

**SOFT**

**DISCIPLINA: Engenharia de software**

**AULA NÚMERO: 08**

**DATA: \_\_\_\_/\_\_\_\_/\_\_\_\_**

**PROFESSOR: Andrey**

## **APRESENTAÇÃO**

O objetivo desta aula é apresentar e discutir conceitos relacionados a modelos e especificações. Nesta aula serão discutidos os conceitos relacionados aos modelos de Análise Estruturada.

## **DESENVOLVIMENTO**

### **Introdução**

Modelos são ferramentas para representar as especificações a serem feitas durante o desenvolvimento do software. A atividade de especificação normalmente é feita em dois processos: construção de modelos e troca de mensagens entre grupos de pessoas.

O modelo de um sistema de software é uma representação da realidade que reflete certas características específicas e relevantes do sistema. Cada modelo pode representar um conjunto diferente de características em um determinado momento do processo de desenvolvimento. Por isso pode ser necessário usar vários modelos de um mesmo sistema para visualizá-lo. Um modelo é útil se ele consegue representar características relevantes do sistema. O uso de modelos nas especificações tem como objetivos:

- representar uma visão do ambiente antes da automação,
- indicar as diferentes alternativas de solução para um sistema,
- apontar as necessidades futuras para um sistema,
- permitir a avaliação e o refinamento das características do sistema,
- representar os componentes do sistema com partes bem definidas e com dependência mínima entre elas
- permitir que se trabalhe gradualmente com a complexidade
- fornecer informações quantitativas sobre o escopo e a complexidade do projeto.

### **Especificações**

O que é especificação? É uma atividade onde são obtidos e organizados os dados a respeito do domínio do problema. A atividade de especificação tem como objetivo mostrar propriedades do sistema. O que diferencia a especificação em cada metodologia distinta e a cada fase do desenvolvimento é o nível de abstração com que estas propriedades são mostradas.

Formalismo das especificações:

- Informais – linguagem natural usando figuras, tabelas e outras notações
- formais – sintaxe e semânticas precisas
- semiformais – Notação padronizada sem se preocupar com uma sintaxe precisa.

### **Linguagens formais:**

Linguagem de Restrição de Objeto (OCL) – Notação formal desenvolvida de modo que os usuários da UML possam adicionar mais precisão a suas especificações. Uma especificação OCL é feita em texto ASCII e inclui vários elementos da lógica e matemática discreta.

Linguagem Z – Aplica conjuntos, relações e funções do contexto da lógica de predicados para construir

esquemas para estruturar uma especificação formal.

## **Linguagens Semiformais**

A Linguagem de modelagem unificada (UML) começou a ser desenvolvida em 1994 por Booch, Rumbaugh e Jacobson, num esforço para unificar as notações das diferentes metodologias Orientadas a Objetos que existiam. Em 1997 ela foi adotada (na sua versão 1.0) pela Object management group (OMG). Hoje ela está na versão 2.1. A maior influência da UML veio da OMT. Do método Booch, veio a capacidade de especificar detalhes de projeto de baixo nível de abstração, além da especificação de componentes. Do método OOSE, veio a notação de Casos de Uso.

Na UML são definidos 13 diagramas, divididos em 2 grandes grupos: diagramas estruturais e diagramas comportamentais. Entre os diagramas estruturais estão: Diagrama de classes, componentes, objetos, implantação, composição de estrutura, pacotes. Entre os diagramas comportamentais estão: Diagramas de casos de uso, comunicação, sequência, interaction overview, estados, temporização e atividades.

A UML é usada principalmente para se fazer modelos para especificações de software orientados a objetos. Recentemente, a OMG lançou uma linguagem chamada SYSML para modelar sistemas complexos, que podem integrar software, hardware, pessoas, procedimentos e facilidades. A SYSML usa um subconjunto da UML 2.0 além de apresentar novos diagramas, como o diagrama de requisitos, por exemplo.

## **Modelos**

Os modelos devem seguir os princípios de engenharia de software como abstração e decomposição, além de permitir a formalização da especificação (semi formal ou formal). O uso de modelos no desenvolvimento do sistema normalmente segue uma abordagem top-down. Um modelo pode ser usado em mais de um nível de abstração.

Carvalho e Chiossi (2001) sugerem o uso de 3 modelos para representar sistemas mais complexos: Modelo do mundo real, modelo de projeto e modelo do plano de projeto.

### **Modelos de análise (ou do mundo real)**

O modelo de análise tem os seguintes objetivos: Descrever o que o cliente exige, estabelecer a base para a criação de um projeto de software e definir um conjunto de requisitos que possam ser validados quando o software for construído (Pressman, 2006)

O modelo de análise faz a ponte entre uma descrição em nível de sistema que descreve sua funcionalidade global, hardware, dados, pessoas e outros elementos, e o modelo de projeto que descreve a arquitetura, interface do usuário e estrutura em nível de componente da aplicação.

O modelo de análise deve manter um nível de abstração mais elevado procurando representar características do sistema que sejam relevantes para os interessados no sistema, procurando abstrair informações mais detalhadas sobre arquitetura e componentes.

Modelos de análise (Carvalho e Chiossi):

- Modelo de funções
- Modelo de dados
- Comportamental (estados)
- Modelo de objetos

- Modelo formal
- Modelo dinâmico
- Dicionário de dados

#### Modelos de análise (Pressman)

- Modelagem de dados
  - Modelo Entidade - Relacionamento
- Modelagem de análise Orientada a Objetos
  - Casos de uso
  - diagramas de atividades
- Modelagem de análise orientada a Fluxo
  - DFD níveis 0 e 1
- Modelagem de Análise baseada em classes
  - Classes de análise: atributos, operações relacionamentos
  - Modelagem CRC
- Modelo modelagem de análise comportamental
  - Diagrama de estados

### **Modelos de projeto**

Enquanto que o modelo de análise representa o “mundo real”, ou seja o domínio do negócio, o modelo de projeto representa a solução do problema, do ponto de vista dos componentes necessários para construir a solução e seus relacionamentos.

O modelo de análise é refinado e transformado no modelo de projeto. Os modelos de projeto usam os mesmos diagramas UML que foram usados no modelo de análise. A diferença é que esses diagramas são refinados e elaborados como parte do projeto, mais detalhamento específico de implementação é acrescentado e são enfatizadas a estrutura e seus componentes.

Modelos de projeto:

- Classes de projeto
- Modelo de Projeto do fluxo de dados
  - DFD dos níveis mais baixos
- Modelo de Projeto de dados - Jakson e Warnier
- Modelo de Projeto arquitetural
- Modelo de Projeto de Componente
- Modelo de Projeto da interface
- Modelos para especificar funções
  - Linguagem de projeto de programas
  - fluxograma
  - Tabela de decisão

### **Comportamento do Sistema**

#### **Métodos estruturados**

- Métodos estruturados fornecem formulações para a modelagem detalhada de sistemas.
- Métodos definem um conjunto de modelos, um processo para derivar esses modelos e regras e diretrizes que se aplicam aos modelos.

- As ferramentas CASE estão disponíveis para dar apoio aos métodos.

Deficiências dos métodos de análise estruturada

Não modelam requisitos não funcionais

Não incluem informações sobre se um método é apropriado para um determinado problema.

Frequentemente produzem muita documentação.

Os modelos produzidos são muito detalhados e de difícil compreensão pelos usuários.

### Modelos (Perspectiva interna)

Modelos funcionais

Modelos de fluxo de dados

Modelos de comportamento

Modelos de máquinas de estado

Modelos de dados

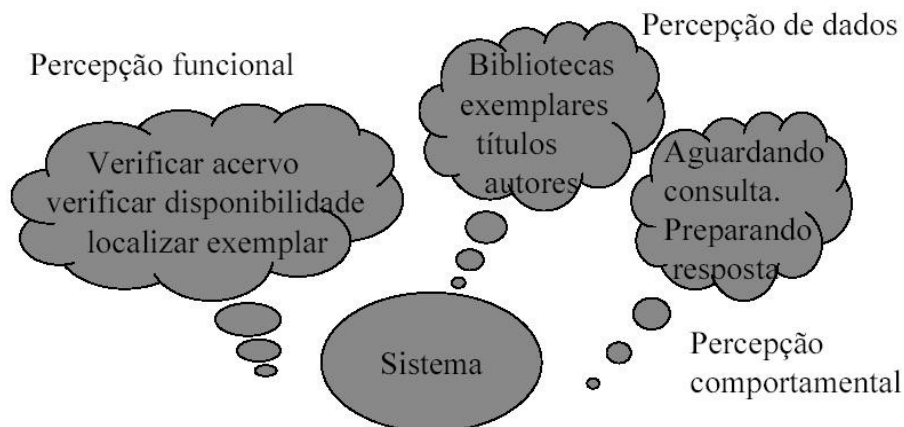
Modelos de Objeto

Modelos de herança

Agregação de objetos

Modelagem de comportamento de objetos

Percepções funcional, de dados e comportamental de um subsistema de consulta a bibliotecas



### Modelos do mundo real

Modelo funcional do sistema

- Representa a percepção do que o sistema faz.
- No exemplo da biblioteca, tem-se as seguintes funções: verifica acervo, verifica disponibilidade e localiza exemplar.

Modelo de dados

- Representa a percepção dos dados que o sistema mantém.
- Para o subsistema de consulta a bibliotecas, os dados mantidos pelo sistema são: bibliotecas, exemplares, títulos e autores.

Modelo comportamental

- Representa a percepção de como o sistema se comporta em resposta a certos eventos externos.
- Focaliza os dados (objetos do sistema) ou as funções.

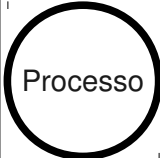


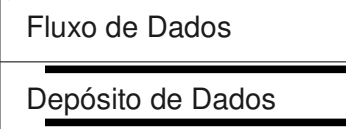
- Para o subsistema de consulta a bibliotecas, identifica-se os seguintes estados para as funções do sistema: aguardando consulta do usuário e preparando resposta à consulta.

### Modelo de Função – DFD Diagrama de Fluxo de Dados

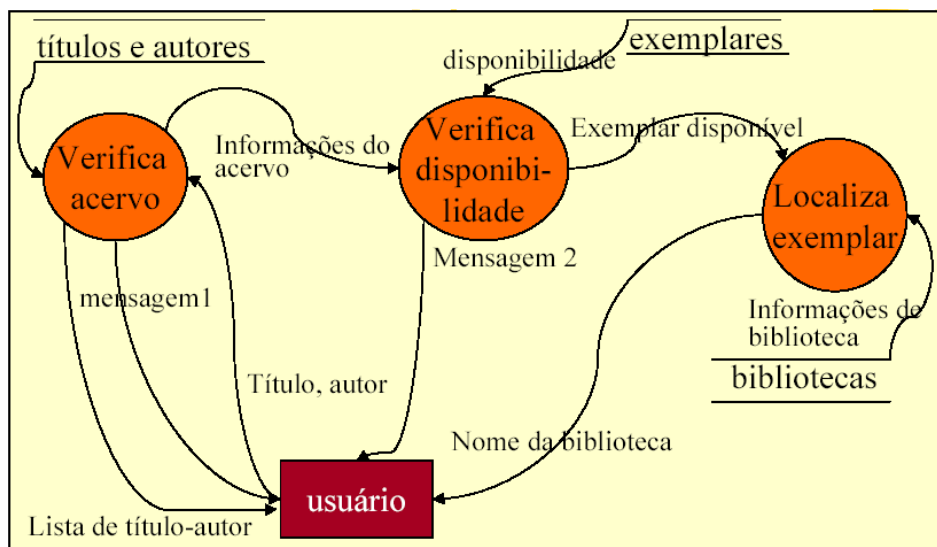
Técnica gráfica que descreve o fluxo de informação e as transformações que são aplicadas à medida que os dados se movimentam da entrada para a saída

Um sistema baseado em computador é representado como uma transformação de informação

Componentes:

 Processo	Um transformador de informação (função) que reside dentro dos limites do sistema a ser modelado
 Entidade Externa	Um produtor ou consumidor de informações que reside fora dos limites do sistema a ser modelado
 Fluxo de Dados	Usada para representar conexão, chamada de fluxo de dados
 Depósito de Dados	Um repositório de dados pode ser usado por um ou mais processos. Pode ser tão simples quanto um buffer ou tão sofisticado quanto um banco de dados relacional

Exemplo: DFD do subsistema Consultar acervo



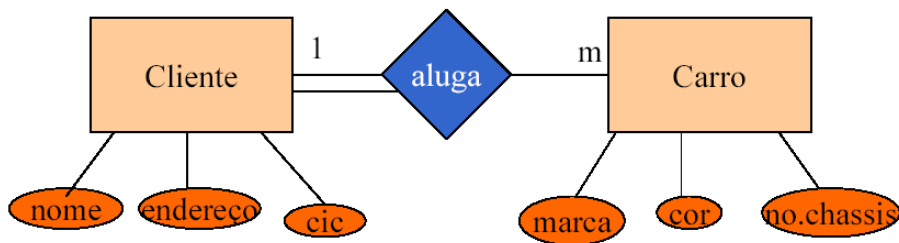
## O Modelo de Dados

- Quais são os dados a serem armazenados pelo sistema?
- Qual é a melhor organização desses dados?
- Quais são os relacionamentos entre grupos de dados?
- Como os dados serão utilizados?

### Componentes do Modelo de Dados

- Entidade (ou objeto): Representa um conceito do mundo real
- Atributos: Características que definem uma entidade
- Relacionamentos: Conexões relevantes entre objetos do mundo real.
- Cardinalidade: ocorrências de um objeto que pode ser relacionado com o número de ocorrências de outro objeto;

Exemplo de um modelo Entidade-Relacionamento

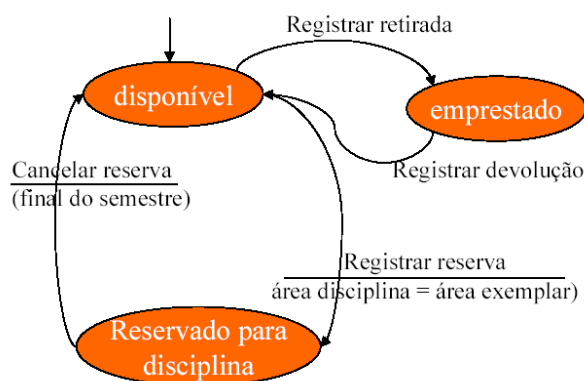


## O modelo comportamental

- Representa os estados e os eventos que alteram esses estados
- Eventos: representados através de condições e ações para mudanças de estado.
- Quando a condição é verdadeira, a ação correspondente é ativada.
- Quando a ação é completada, o componente passa ao estado determinado pelo evento correspondente.

## Máquinas de Estados Finitos

exemplo: MEF para os títulos de uma biblioteca (exemplar)



## **ATIVIDADE**

1. Qual a importância de uma especificação?
2. Quais os tipos de especificações?
3. Qual a diferença entre modelos de análise e de projeto?
4. Quais os principais modelos de análise?
5. O que representa um DFD?
6. O que representa Modelo Entidade Relacionamento?
7. Construa uma máquina de estados finitos que represente um funcionário em uma empresa. Construa o Modelo Entidade Relacionamento para o subsistema de Consulta a bibliotecas. O sistema recebe como entrada um título ou um autor. Quando título e autor são fornecidos pelo usuário, o sistema deve consultar o acervo da universidade e verificar se o par título-autor pertence ao acervo. Em caso afirmativo, o sistema deve verificar se há disponibilidade de exemplares correspondentes ao par título-autor e, nesse caso encontrar a localização desses exemplares. Em caso negativo, o sistema deve informar ao usuário que o par não pertence ao acervo. Quando apenas o título é fornecido, o sistema deve listar todas as ocorrências do título no acervo. Quando apenas o autor é fornecido, o sistema deve informar ao usuário todos os títulos daquele autor pertencentes ao acervo.

## **BIBLIOGRAFIA BÁSICA**

PRESSMAN, R. S.. Engenharia de Software. Makron Books. 1995

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I.. UML guia do usuário. Editora Campus. 2000.

BEZERRA, E.. Princípios de Análise e Projeto de Sistemas com UML. Editora Campus. 2003.

CARVALHO, A. M. B. R.; CHIOSSI, T. C. S.. Introdução à engenharia de software. Editora da Unicamp. 2001.