AULA 03 – Modelagem de Software

Introdução:

Em nossa terceira aula, falaremos sobre a modelagem de software, desde os primeiros tipos de diagramas e modelos até a atual linguagem unificada de modelagem de sistemas. Falaremos sobre os aspectos dinâmicos e estáticos no processo de modelagem de software.

Contextualizando:

Modelar ou não sistemas através de diagramas é uma decisão em conjunto com o tipo de projeto e o modelo de processo a ser adotado. Há modelos de processo que não recomendam o uso de diagramas, porém, modelar os principais aspectos de um sistema é bastante interessante para geração de documentação e compreensão do mesmo.

MODELAGEM DE SOFTWARE

Entende-se por método o caminho a ser percorrido através de etapas, aplicando-se um conjunto de técnicas, permitindo a construção de um software eficiente e seguro. Métodos sempre envolvem tarefas, tais como:

1. Planejamento do projeto
2. Análise de requisitos
3. Projeto de estruturas de dados, arquitetura e algoritmos
4. Codificação, teste e manutenção.

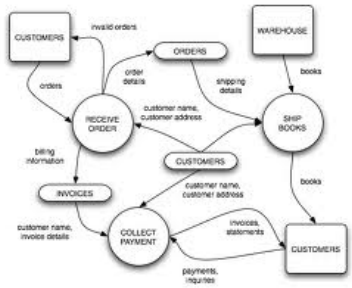
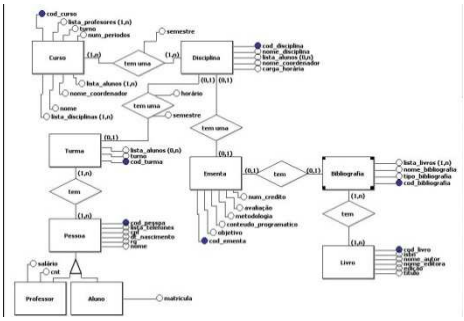
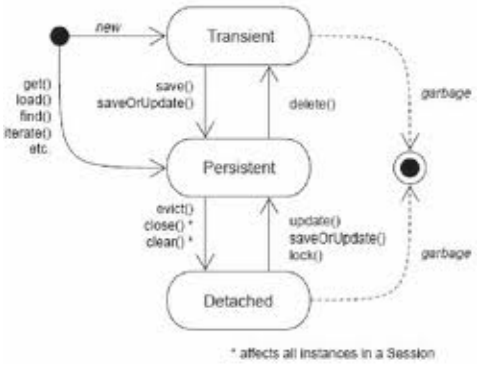
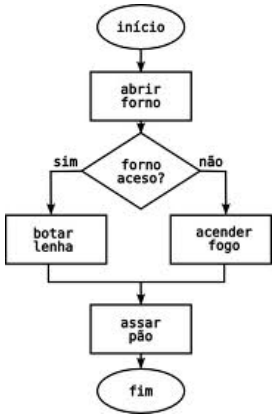
Utilizamos um método para facilitar o treinamento de novos integrantes de nossa equipe de desenvolvimento e para eliminar perdas na falta de controle de resultados.

Alguns tipos de modelagem de software

1. Estruturado: possui uma visão macro e parte-se de um marco zero para se enxergar a totalidade. Ou então uma visão todo-parte, parte-se da totalidade

em visões cada vez menores. Outras características de um modelo estruturado é possuir características top-down, modelagem de banco de dados e de processos.

Exemplos de diagramas da modelagem estruturada:

<table>
<tr>
<td>Diagrama de Fluxo de Dados
 
</td>
<td>Modelo Entidade-Relacionamento
  </td>
</tr>
<tr>
<td>Diagrama de Estados
 
</td>
<td>Fluxograma
 
</td>
</tr>
<tr>
<td>Dicionário de Dados
<table>
<tr><td colspan="4">LIVRO: Relação que armazena os dados dos livros das bibliografias.</td></tr>
<tr><th>Atributo</th><th>Descrição</th><th>Tipo</th><th>Restrições</th></tr>
<tr><td>Cod-livro</td><td>Atributo que identifica o código de identificação do livro.</td><td>Inteiro</td><td>Chave Primaria</td></tr>
<tr><td>Isbn</td><td>Atributos que representa a identificação numérica do livro segundo titulam, autor, edição, editora, pais.</td><td>Varchar</td><td>Não nulo</td></tr>
<tr><td>Titulo</td><td>Atributo que representa o nome do título do livro.</td><td>Varchar</td><td>Não nulo</td></tr>
<tr><td>Nome_autor</td><td>Atributo que representa o nome do autor do livro.</td><td>Varchar</td><td>Não nulo</td></tr>
<tr><td>Edição</td><td>Atributo que representa o número da edição do livro.</td><td>Varchar</td><td>Não nulo</td></tr>
<tr><td>Nome_editora</td><td>Atributo que representa o nome da editora do livro.</td><td>Varchar</td><td>Não nulo</td></tr>
</table></td>
<td>Pseudo-código

```
I : = 1            {índice de trabalho}
LIM : = MAX        {tamanho do vetor a classificar}
enquanto (I < LIM) faça
    COR : = V [ I ]      {elemento corrente}
    INU : = I
    J : = I + 1
    enquanto (J < = LIM) faça
        se V [ J ] < COR então
            COR : = V [ J ]
            INU : = J
        fimse
        J : = J + 1
    fimenquanto
    AUX : = V [ I ]
    V [ I ]: = V [INU]
    V [ INU ] : = AUX
    I : = I + 1
fimenquanto
```

</td>
</tr>
</table>

2. Essencial: O modelo essencial de sistemas surgiu em decorrência do grande número de diagramas e documentação que a modelagem estruturada demandava. Este modelo tenta utilizar o mínimo possível de diagramas e que possuam uma manutenção efetiva.

3. Orientado a objetos: A UML é composta por diagramas derivados das metodologias de Rumbaugh, Coad e Booch. Utiliza-se da abstração de objetos da realidade mapeados para os sistemas. Possui uma organização que mantém as especificações de seus diagramas (OMG – Object Managment Group).

MODELAGEM DE SOFTWARE ORIENTADA A OBJETOS – UML

Figura 1 – Diagramas da UML

A linguagem de modelagem unificada é muito mais difundida e utilizada com efetividade que os modelos anteriores por conseguir representar conceitos reais

através de objetos representados em uma linguagem visual. Sua principal característica, quando utilizada no nível de análise de requisitos é ser independente de linguagem de programação e processo de desenvolvimento. Toda a especificação da UML é encontrada no link www.omg.org.

A UML trabalha com as visões de:

1.Casos de uso
2. Projeto
3. Implementação
4. Implantação e
5. Processo.

Possibilita a construção de modelos comportamentais, que representam as funcionalidades do sistema, além dos modelos estáticos, que mapeiam classes, objetos e relacionamentos do sistema.

A figura 2 destaca a divisão dos diagramas em modelos dinâmicos e estáticos.

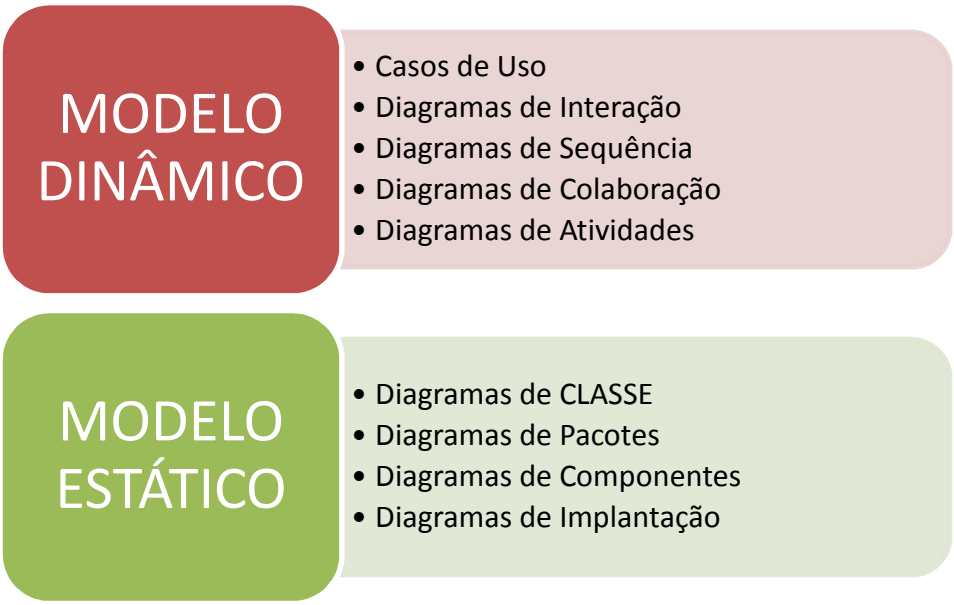


Figura 2 – Diagramas da UML

Diagrama de Casos de Uso

Estes diagramas representam as funcionalidades externas observáveis do sistema e de elementos que interajam com o sistema. Sua concepção ocorreu em 1970 por Ivar Jacobson, o qual trabalhava na Ericson como engenheiro. O diagrama foi incorporado à UML juntamente com os diagramas de Rumbaugh e Booch. Ele direciona o restante do ciclo de vida do sistema. Sua modelagem é centrada no usuário. Representa quem faz o que com o sistema, sem considerar o comportamento interno do sistema.

Elementos do Diagrama:

O ATOR

- Stakeholders ou outros sistemas externos que interagem com o sistema.
- Corresponde a um papel.
- Um ator pode participar de n casos de uso.

O CENÁRIO

- Descrição de uma das maneiras pelas quais um caso de uso pode ser realizado.

O RELACIONAMENTO

- Casos de uso e atores não existem sozinhos.
- Tipos: comunicação, inclusão, extensão e generalização.

Este diagrama:

- Descreve o que acontece dentro do sistema
- Ajuda na comunicação entre *Stakeholders* e desenvolvedores
- Demonstra as funcionalidades do sistema
- Captura o comportamento do sistema

O que especificar:

- Identificador do caso de uso
- Breve descrição
- Ator
- Prioridade
- Requisitos não funcionais associados
- Pré condições
- Pós condições
- Fluxo de eventos principal
- Fluxos secundários: alternativos e de exceção
- Interfaces associadas

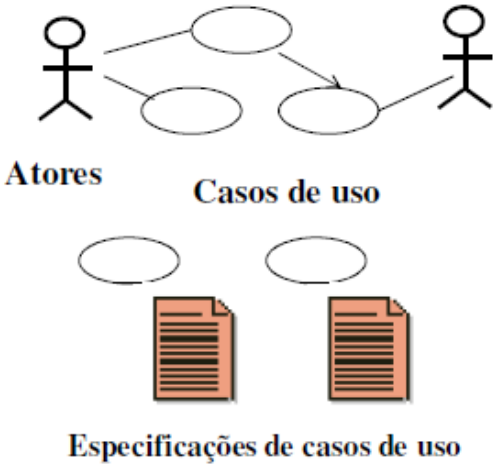


Figura 3 : Principais elementos do diagrama de casos de uso

Diagrama de Classes

Este diagrama permite a compreensão da estrutura interna para que as funcionalides externas sejam produzidas. É um modelo estático porque demonstra a estrutura das classes de objetos e as relações entre as mesmas.

- **Tipos de diagramas de classes:**

  - Domínio: representa classes do domínio do negócio. Construído na fase de ANÁLISE.

Especificação: extensão do modelo de domínio. Adição de detalhes específicos à solução escolhida. Construído no PROJETO.

Implementação: extensão do modelo de especificação. Classes já descritas na linguagem de programação. Construído na IMPLEMENTAÇÃO do software.

- **Elementos importantes:**

    - CLASSE
    - Atributo
    - Método
    - Relacionamento

Podem ser dirigido por dados: ênfase na identificação da estrutura dos conceitos relevantes. Ou dirigidos por responsabilidades: identificado nas responsabilidades que cada classe deve ter dentro do sistema.

- RELACIONAMENTOS entre as classes:
    - Associação
    - Agregação
    - Generalização

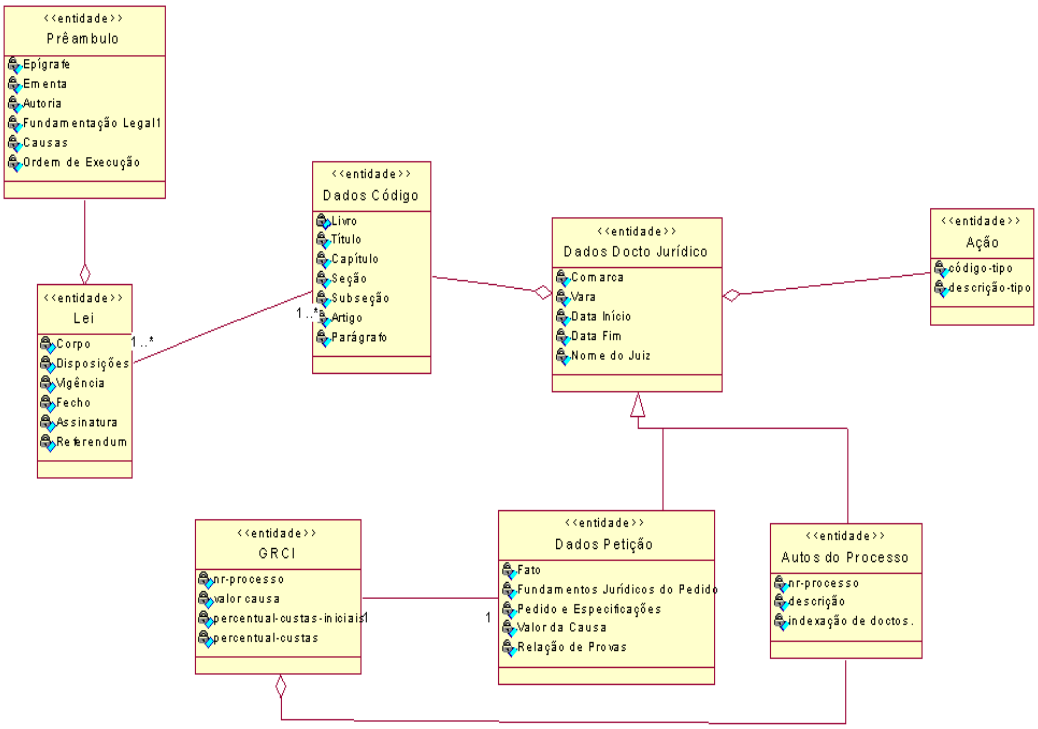


Figura 4 – exemplo de um diagrama de classes

Diagramas de Interação – Sequência e Colaboração

Estes diagramas consolidam o entendimento dos aspectos dinâmicos do sistema, iniciado nos diagramas de casos de uso. Eles interagem através da troca de mensagem, demonstrando a comunicação entre atores e objetos dentro do sistema.

- São dois tipos:

Sequência

Colaboração.

- Elementos importantes:

Conexões

Mensagens

  - Fluxo de controle

Diagrama de sequência:

A troca de mensagens ocorre dentro de uma LINHA DO TEMPO.

- Principais Elementos:
  - Atores
  - Objetos
  - Classes
  - Linhas da vida
  - Loop
  - Condição
  - Recursividade

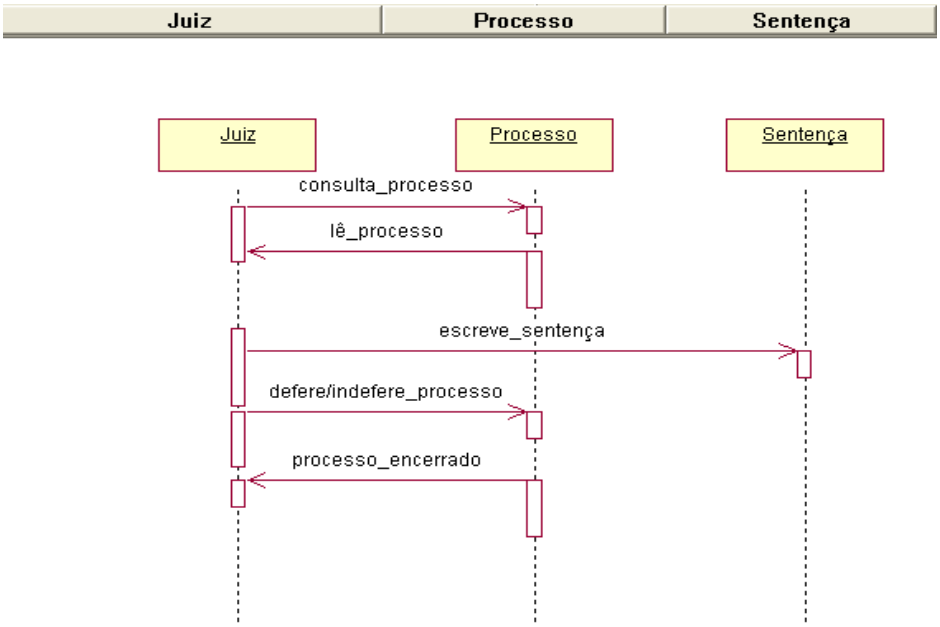


Figura 4: Exemplo de um diagrama de sequência

Diagrama de Colaboração

É o tipo de diagrama que demonstra a troca de mensagens entre os objetos com foco nas mensagens trocadas. Similar ao diagrama de sequência, porém são adicionados setas e rótulos de mensagens nas ligações entre objetos.

- Elementos importantes:
  - Atores
  - Objetos
  - Classes
  - Linhas da vida
  - Loop
  - Condição
  - Recursividade
  - Rótulos de mensagens

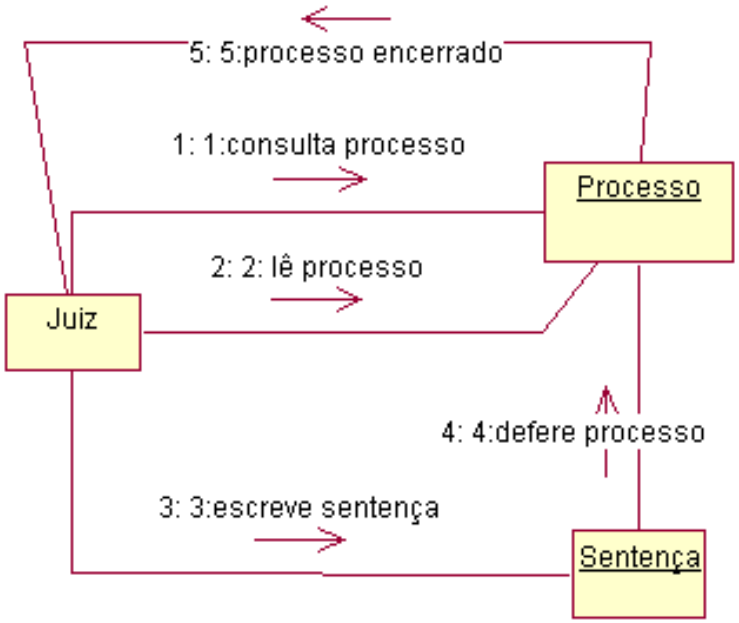


Figura 5: Exemplo de diagrama de colaboração

Diagrama de Atividades

É outro tipo de diagrama que especificam o comportamento de uma entidade (objeto ou CLASSE). Tipo especial de diagrama de estados, no qual representam-se os estados de uma atividade. Orientado a fluxos de controle.

Elementos principais:

- Estado da ação
- Estado da atividade
- Estados inicial, final e condição
- Sincronização
- Raias
- Fluxo de objeto

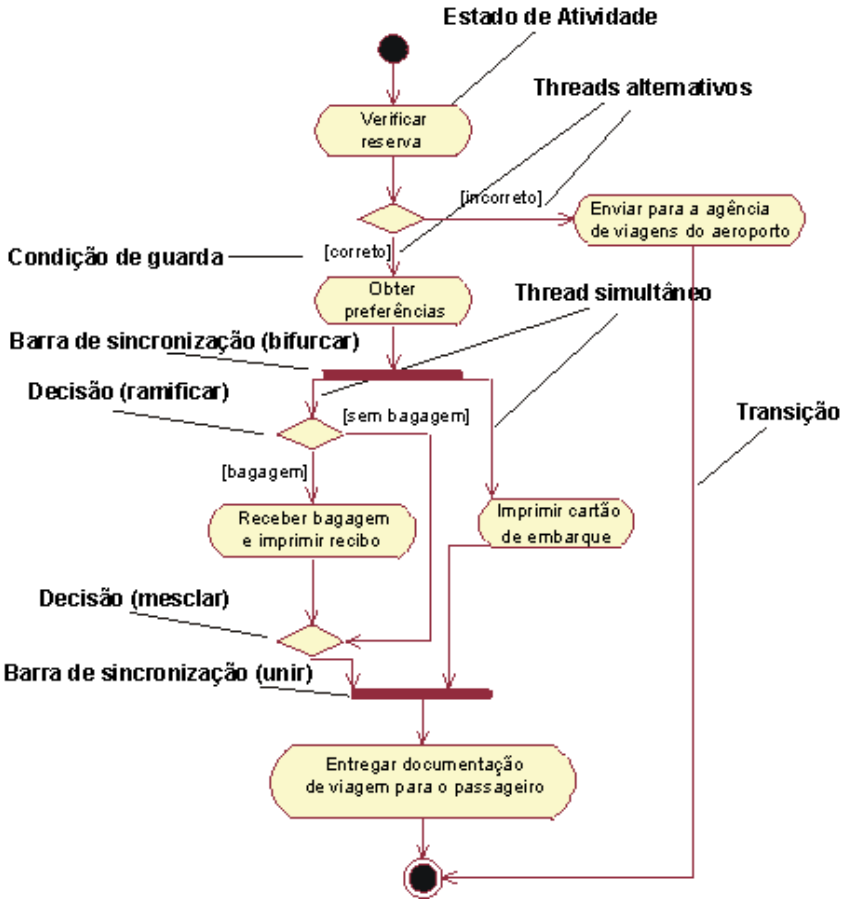


Figura 6: Exemplo de diagrama de Atividades

Construímos uma visão geral dos diagramas da UML. O ideal, caso seja seu interesse conhecer melhor cada diagrama, sugiro buscar livros dos próprios idealizadores da UML, bem como, pesquisar no link da www.omg.org.

Pesquisa

Desenvolva uma pesquisa sobre opiniões favoráveis e desfavoráveis sobre o uso de UML com métodos ágeis. Há uma grande polêmica em torno do uso de documentação de sistemas com métodos ágeis. Será bem interessante para você conhecer o ponto de vista de desenvolvedores de diversas plataformas e aplicações.

Trocando Ideias

Modelos do sistema com o uso de diagramas tornam a compreensão do domínio do problema e suas delimitações mais rápida. Particularmente eu gosto da

ideia de utilizarmos um diagrama de casos de uso, para documentação dos requisitos, um modelo de classes, para compreensão da estrutura do sistema, bem como um modelo entidade-relacionamento, caso o banco de dados seja do tipo relacional.

Síntese

Nessa aula tivemos uma visão geral sobre vários tipos de diagramas, em especial da UML. Também falamos um pouco sobre os precursores da modelagem de sistemas.

Compartilhando

Tente imaginar uma aplicação que você gostaria de desenvolver durante o seu curso de graduação. Vá estabelecendo os principais modelos de suas ideias, até o momento que você consiga implementá-lo através de uma linguagem de programação.

Autoavaliação

- Após a escolha do modelo de processos, é importante estabelecer o nível de documentação através de modelagens de sistema. Comente sobre as vistas neste encontro.
- Quais os diagramas da UML que representam as funcionalidades do sistema. Modelo dinâmico.
- Com quais diagramas da UML podemos documentar a estrutura estática do sistema?
- E quanto aos diagramas de colaboração, sequência e atividade. Devemos utilizar todos? Como escolher o melhor para cada tipo de aplicação/problema?