**Iniciando com UML**

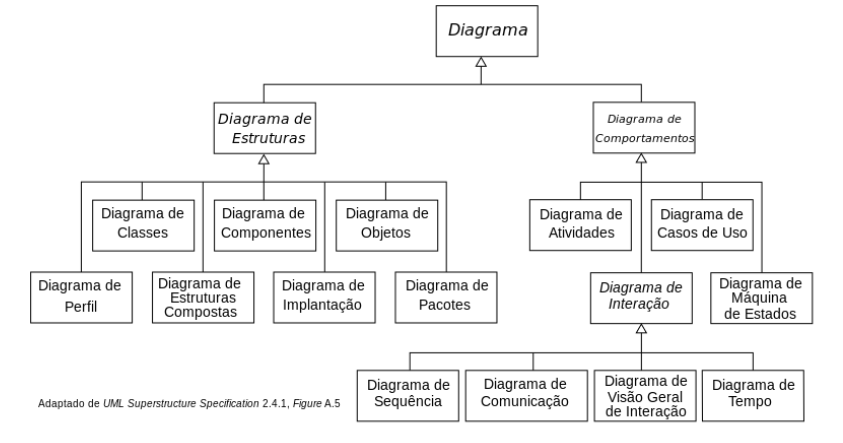
* **O que é a Linguagem UML?**

[UML](http://www.omg.org/spec/UML/), abreviação de Unified Modeling Language, é uma linguagem de modelagem padronizada que consiste em um conjunto integrado de diagramas, desenvolvido para ajudar os desenvolvedores de sistemas e software a especificar, visualizar, construir e documentar os artefatos de sistemas de software, bem como modelagem de negócios e outros. A UML representa uma coleção das melhores práticas de engenharia de software que se mostraram bem-sucedidas na modelagem de sistemas grandes e complexos. A UML é uma parte muito importante do desenvolvimento de software orientado a objetos e do processo de desenvolvimento de software. A UML usa principalmente notações gráficas para expressar o design de projetos de software. O uso da UML ajuda as equipes de projeto a se comunicar, explorar possíveis projetos e validar o design arquitetônico do software.

* **Quais são os diagramas de UML?**

*Existem três classificações de diagramas UML:*

* **Diagramas de comportamento:**Um tipo de diagrama que descreve recursos comportamentais de um sistema ou processo de negócios. Isso inclui atividade, máquina de estado e diagramas de casos de uso, bem como os quatro diagramas de interação.
* **Diagramas de interação:**Um subconjunto de diagramas de comportamento que enfatizam as interações com objetos. Isso inclui comunicação, visão geral da interação, sequência e diagramas de tempo.
* **Diagramas de estrutura:** Um tipo de diagrama que descreve os elementos de uma especificação independentemente do tempo. Isso inclui diagramas de classe, estrutura composta, componente, implantação, objeto e pacote.





**Diagramas estruturais** devem ser utilizados para especificar detalhes da estrutura do sistema (parte estática), por exemplo: classes, métodos, interfaces, namespaces, serviços, como componentes devem ser instalados e como deve ser a arquitetura do sistema, entre outros

**Diagramas comportamentais** devem ser utilizados para especificar detalhes do comportamento do sistema (parte dinâmica), por exemplo: como as funcionalidades devem funcionar, como um processo de negócio deve ser tratado pelo sistema, como componentes estruturais trocam mensagens e como respondem às chamadas, entre outros

**Diagramas de interação** mostram como os objetos interagem entre si. Permitem assim modelar os aspectos dinâmicos de um sistema.

* **Classe e Visibilidade**

**Classes**: um modelo que vai representar um conjunto de objetos que possuem comportamento e características em comum.

**Perspectivas da Classe:**

* Análise: em um sistema de informação para uma biblioteca, as classes podem ser, por exemplo, Livro, Usuário, Empréstimo.
* Projeto: por exemplo, classes de controle, classes de fronteiras, classes de entidades.
* Implementação: refinamentos das classes de projetos onde métodos aparecem obrigatoriamente e são considerados aspectos de controle, estereótipos, pacotes, etc.
* Visibilidade: é um recurso para dar autoridade para outros elementos dentro da orientação a objetos utilizarem, mesmo estando em outro lugar. E funciona para método, atributo, classe.

**Tipos de visibilidade**:

* Público (public): qualquer classe do sistema pode ter acesso aos atributos e métodos definidos como public. Notação +
* Privado (private): somente a própria classe pode ter acesso. Notação -
* Protegido (protected): somente a própria classe e suas subclasses podem ter acesso. Notação #
* Pacote (default): qualquer classe dentro do mesmo pacote pode ser acessada. Notação ~

**​**