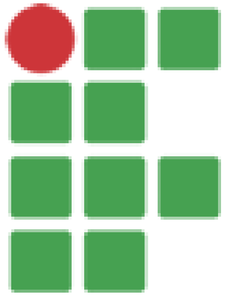


INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

Disciplina

Modelagem de Domínio e Conceitos de Orientação a Objetos

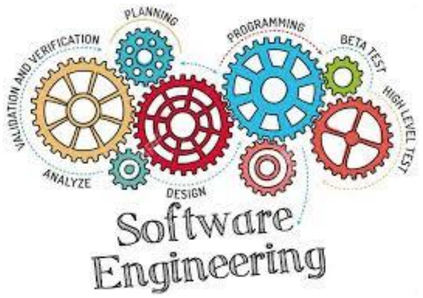
PROF. FERNANDO TAMBERLINI ALVES

Revisão

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

- **Engenharia de Software**: processos, métodos e ferramentas de apoio que nos auxiliam no desenvolvimento de software;
- **Arquitetura de Software:** características de como o software será desenvolvido;
- **Modelagem de Domínio:** identificação do contexto e levantamento dos requisitos do software que será desenvolvido;
- **Metodologia de Desenvolvimento:** etapas específicas a serem percorridas durante o desenvolvimento de um software;
- **Tipos de Dados em JavaScript:** linguagem dinamicamente e fracamente tipada.

Engenharia de Software

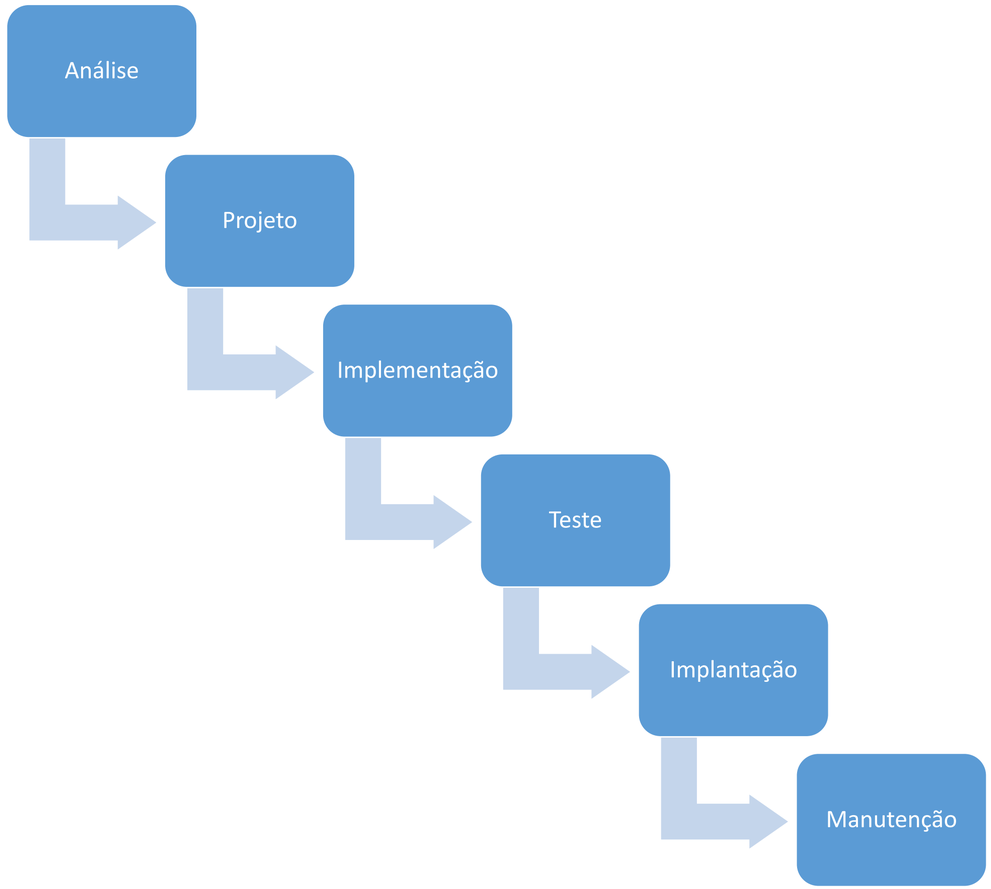


- **Processo:** refere-se ao conjunto de ações sequenciais para atingir um resultado. Envolve várias etapas interdependentes que precisam ser seguidas para chegar a um objetivo final.É mais amplo que o método, pois abrange o contexto, as etapas e a organização das atividades.

- **Método:** refere-se ao como algo é feito. É uma abordagem estruturada para realizar uma tarefa ou alcançar um objetivo. Envolve técnicas, passos ou práticas específicas, definindo a forma de execução

- **Ferramenta:** é um programa ou aplicação projetada para auxiliar a equipe responsável no desenvolvimento, manutenção e gerenciamento do software;

Processo



- **Análise (planejamento):** definição dos objetivos, escopo e recursos necessários. Identificação e das funcionalidades e restrições do software;
- **Projeto (design) :** definição da arquitetura adequada e design do sistema
- **Implementação (codificação):** codificação e construção do software conforme projeto;
- **Teste**: verificação e validação do software para garantir qualidade e correção.
- **Implantação**: Lançamento do software no ambiente de produção.
- **Manutenção**: Atualização, correção de erros e melhorias contínuas após a implantação.

Ferramentas

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

- **Ferramentas de Desenvolvimento IDE** (Ambiente de Desenvolvimento Integrado) como Visual Studio que oferecem editores de código, depuradores e outras funcionalidades integradas.

- **Compiladores e Interpretadores** que traduzem o código-fonte para um formato executável pelo computador.

- **Controle de Versão** como as ferramentas Git, SVN e Mercurial ajudam a gerenciar e acompanhar as mudanças no código ao longo do tempo, facilitando a colaboração e a reversão de alterações.

- **Ferramentas de Modelagem e Design**, como Lucidchart e Draw.io, que permitem a criação de diagramas e modelos que representam a arquitetura e o design do sistema.

Modelagem de Domínio

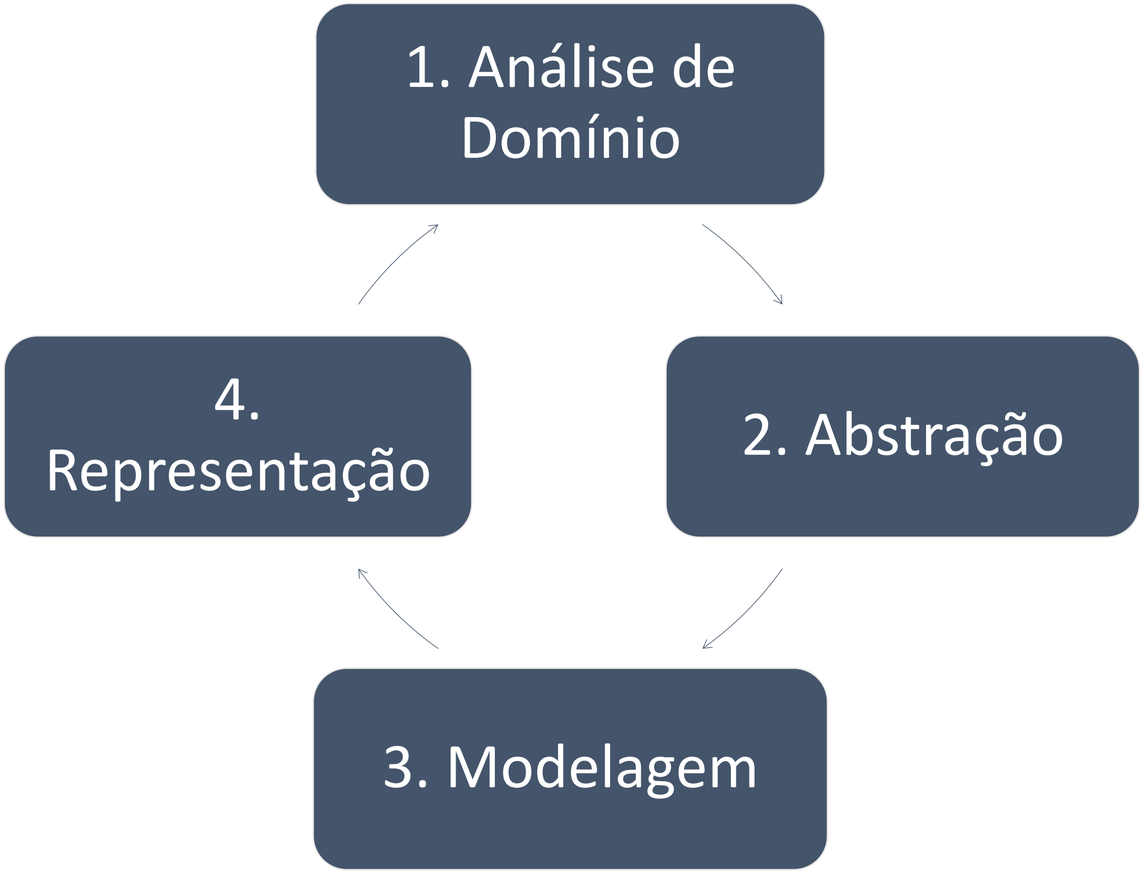
INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

1. Identificar o contexto do Sistema de Informação (Dados, Processo e Atores Envolvidos);

2. Idealizar como funcionará o Sistema de Informação pretendido;

3. Modelar a solução idealizada;

4. Representar o modelo idealizado numa linguagem clara e legível.



Javascript - Tipo de Dados

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

JavaScript é uma linguagem de programação que é fracamente e dinamicamente tipada, o que significa que o tipo de dados de uma variável é determinado em tempo de execução e sem a necessidade de declarar o seu tipo.

Em JavaScript, os tipos de dados podem ser divididos em duas categorias principais: primitivos e não primitivos (também chamados de objetos).

Primitive Data Types	Reference Data Types
• Boolean • Number • BigInt • String • Undefined • Null • Symbol	• Array • Object • RegEx • Date • ....

Programação Orientada a Objetos

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

A Orientação a Objetos (OO) é um paradigma de programação que se baseia na ideia de organizar o código em torno de objetos, que são instâncias de classes. É uma abordagem poderosa e amplamente utilizada no desenvolvimento de software, pois ajuda a criar sistemas **mais organizados, modulares e reutilizáveis**.

A Orientação a Objetos promove uma abordagem mais natural e organizada para modelar o mundo real em código, tornando-o mais legível, manutenível e flexível. É amplamente utilizada em linguagens de programação como Java, C++, Python, C#, entre outras, e é aplicável a uma variedade de domínios de desenvolvimento de software, desde aplicações desktop até sistemas distribuídos e aplicativos web.

Programação Orientada a Objetos

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

Aqui estão os principais conceitos do paradigma de Orientação a Objetos:

Objeto: Um objeto é uma instância concreta de uma classe. Ele representa uma entidade do mundo real e possui atributos (propriedades) que descrevem seu estado e métodos que definem seu comportamento. Por exemplo, um objeto "Carro" pode ter atributos como "cor" e "velocidade" e métodos como "ligar" e "desligar".

Classe: Uma classe é um modelo ou uma definição abstrata que descreve as características comuns a um conjunto de objetos. Ela serve como um plano para a criação de objetos. A classe define os atributos e métodos que seus objetos terão em comum. Por exemplo, uma classe "Carro" define os atributos e métodos que todos os carros terão

Como representar um Software?

A Linguagem de Modelagem Unificada, ou **UML (Unified Modeling Language),** foi criada através da colaboração de um grupo de pessoas influentes na área de engenharia de software nos anos 90.

Nos anos 90, a engenharia de software estava passando por uma fase de transição. As metodologias tradicionais estavam sendo questionadas e uma necessidade crescente de uma linguagem padrão para modelagem de sistemas de software estava surgindo. Havia várias metodologias de modelagem em uso, como Booch, OMT (Object Modeling Technique) e OOSE (Object-Oriented Software Engineering), cada uma com suas próprias notações e abordagens.

Como representar um Software?

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

A UML foi inicialmente desenvolvida como uma fusão de três metodologias principais:

Booch Method:

- Criado por Grady Booch, era uma das metodologias mais antigas e amplamente adotadas para modelagem de software.
- Focava em diagramas de classes, diagramas de estado, diagramas de sequência, entre outros.

Object Modeling Technique (OMT):

- Desenvolvido por James Rumbaugh, era outra abordagem popular para modelagem de sistemas orientados a objetos.
- Tinha um foco especial em diagramas de objetos, diagramas de estados, entre outros.

Object-Oriented Software Engineering (OOSE):

- Desenvolvido por Ivar Jacobson, foi outra metodologia influente com forte ênfase na análise e design orientados a objetos.
- Introduziu conceitos como casos de uso.

Como representar um Software?

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

Unificação, Desenvolvimento e Evolução

Em 1995, Grady Booch, James Rumbaugh e Ivar Jacobson uniram forças para criar uma linguagem de modelagem padrão, unificando o melhor de suas respectivas metodologias. Eles foram apoiados por outras empresas e especialistas em engenharia de software, formando o "Three Amigos" (Três Amigos).

O objetivo era criar uma linguagem que pudesse abranger todos os aspectos do processo de desenvolvimento de software, desde a concepção até a implementação. A UML foi destinada a ser uma linguagem gráfica para visualizar, especificar, construir e documentar os artefatos de um sistema de software.

O trabalho inicial resultou na primeira versão da UML, conhecida como UML 0.8, em 1996. Ela foi seguida pela UML 0.9 e, finalmente, pela UML 1.0 em 1997, que foi amplamente adotada pela indústria de software. Desde então, a UML passou por várias revisões e atualizações. A mais recente versão 2.5 com uma ampla gama de diagramas e elementos para modelagem de sistemas complexos.

Como representar um Software?

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

Contribuições e Aceitação

A UML se tornou a linguagem padrão para modelagem de software. Sua aceitação foi impulsionada por uma série de fatores:

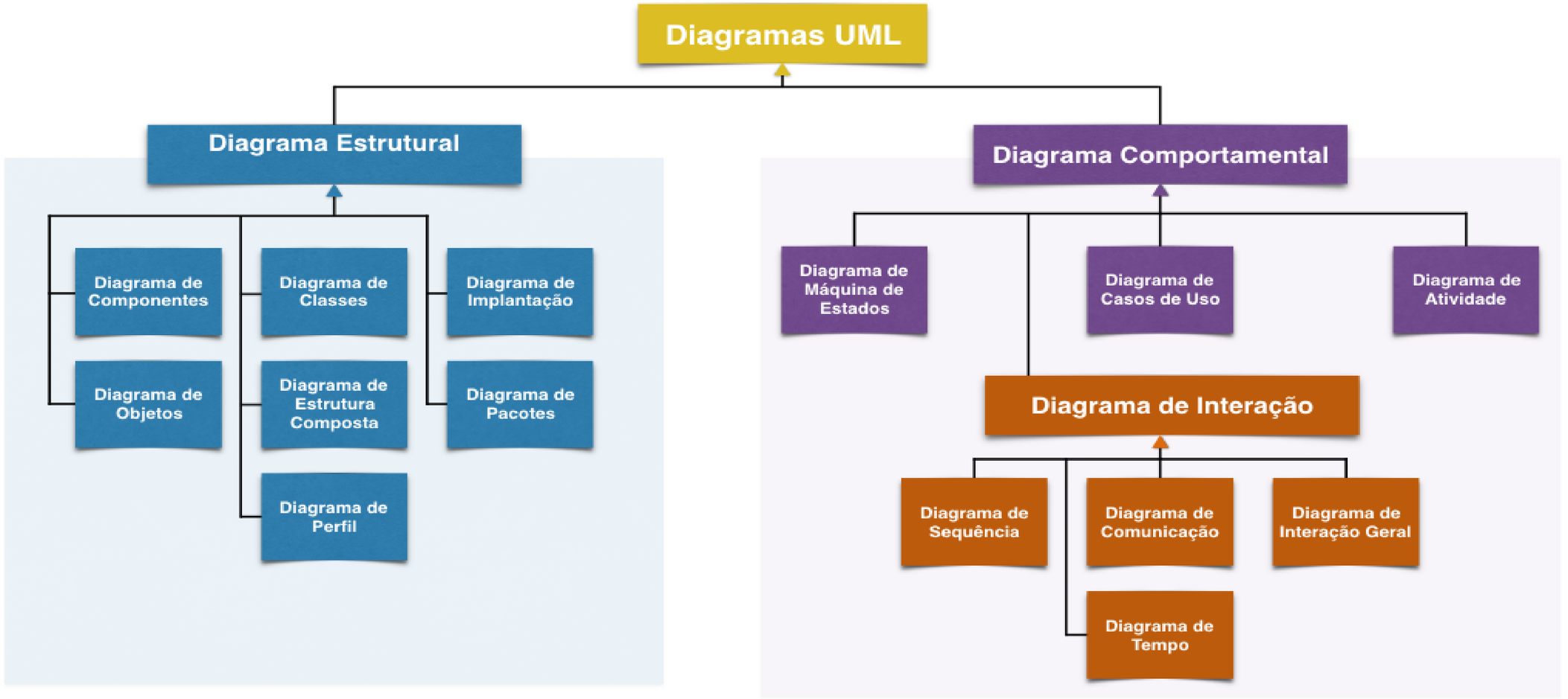
Padronização: Tornou-se um padrão da indústria, facilitando a comunicação entre equipes de desenvolvimento e organizações.

Flexibilidade: Oferece uma variedade de diagramas para atender às diferentes fases e aspectos do desenvolvimento de software.

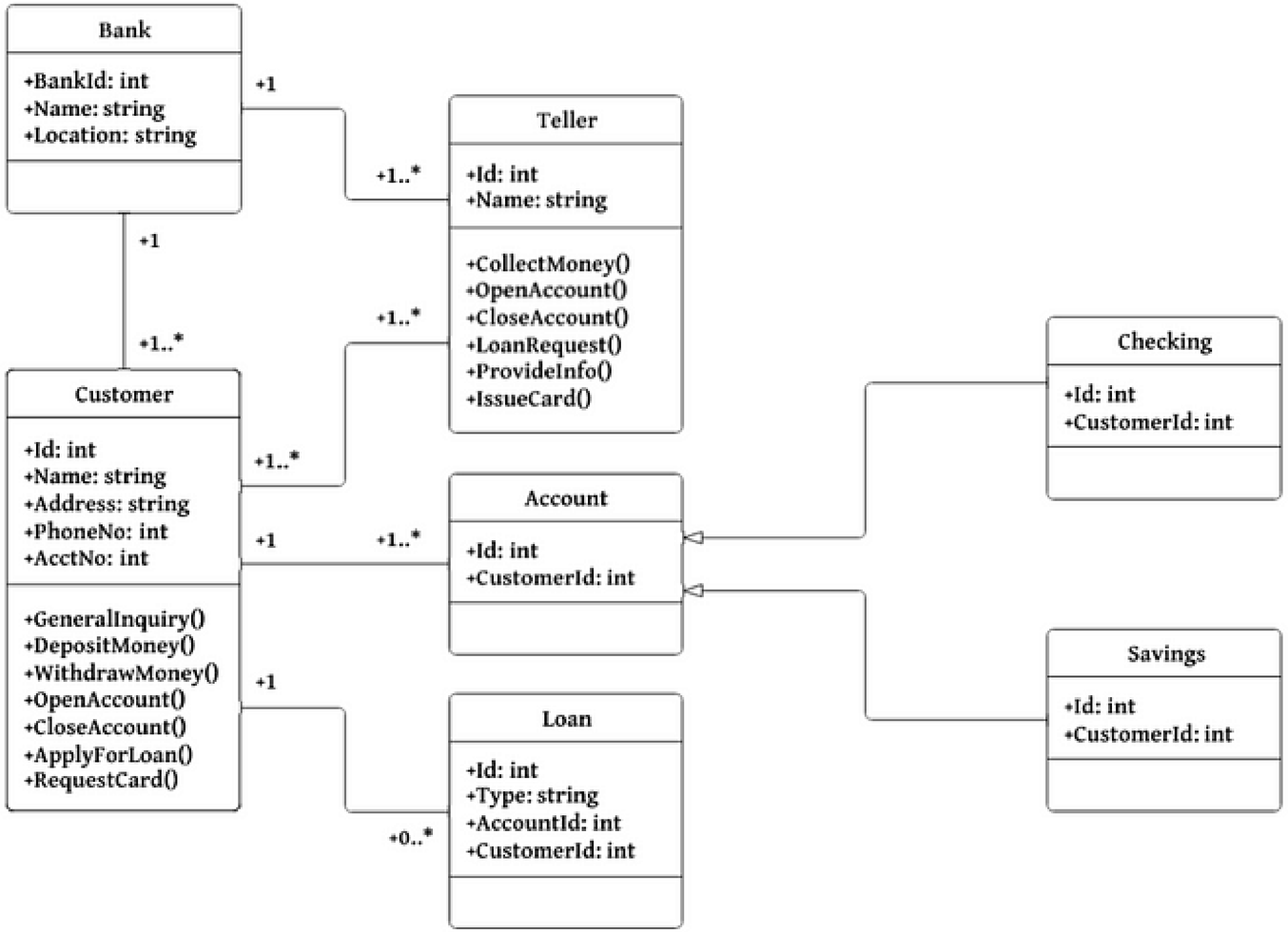
Ampla Comunidade: Conta com uma comunidade ativa de desenvolvedores, profissionais de engenharia de software e pesquisadores, que contribuem para seu desenvolvimento contínuo.

Ferramentas de Suporte: Há uma ampla variedade de ferramentas de modelagem UML disponíveis, tanto comerciais quanto de código aberto.

Diagramas UML



Diagramas de Classe

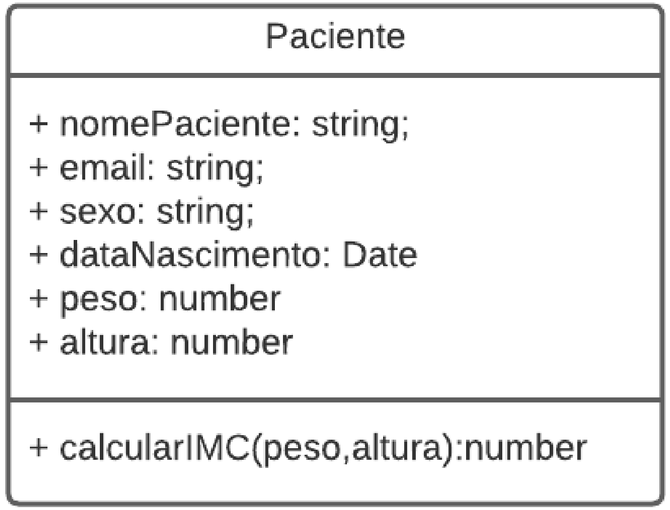


Representando o 1º Objeto

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

- Representando um objeto a partir de um Diagrama de Classe

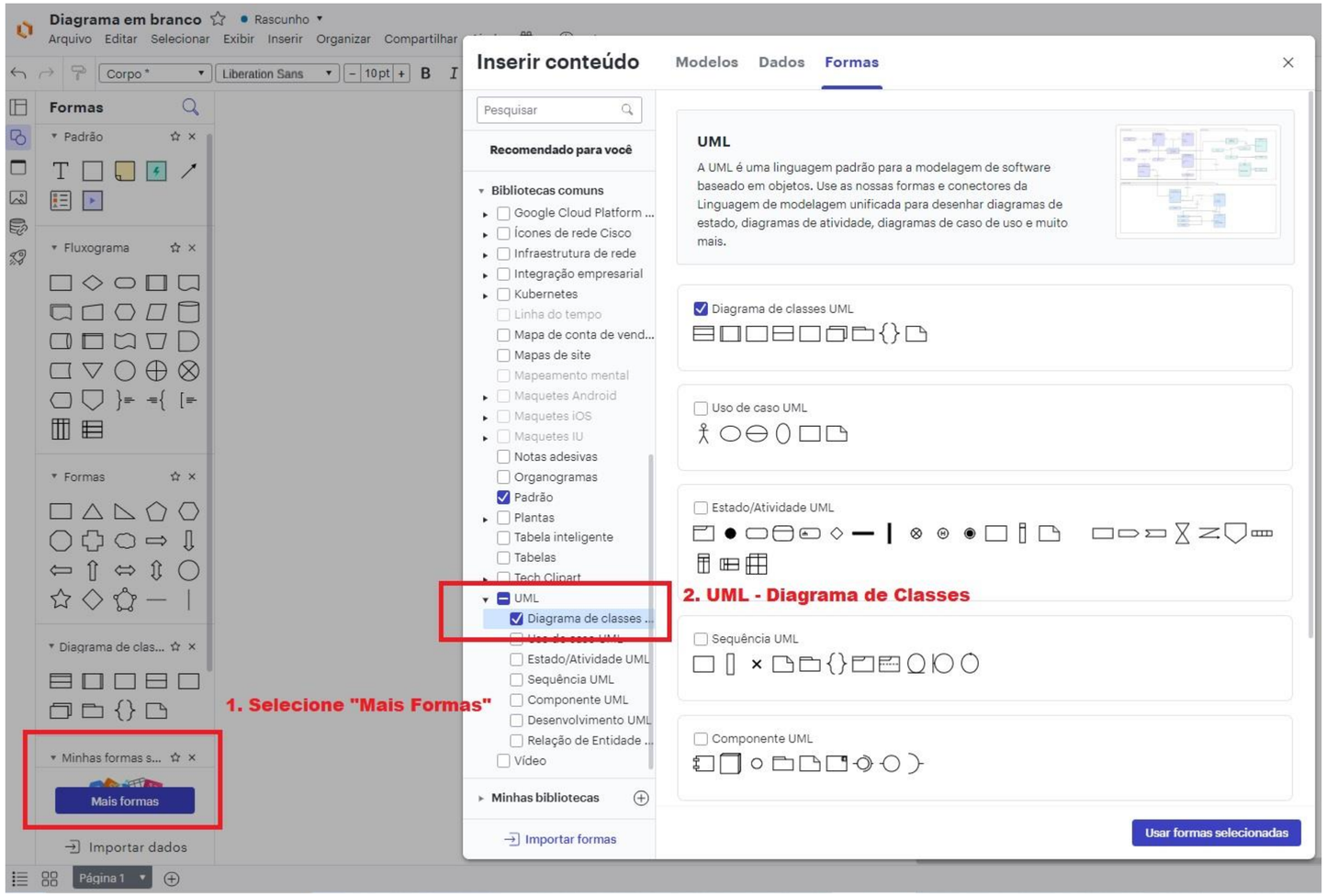
- Vamos utilizar a ferramenta LucidChart ou Diagrams (Draw.io)



Representando o 1º Objeto

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

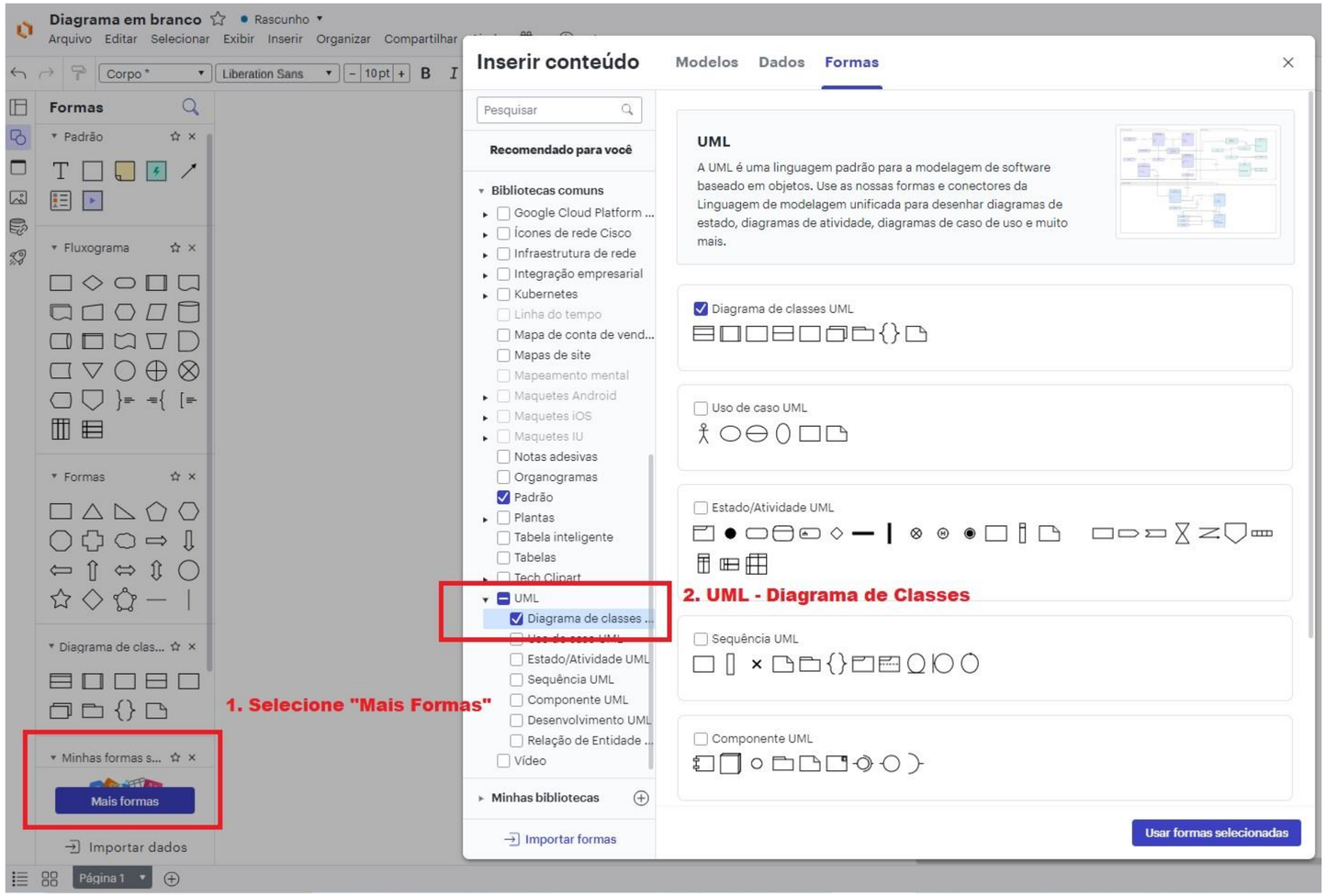
- Acesse o link do aplicativo LucidChart disponível no site do professor
- Inclua o UML – Diagrama de Classes



Representando o 1º Objeto

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

- Acesse o link do aplicativo LucidChart disponível no site do professor
- Inclua o UML – Diagrama de Classes



Atividade Prática

1. Escolha uma ferramenta de Edição de Código (VSCODE ou GITHUB);

2. Crie ou edite o arquivo index.html. O código deste arquivo deverá ser igual ao código da aula03 disponível no site do professor.

3. Crie um arquivo imc.js. O código deste arquivo deverá ser igual ao código da imc.js disponível no site do professor.

4. Teste este aplicativo criado e encontrado.

5. Vamos executar este aplicativo localmente ou no GitHub

Dúvidas?

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Rio de Janeiro

Dúvidas?

fernando.alves@ifrj.edu.br

ftamberlini.dev.br