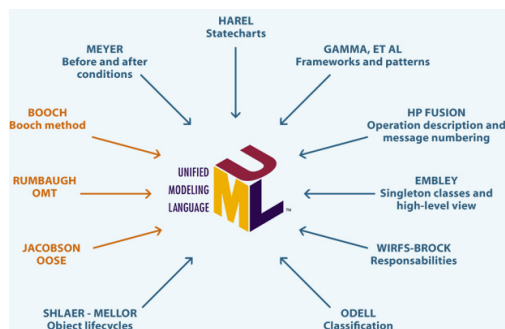


² IFSC - Programação Orientada a Objetos

UML

- A UML (Unified Modeling Language) é uma notação para descrição de sistemas orientados:
 - “The Unified Modeling Language for Object- Oriented Development” de Grady Booch, James Rumbaugh e Ivar Jacobson.
- Baseia-se na experiência dos principais autores dos 3 principais métodos OO.
- Esta notação foi padronizada pela OMG (Object Management Group) em 1997.

³ IFSC - Programação Orientada a Objetos



⁴ IFSC - Programação Orientada a Objetos

Uso da UML



- As diversas notações da UML podem ser utilizadas em várias situações:
 - Esboço e discussão sobre a estrutura de um sistemas.
 - Melhor entendimento na análise
 - Compreensão do que se está projetando
 - Documentação do projeto / sistema
 - Base para a codificação do sistema e elaboração de testes de funcionalidades
 - Documentação de estruturas já existentes
 - Engenharia reversa

5 IFSC - Programação Orientada a Objetos

Diagramas UML



- **Diagramas Estruturais**
 - Descrição estática de estruturas de um sistema
 - classes, atributos, operações e relacionamentos
 - Diagrama de Classe, Componentes, Pacotes
- **Diagramas Comportamentais**
 - Detalham o funcionamento (comportamento)
 - Diagrama de Casos de Uso, Atividades, Transição de Estados
- **Diagramas de Interação**
 - Subgrupo dos diagramas comportamentais
 - Interações entre objetos de uma aplicação
 - Diagramas de Sequência, Interatividade, Colaboração e Tempo

6 IFSC - Programação Orientada a Objetos

Diagramas Estruturais



- Diagrama de **Classes**
 - Conjunto de classes, interfaces e colaborações e seus relacionamentos
 - **Diagrama mais comum !**
 - Ilustram a visão estática do sistema
- Diagrama de **Componentes**
 - Partes internas, os **conectores** e **portas** que implementam um componente
- Diagrama de **Objetos**
 - Conjunto de objetos e seus relacionamentos
 - Visão estática do sistema, contudo considerando casos reais (objetos instanciados)
- Outros diagramas: **estrutura composta, artefatos, implantação**

7 IFSC - Programação Orientada a Objetos

Diagrama de Classe



- Cada classe é representada por um retângulo dividido em três partes
 - Nome
 - Atributos (Estado)
 - Operações (Comportamento)
- Modificadores são utilizados para indicar a visibilidade dos atributos e operações
 - '+' : visibilidade Pública
 - '#' : visibilidade Protegida
 - '-' : visibilidade Privada
- Por padrão, atributos são privados e operações são públicas

8 IFSC - Programação Orientada a Objetos

Diagrama de Classe

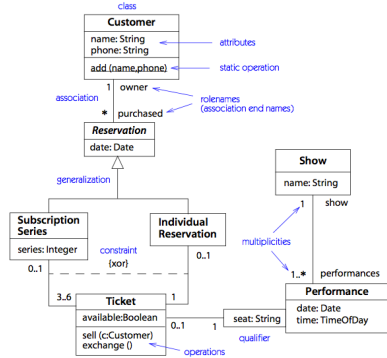


Figure 3-1. Class diagram

9 IFSC - Programação Orientada a Objetos

Diagrama de Componente

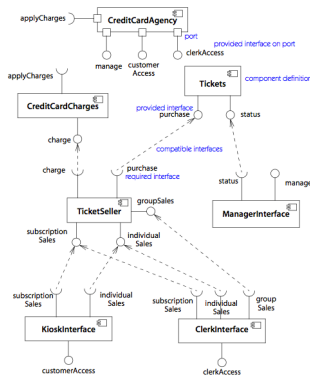


Figure 3-5. Component diagram

10 IFSC - Programação Orientada a Objetos

Diagramas Comportamentais

- Diagrama de **Casos de Uso**
 - Apresentação de funcionalidades e características do sistema
 - Relacionamento entre o sistema e os usuários e entidades assim como de que forma tais elementos se relacionam com usuários e entidades externas envolvidas num determinado processo.
- Diagrama de **Atividades**
 - Contempla as diversas tarefas desempenhadas na execução de uma atividade, sendo utilizado geralmente na representação de processos dentro de uma empresa/organização.
- Diagrama de **Transição de Estados**
 - Detalha os diferentes estados pelos quais pode passar um objeto, tomando por base a execução de um processo dentro do sistema que se está considerando.

11 IFSC - Programação Orientada a Objetos

Diagramas de Caso de Uso

- Atores:** Um papel que um usuário “interpreta” em relação ao sistema, incluindo pessoas reais como outros sistemas (ex. um robô, um sistema externo que utiliza uma informação do sistema modelado)
- Caso de Uso:** Um conjunto de cenários descrevendo a interação entre um usuário e o sistema, incluindo cenários alternativos



Actor



Use Case

12 IFSC - Programação Orientada a Objetos

Diagramas de Caso de Uso



- **Associação:** comunicação entre um ator e um caso de uso
Representado por uma linha sólida
- **Generalização:** relação entre um caso de uso genérico e um caso especial de uso (especificação de alternativas)
Representado por uma linha com uma flecha triangular apontando para o caso de uso “pai”.



13 IFSC - Programação Orientada a Objetos

Diagramas de Caso de Uso



- **Inclusão:** Uma linha pontilhada rotulada com “<<include>>” iniciando no caso de uso “base” e terminando com uma flecha apontando para o caso de uso incluído. Inclusão pode ser utilizado para refatorar partes de especificação comum a diversos casos de uso.



- **Extensão:** : Uma linha pontilhada rotulada com “<<extend>>” com uma flecha apontando para o caso de uso “base”.



14 IFSC - Programação Orientada a Objetos

Diagrama de Casos de Uso

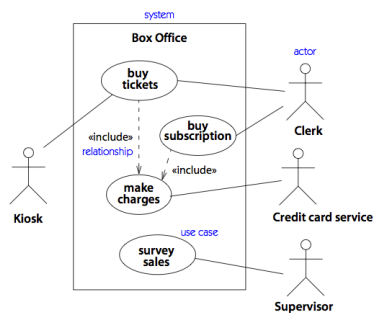


Figure 3-2. Use case diagram

15 IFSC - Programação Orientada a Objetos

Diagrama de Atividades

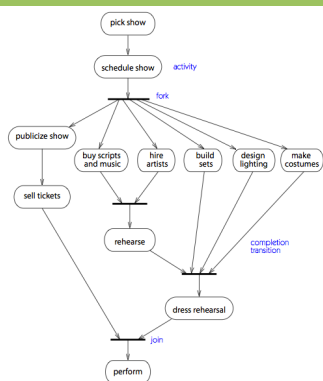


Figure 3-4. Activity diagram

16 IFSC - Programação Orientada a Objetos

Diagrama de Transição de Estados

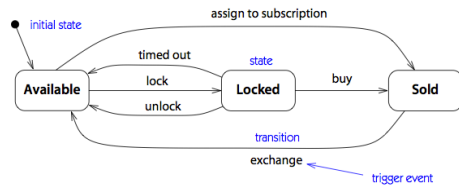


Figure 3-5. Statechart diagram

Diagramas de Interação



- Diagrama de **Sequência**
 - Interações entre diferentes objetos na execução de uma operação
 - Ordem em que tais ações acontecem em um intervalo de tempo
- Diagrama de **Colaboração** ou Comunicação
 - Similar a diagramas de sequência
 - Não apresenta estrutura rígida, geralmente derivado do diagrama de objetos

Diagrama de Sequência

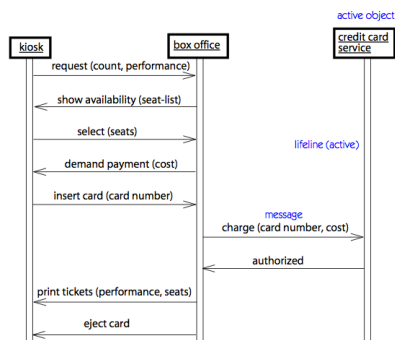


Figure 3-3. Sequence diagram

Diagrama de Colaboração

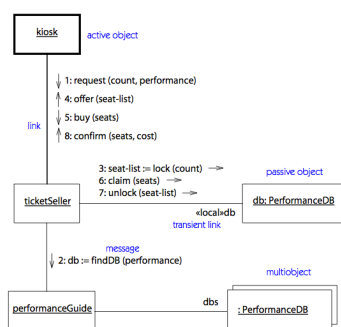


Figure 3-4. Collaboration diagram

Benefícios da UML



- A UML foca na representação visual de diferentes elementos e aspectos de um software
- Compreensão mais rápida, assim como abrangente, de componentes e funcionalidades que fazem parte de uma aplicação;
 - **Simplifica** a apresentação dos relacionamentos complexos entre as diferentes partes que compõe um sistema complexo
 - **Independente** de plataforma - melhor compreensão entre a equipe de um projeto complexo
 - Excelente para a demonstração de conceitos de **orientação a objetos** (é a sua origem)
 - Ênfase na **padronização** da linguagem, facilitando comunicação e transmissão de idéias

21 IFSC - Programação Orientada a Objetos

Cuidados no uso da UML



- Sincronização entre implementação e modelos UML
- Diagramas devem priorizar partes mais complexas ou críticas do sistema.
 - Documentar funcionalidades e estruturas relativamente simples pode não agregar muito ao projeto!
- Cuidado com diagramas muito extensos !
 - Podem dificultar a compreensão
 - Solução ? Diminuir escopo
 - Melhor entendimento

22 IFSC - Programação Orientada a Objetos

Ferramentas UML



- Gratuitas
 - ArgoUML - Java
 - Umbrella (KDE - Linux)
 - Papyrus (Eclipse)
 - <https://www.eclipse.org/papyrus/>
- Pagas
 - IBM Rational
 - Together
 - Poseidon

http://en.wikipedia.org/wiki/List_of_Unified_Modeling_Language_tools

23 IFSC - Programação Orientada a Objetos

Referências



- James Rumbaugh, Ivar Jacobson, and Grady Booch. 2004. **Unified Modeling Language Reference Manual, the (2nd Edition)**. Pearson Higher Education.
- IBM Rational
<http://www-306.ibm.com/software/rational/uml/>
- Practical UML — A Hands-On Introduction for Developers
http://www.togethersoftware.com/services/practical_guides/umlonlinecourse/

24 IFSC - Programação Orientada a Objetos
