Aula 03 - Engenharia de Software II

Prof. Me. Deivison S. Takatu

deivison.takatu@fatec.sp.gov.br

Sumário

- Por que modelar sistemas?
- UML (Unified Modeling Language) ;
- Diagramas Estruturais;
- Diagramas Comportamentais;
- Ferramentas Colaborativas;
- Atividade.

Agenda - Aulas de Reposição

Data	Horário	Conteúdo
04/10/2025	07:40 - 13:00	SCRUM
01/11/2025	13:00 - 18:20	Product Management
22/11/2025	07:40 - 13:00	Engenharia / Gamificação

Por que modelar sistemas?

- Comunicação entre equipes - Facilita o entendimento comum entre desenvolvedores, analistas e stakeholders.
- Redução de ambiguidades - Torna requisitos e design mais claros e precisos.
- Visualização da estrutura - Permite enxergar o sistema antes de construí-lo.

Referência: LARMAN (2007).

UML (Unified Modeling Language)

- É uma linguagem de modelagem padronizada para visualizar, especificar, construir e documentar sistemas de software.
- Não é uma linguagem de programação, mas sim uma forma de representar graficamente a estrutura e o comportamento de sistemas.
- Ajuda a comunicar ideias entre stakeholders.

Objetivos da UML

Fornecer uma linguagem comum - Estabelecer uma notação padrão que possa ser compreendida por todos os profissionais envolvidos no desenvolvimento.

Padronizar a modelagem de software - Unificar as melhores práticas de modelagem em uma abordagem consistente.

Apoiar todas as fases do desenvolvimento - Oferecer ferramentas de modelagem que sejam úteis desde a concepção até a implementação e manutenção do software.

Características principais do UML

- **Visual** - Utiliza representações gráficas para facilitar a compreensão de sistemas complexos
- **Padronizada** - Adotada como padrão internacional pela OMG e ISO/IEC
- **Extensível** - Permite a criação de perfis e estereótipos para adaptar-se a domínios específicos
- **Independente de metodologia** - Pode ser utilizada com diferentes processos de desenvolvimento
- **Abrangente** - Usada em análise, design e documentação de sistemas de software

Diagramas Estruturais – UML

1. **Diagrama de Classes**: mostra atributos, métodos e relacionamentos das classes.
2. **Diagrama de Objetos**: representa instâncias concretas de classes em um momento específico.
3. **Diagrama de Perfil**: permite personalizar UML com estereótipos e restrições.
4. **Diagrama de Pacotes**: organiza classes e elementos em grupos lógicos.
5. **Diagrama de Componentes**: ilustra a estrutura física do sistema em módulos reutilizáveis.
6. **Diagrama de Implantação**: descreve a distribuição física de software e hardware.

Diagramas Comportamentais – UML

1. **Diagrama de Casos de Uso**: mostra interações entre atores e o sistema.
2. **Diagrama de Atividades:** representa fluxos de trabalho e decisões dentro de processos.
3. **Diagrama de Sequência:** descreve a troca de mensagens entre objetos ao longo do tempo.
4. **Diagrama de Comunicação:** destaca as colaborações e vínculos entre objetos.
5. **Diagrama de Máquina de Estados:** modela os estados e transições de um objeto durante seu ciclo de vida.

Modelagem com Ferramentas Colaborativas

Durante a modelagem de um sistema, é comum utilizar ferramentas que permitem que equipes trabalhem juntas em tempo real, mesmo à distância.

Facilitam a criação, edição e compartilhamento de diagramas UML e outros modelos, além de promover padronização e reduzem erros de comunicação.

Modelagem Colaborativa com Miro

Plataforma online de colaboração visual em tempo real. Permite criar e editar diagramas UML, fluxogramas e mapas mentais de forma intuitiva.

Oferece templates prontos para acelerar o processo de modelagem. Suporta edição simultânea, comentários e votação — facilitando o alinhamento da equipe.

Fonte: Miro.

Modelagem Colaborativa com Mermaid

Plataforma baseada em texto para criação de diagramas. Utiliza uma sintaxe simples em linguagem de marcação, permitindo gerar diagramas UML, fluxogramas, organogramas, gráficos de Gantt e muito mais.

Ideal para equipes de desenvolvimento que desejam manter a documentação próxima ao código, garantindo rastreabilidade e colaboração.

Fonte: Mermaid.

Atividade

Nesta atividade, cada grupo deverá selecionar cinco dos requisitos ou User Stories definidos na atividade anterior e realizar a modelagem colaborativa utilizando as ferramentas Miro e Mermaid. No Miro, construam diagramas visuais que representem o requisito escolhido. No Mermaid, representem o mesmo requisito em formato textual, gerando diagramas por meio da sintaxe em linguagem de marcação.

Referências

1. PRESSMAN, Roger S.; MAXIM, Bruce R. Engenharia de Software: uma abordagem profissional. 8. ed. McGraw-Hill, 2016.
2. SOMMERVILLE, Ian. Engenharia de Software. 10. ed. Pearson, 2019.
3. LARMAN, Craig. Utilizando UML e Padrões: Uma introdução à análise e ao projeto orientados a objetos. 3. ed. Bookman, 2007.
4. PFLEEGER, Shari Lawrence; ATLEE, Joanne M. Engenharia de Software: teoria e prática. 4. ed. Pearson, 2010.
5. JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, James. UML: Guia do Usuário. 2. ed. Bookman, 2000.
6. KOTONYA, Gerald; SOMMERVILLE, Ian. Engenharia de Requisitos. 1. ed. LTC, 1998.
7. IEEE Computer Society. Guide to the Software Engineering Body of Knowledge (SWEBOK V3.0). IEEE, 2014.
8. ISO/IEC/IEEE. 29148:2018 – Systems and software engineering — Life cycle processes — Requirements engineering. ISO/IEC/IEEE, 2018.
9. Object Management Group (OMG). Unified Modeling Language (UML) Specification. Versão 2.5.1, 2017.

Aula 03 - Engenharia de Software II

Prof. Me. Deivison S. Takatu

deivison.takatu@fatec.sp.gov.br