UniCesumar
EDUCAÇÃO PRESENCIAL E A DISTÂNCIA

ENGENHARIA DE SOFTWARE

Aula 8

Tópicos da aula

- Introdução à UML;
- Ferramentas CASE.

Introdução

Introdução

UML é **uma linguagem padrão para projetar sistemas de software**, utilizada na visualização, especificação, construção e documentação de artefatos em sistemas complexos (BOOCH, 2005, p. 13).

Introdução

A UML é uma linguagem padronizada que fornece um conjunto de notações para descrever sistemas. Com diagramação visual, ela facilita a comunicação entre desenvolvedores e partes interessadas. Vamos desvendar suas componentes e aplicações na prática.

Introdução

❖ UML **não** é uma linguagem de programação.

✔ UML **é** uma linguagem de modelagem.

UML auxilia engenheiros de software **na definição das características do sistema**, incluindo requisitos, comportamento, estrutura lógica, dinâmica dos processos e necessidades físicas do equipamento onde o software será implantado.

Introdução

A importância da UML reside em sua capacidade de simplificar a complexidade do desenvolvimento de software. Ela ajuda na identificação de requisitos, na análise de sistemas e na documentação de projetos, promovendo uma melhor compreensão entre equipes.

UML

- **Rational Software:** Empresa que desenvolvia ferramentas para engenharia de software, adquirida pela IBM em 2003.

- **OMG** (Object Management Group): **Consórcio internacional que define padrões industriais** para diversos setores, incluindo sistemas financeiros, saúde, armazenamento de dados, blockchain, IoT, cibersegurança e astronomia.

- **Diagrama:** Representação gráfica, geralmente parcial, de um modelo de sistema ou de um processo específico.

UML

- **Desenvolvida** por Grady Booch, Ivar Jacobson e James Rumbaugh na Rational Software **entre 1994 e 1995;**
- A UML foi **adotada como padrão** pela OMG **em 1997** e tornou-se um **padrão ISO em 2005**;
- A notação UML utiliza **símbolos gráficos** com semântica bem definida, **permitindo a elaboração modelos**.

UML

- Não depende do domínio de aplicação (Área de conhecimento);
- Não está vinculado a um processo ou metodologia de desenvolvimento específico;
- Não está limitado às ferramentas de modelagem utilizadas.

UML

- **A UML facilita a clareza do escopo ao centralizar um conceito em um diagrama**, usando uma linguagem que os envolvidos no projeto podem entender facilmente;
- **Ajuda é valiosa quando usada na medida certa**, ou seja, apenas quando realmente necessário.
- O **principal desafio na produção de software**, em qualquer parte do mundo, **é a má comunicação**;

UML – Eliminando Ruídos

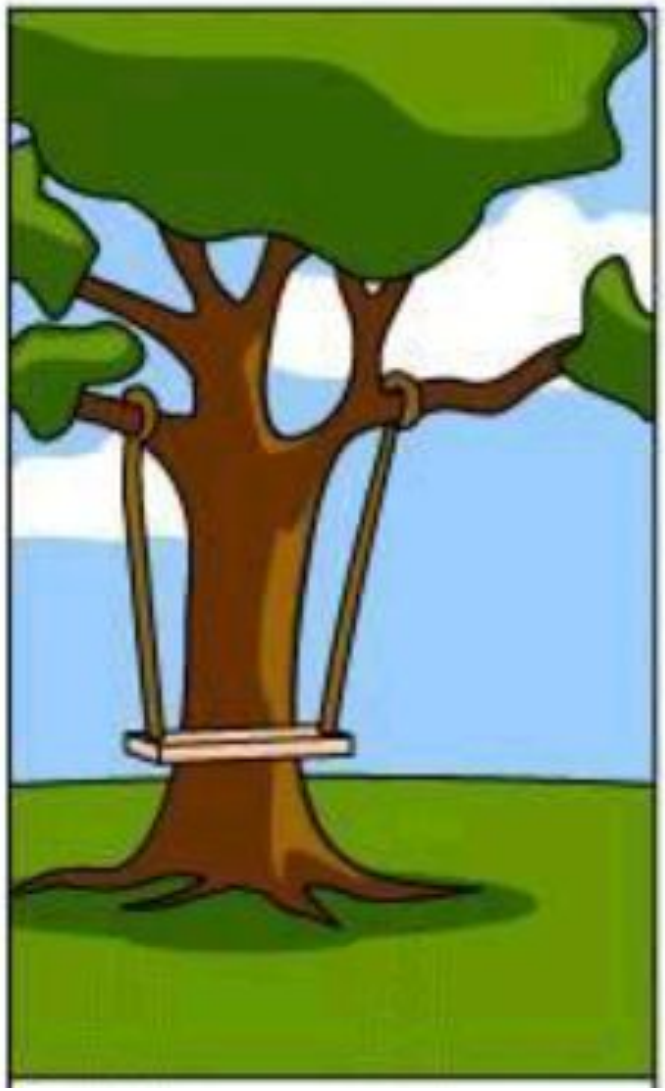
João quer um **BALANÇO** na árvore!

UML – Eliminando Ruídos

João explica como será o **BALANÇO** que deseja para a equipe.

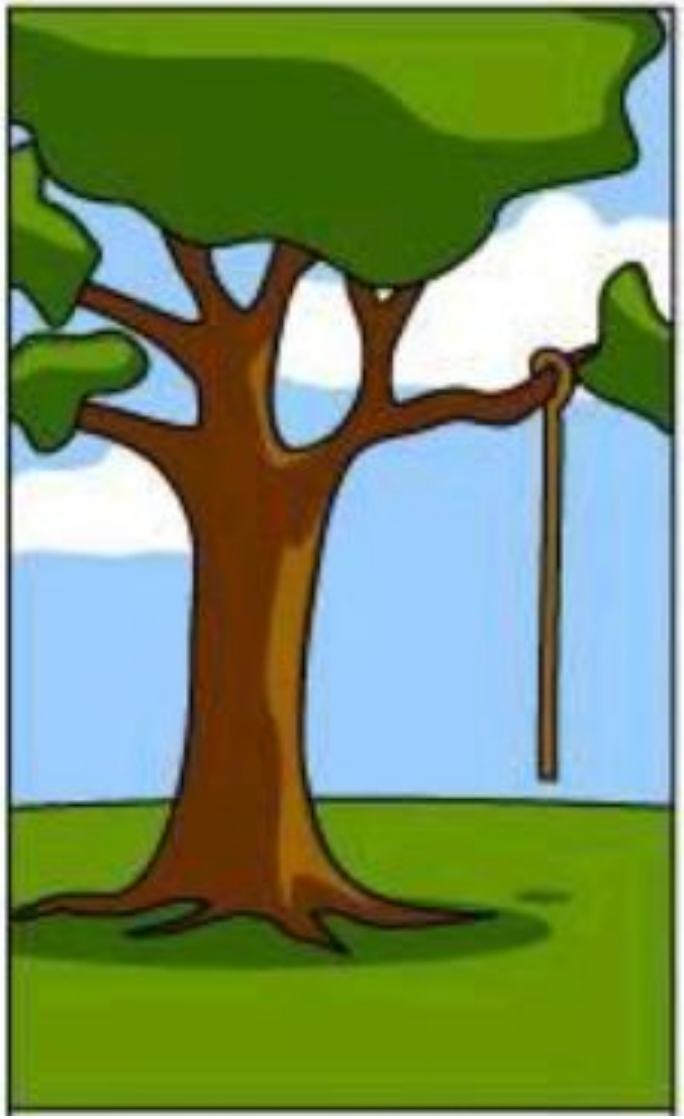
UML – Eliminando Ruídos

Na equipe Marcos interpreta o que João deseja como **BALANÇO** e explica a José como construir o **BALANÇO**



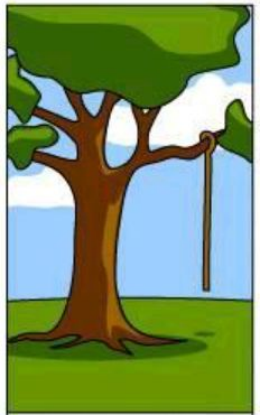
UML – Eliminando Ruídos

José, por sua vez, executa o **BALANÇO** conforme entendeu as explicações de Marcos.



UML – Eliminando Ruídos

Marcos verifica o que José construiu, e diz a ele que não era isso, e que José precisa corrigir



UML – Eliminando Ruídos

José corrige o **BALANÇO** conforme as novas explicações de Marcos e realiza a entrega.

UML – Eliminando Ruídos

João queria um **BALANÇO**.

UML

A UML atua como uma linguagem universal para profissionais de software, funcionando como **um "Google Translate"** que facilita a comunicação clara e objetiva **entre todos os envolvidos no processo de desenvolvimento**, como analistas de negócios, product owners, scrum masters, arquitetos, desenvolvedores, gerentes de projeto e demais partes interessadas.

UML

Alguns **profissionais de diferentes níveis de experiência acreditam que usar UML** em equipes de software **é perda de tempo**.

- **Confundir o uso da UML** para **comunicação** com seu uso para **documentação** é um erro comum.

- **Profissionais jovens** que entram no mercado na era do "agilismo" frequentemente **aceitam ideias** de terceiros **sem questionar** os fundamentos.

UML

- Atualmente, há **14 diagramas**;
- Cada um **formados por elementos gráficos inter-relacionados**;
- Na UML, **os diagramas** são classificados em **dois** grandes **grupos**:

  Diagramas estruturais (estáticos)

  Diagramas comportamentais (dinâmicos).

UML - Diagramas Estruturais

Os diagramas estruturais da UML, como o diagrama de classes e o diagrama de componentes, são fundamentais para descrever a estrutura do sistema. Eles permitem uma visão clara das relações e interações entre os elementos do software.

UML - Diagramas Comportamentais

Os diagramas comportamentais, como o diagrama de casos de uso e o diagrama de sequência, ajudam a capturar o comportamento do sistema. Eles são essenciais para entender como os usuários interagem com o software e como os processos se desenrolam.

UML

Diagramas Estruturais

- Classes;
- Objetos;
- Pacotes;
- Componentes;
- Implantação;
- Estrutura Composta;
- Perfil.

UML

Diagramas Comportamentais

- Caso de Uso;
- Sequência;
- Comunicação;
- Máquina de Estados;
- Atividade;
- Visão Geral de Interação;
- Temporização.

UML

O objetivo de um diagrama UML **é comunicar informações de forma padronizada**, garantindo que todos os receptores compreendam o padrão utilizado.

É o famoso: “entendeu? Ou quer que desenha?”

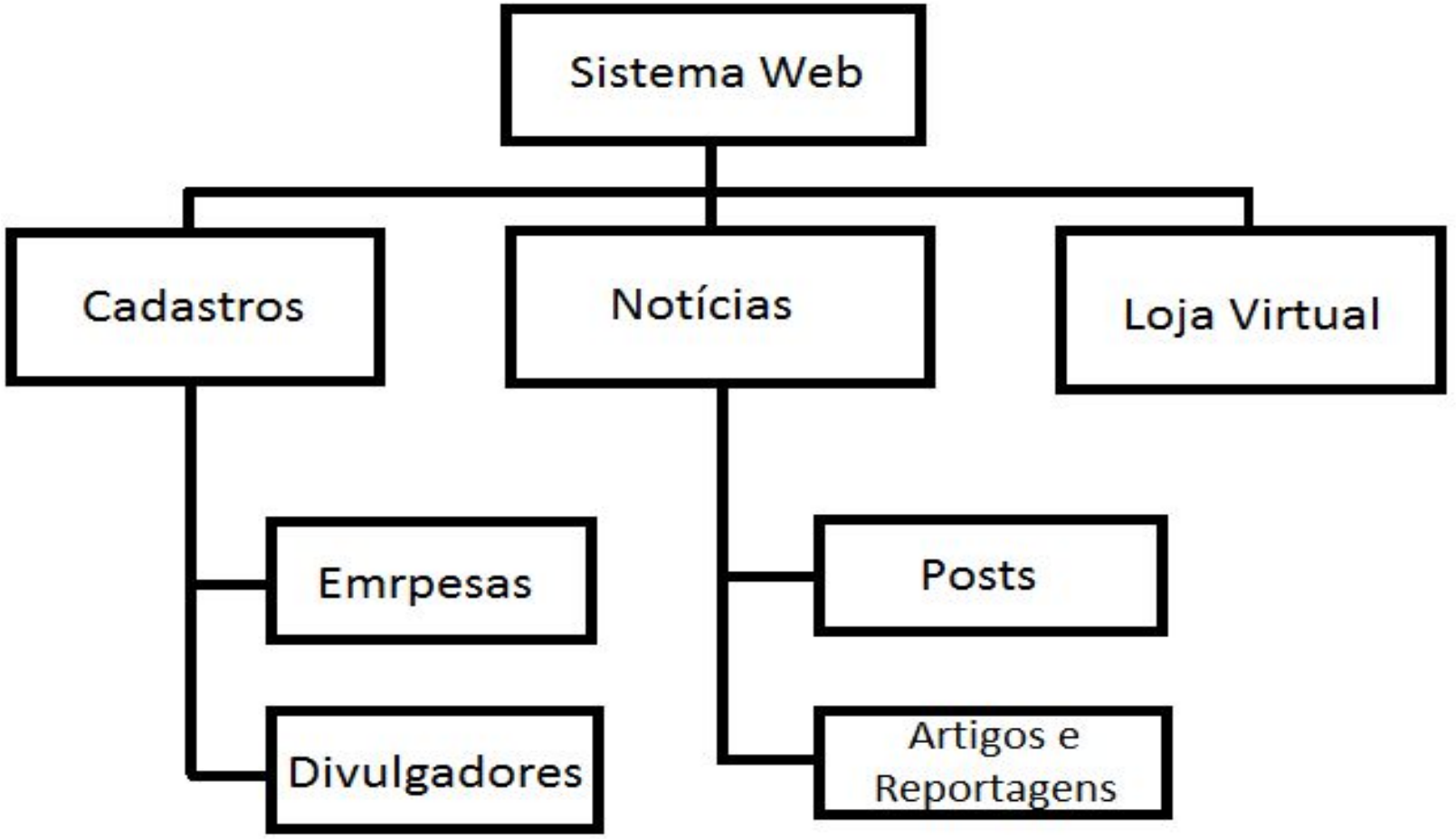
UML (resumindo)

Imagine **uma sala sem internet ou telefone**, onde estão presentes um **chinês**, um **francês** e um **brasileiro**, todos falando **apenas** seu **idioma nativo**.
Com **apenas papel e lápis à disposição**, o francês deseja solicitar um café.

Qual seria a **maneira mais eficiente**, dado o cenário e os recursos disponíveis, **para que o francês comunique "quero um café"**?

UML (resumindo)

Exemplos de Modelagem



Documentação com UML

A documentação é uma parte crucial do desenvolvimento de software. A UML fornece uma maneira estruturada de documentar requisitos, design e arquitetura, facilitando a manutenção e a evolução do sistema ao longo do tempo.

Desafios da UML

Apesar de suas vantagens, a UML enfrenta alguns desafios. A complexidade dos diagramas pode ser intimidante para iniciantes, e a padronização pode levar a interpretações variadas. Vamos discutir como superar essas barreiras.

Exercícios

Tomando como base o processo de desenvolvimento de software, em que momento devemos realizar a modelagem do sistema?

Exercícios

Tomando como base o processo de desenvolvimento de software, em que momento devemos realizar a modelagem do sistema?

Os modelos **devem ser gerados após a aquisição dos requisitos e com base nos mesmos**. Ainda, devem ser elaborados antes do início da implementação do sistema.

Exercícios

O que é um modelo gerado no processo de criação de um software?

Exercícios

O que é um modelo gerado no processo de criação de um software?

Os modelos gerados no processo de criação de um software é **uma representação abstrata e simplificada do sistema**. **Descreve aspectos do software**, como sua estrutura, comportamento e interações, para ajudar na compreensão, planejamento e desenvolvimento do projeto.

Ferramentas Case

- Ferramentas **CASE** (Computer-Aided Software Engineering – **Engenharia de Software Auxiliada por Computador**);

- **Software que auxilia em atividades do processo de desenvolvimento de software**, incluindo engenharia de requisitos, projeto, programação e testes.

- **Podem incluir editores de projeto**, dicionários de dados, compiladores, depuradores, ferramentas de construção de sistemas e entre outros.

Ferramentas Case

Exemplos de atividades que podem ser automatizadas utilizando CASE

- A criação de modelos gráficos de sistemas como parte das especificações de requisitos ou do projeto de software.

- Compreende-se um projeto através de um dicionário de dados, que fornece informações sobre entidades e suas relações no sistema.

Ferramentas Case

- Criação de interfaces de usuário a partir de uma descrição gráfica interativa fornecida pelo próprio usuário.
- A depuração de programas pelo fornecimento de informações sobre um programa em execução.
- A tradução automatizada de programas de uma linguagem de programação antiga, como Cobol \ Delphi, para uma versão atual.

Ferramentas Case - Classificação

Upper CASE ou U-CASE ou Front-End: Voltadas para as fases iniciais do desenvolvimento, como análise de requisitos, design lógico e documentação

Exemplos:

- **Análise de Requisitos:** Rational RequisitePro, IBM Engineering Requirements Management;
- **Projeto Lógico:** Enterprise Architect, Visual Paradigm;
- **Documentação:** Microsoft Word, Confluence;

Ferramentas Case - Classificação

O Lower CASE ou L-CASE ou Back-End: Suporte nas fases finais do desenvolvimento, como codificação, testes e manutenção.

Exemplos:

- **Codificação:** Visual Studio Code, Visual Studio, Eclipse
- **Testes:** JUnit, Postman, Selenium, TestComplete
- **Manutenção:** Git, Jenkins

Ferramentas Case - Classificação

Integrated CASE ou I-CASE: Combina as funcionalidades de Upper e Lower CASE, além de oferecer recursos adicionais como controle de versão.

Exemplos:

- Enterprise Architect
- IBM Rational Rose
- MagicDraw

Ferramentas Case - Classificação

Best in Class ou Kit de Ferramenta: Como as ferramentas I-CASE, os Kits de Ferramenta também têm a capacidade de integrar e complementar outras ferramentas externas.

Exemplos:

- **JIRA:** Integra com ferramentas de desenvolvimento e gestão de projetos.
- **GitHub:** Complementa com ferramentas de CI/CD e monitoramento.
- **Slack:** Conecta-se a diversas ferramentas de produtividade e gerenciamento.

Ferramentas Case – Outros Exemplos

- IBM – Rational Rose
- Astah http://astah.net/student-license-request
- ArgoUML
- Microsoft Virtual Modeler
- Visual Paradigm.
- **Lucidchart**
- Draw.io

Quem se beneficia dos diagramas UML?

- Proprietário do Produto;
- Analista de Negócios;
- Analista de Sistemas;
- Operadores do Sistema;
- Desenvolvedor / Programador;
- Gerentes de Qualidade.

UML – Quando Usar?

- É essencial **definir claramente o desejo do cliente** que será traduzido no software.
- Quando membros de uma **equipe precisam ter uma visão única** e padronizada sobre um determinado aspecto;
- **Comunicar** ao mundo externo os protocolos (contratos) das **interfaces do sistema** que devem ser consumidas por terceiros.
- Ilustrar as topologias arquitetônicas físicas e lógicas.

https://www.lucidchart.com/blog/pt/tipos-de-diagramas-UML

Dúvidas ou perguntas?

UniCesumar
EDUCAÇÃO PRESENCIAL E A DISTÂNCIA