Visão Geral do Sistema

Herysson R. Figueiredo
herysson.figueiredo@ufn.edu.br

Modelagem de sistema de software

Uma característica intrínseca de sistemas de software é a complexidade de seu desenvolvimento, que aumenta à medida que o tamanho do sistema cresce.

Modelagem de sistema de software

Para a construção de sistemas de software mais complexos, é necessário um planejamento inicial ou seja o desenvolvimento de um ***modelo.***

Razões para se utilizar modelos na construção de sistemas

Gerenciamento da complexidade: um dos principais motivos de utilizar modelos é que há limitações no ser humano em como lidar com a complexidade. Pode haver diversos modelos de um mesmo sistema, cada qual descrevendo uma perspectiva do sistema a ser construído.

Razões para se utilizar modelos na construção de sistemas

Comunicação entre as pessoas envolvidas: sem dúvida o desenvolvimento de um sistema envolve a execução de uma quantidade significativa de atividades. Essas atividades se traduzem em informações sobre o sistema em desenvolvimento. Grande parte dessas informações corresponde aos modelos criados para representar o sistema

Razões para se utilizar modelos na construção de sistemas

Redução dos custos no desenvolvimento: no desenvolvimento de sistemas, seres humanos estão invariavelmente sujeitos a cometer erros, que podem ser tanto individuais quanto de comunicação entre os membros da equipe.

Razões para se utilizar modelos na construção de sistemas

Previsão do comportamento futuro do sistema: o comportamento do sistema pode ser discutido mediante uma análise dos seus modelos. Os modelos servem como um “laboratório”, em que diferentes soluções para um problema relacionado à construção do sistema podem ser experimentadas.

O paradigma da Orientação a Objetos

Pode-se dizer, então, que o termo “paradigma da orientação a objetos” é uma forma de abordar um problema.

O paradigma da Orientação a Objetos

Há alguns anos, Alan Kay, um dos pais do paradigma da orientação a objetos, formulou a chamada “analogia biológica”. Por meio dela, ele imaginou um sistema de software que funcionasse como um ser vivo. Nesse sistema, cada “célula” interagiria com outras células através do envio de mensagens com o objetivo realizar um objetivo comum. Além disso, cada célula se comportaria como uma unidade autônoma.

O paradigma da Orientação a Objetos

De uma forma mais geral, Kay pensou em como construir um sistema de software a partir de agentes autônomos que interagem entre si. Ele estabeleceu então os seguintes princípios da orientação a objetos:

- Qualquer coisa é um objeto.
- Objetos realizam tarefas por meio da requisição de serviços a outros objetos.
- Cada objeto pertence a uma determinada classe. Uma classe agrupa objetos similares.
- A classe é um repositório para comportamento associado ao objeto.
- Classes são organizadas em hierarquias.

O paradigma da Orientação a Objetos

O paradigma da orientação a objetos visualiza um sistema de software como uma coleção de agentes interconectados chamados objetos. Cada objeto é responsável por realizar tarefas específicas. Para cumprir com algumas das tarefas sob sua responsabilidade, um objeto pode ter que interagir com outros objetos. É pela interação entre objetos que uma tarefa computacional é realizada

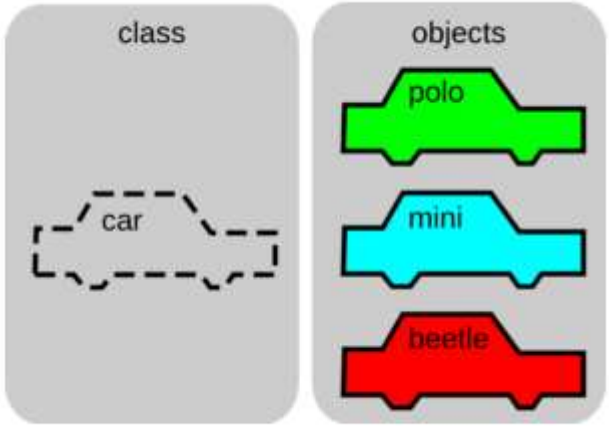
O paradigma da Orientação a Objetos

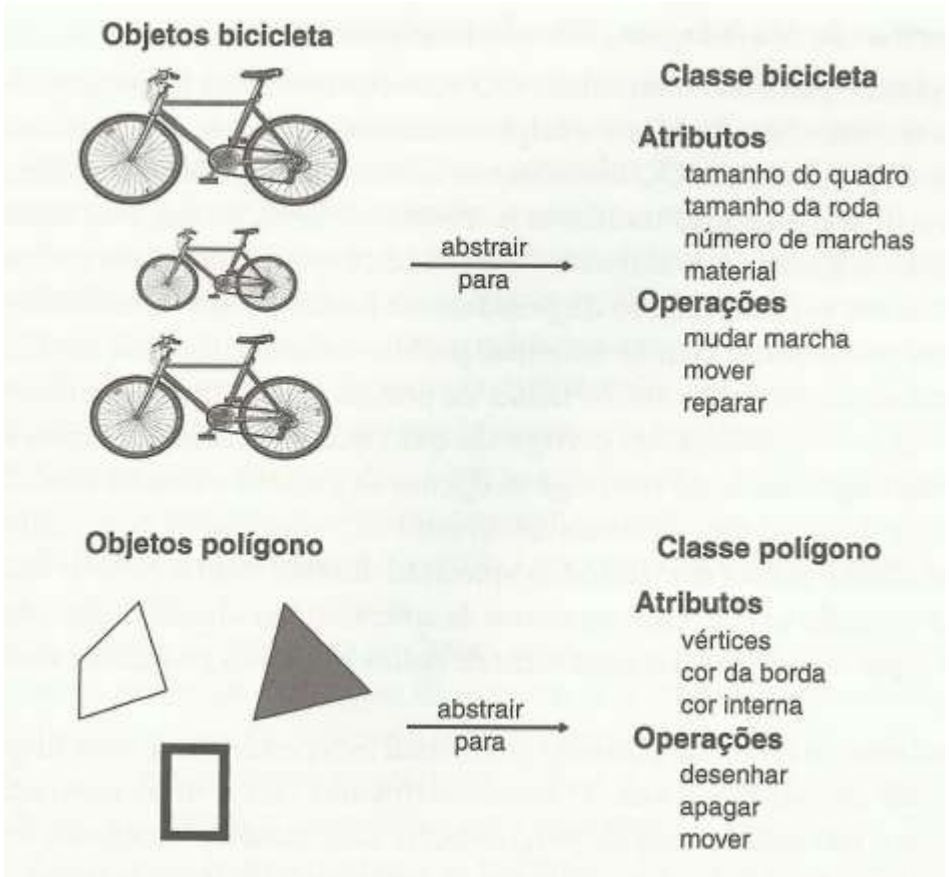
Um sistema de software orientado a objetos consiste em objetos em colaboração com o objetivo de realizar as funcionalidades desse sistema. Cada objeto é responsável por tarefas específicas. É graças à cooperação entre objetos que a computação do sistema se desenvolve.

Classe e Objeto

“Uma classe é como um molde a partir do qual objetos são construídos”

Classe e Objeto





Operação, mensagem e estado

Dá-se o nome de **operação** a alguma ação que um objeto sabe realizar quando solicitado.

Objetos não executam suas operações aleatoriamente. Para que uma operação em um objeto seja executada, deve haver um estímulo enviado a esse objeto.

Operação, mensagem e estado

Seja qual for a origem do estímulo, quando ele ocorre diz-se que o objeto em questão está recebendo uma **mensagem** requisitando que ele realize alguma operação.

Quando se diz na terminologia de orientação a objetos que **objetos de um sistema estão trocando mensagens** significa que esses objetos estão enviando mensagens uns aos outros com o objetivo de realizar alguma tarefa dentro do sistema no qual eles estão inseridos.

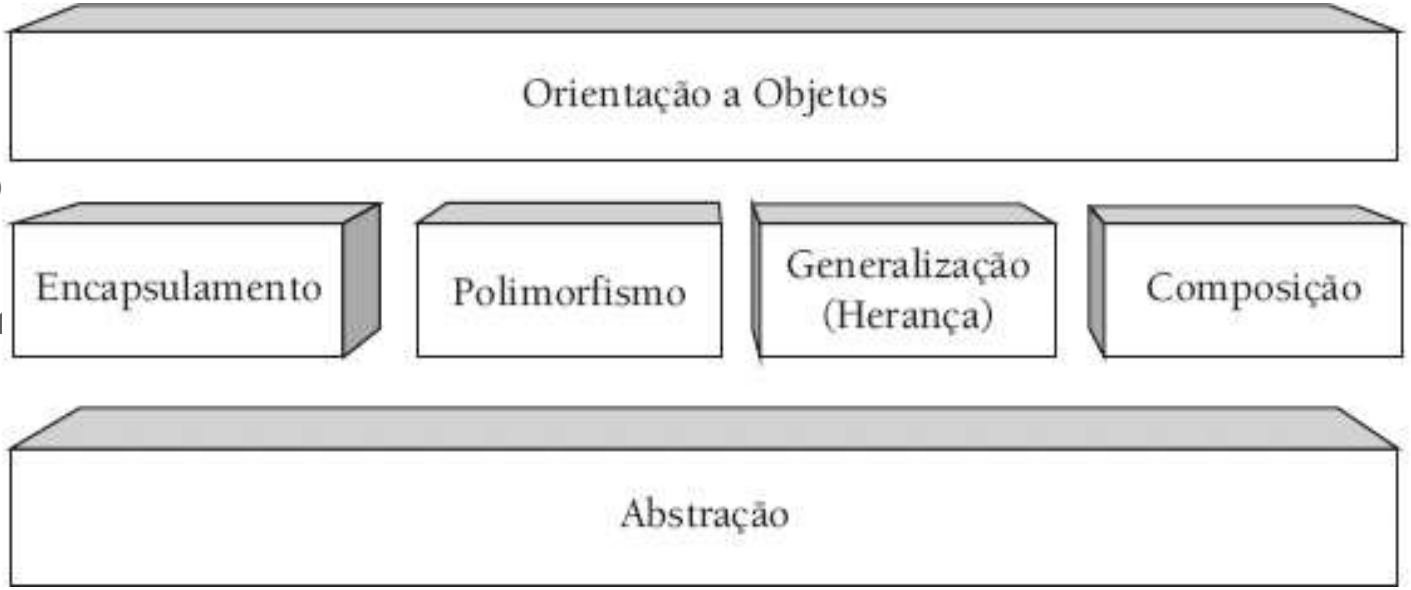
Operação, mensagem e estado

Por definição, o **estado** de um objeto corresponde ao conjunto de valores de seus atributos em um dado momento. Uma mensagem enviada a um objeto tem o potencial de mudar o estado desse objeto.

O Papel da abstração na orientação a objetos

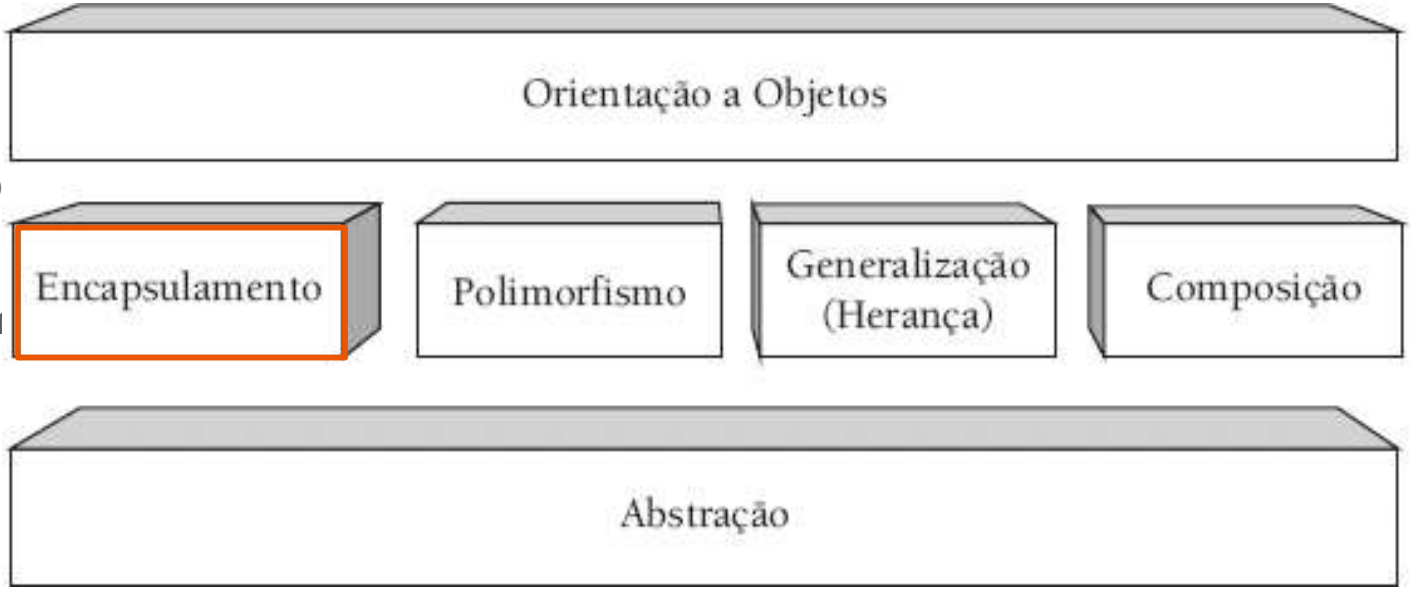
A **abstração** é um processo mental pelo qual nós, seres humanos, nos atemos aos aspectos mais importantes (relevantes) de alguma coisa, ao mesmo tempo em que ignoramos os menos importantes.

O Papel da abstração na orientação a objetos



A **ab** ...os
atem ...ao
mesn

O Papel da abstração na orientação a objetos

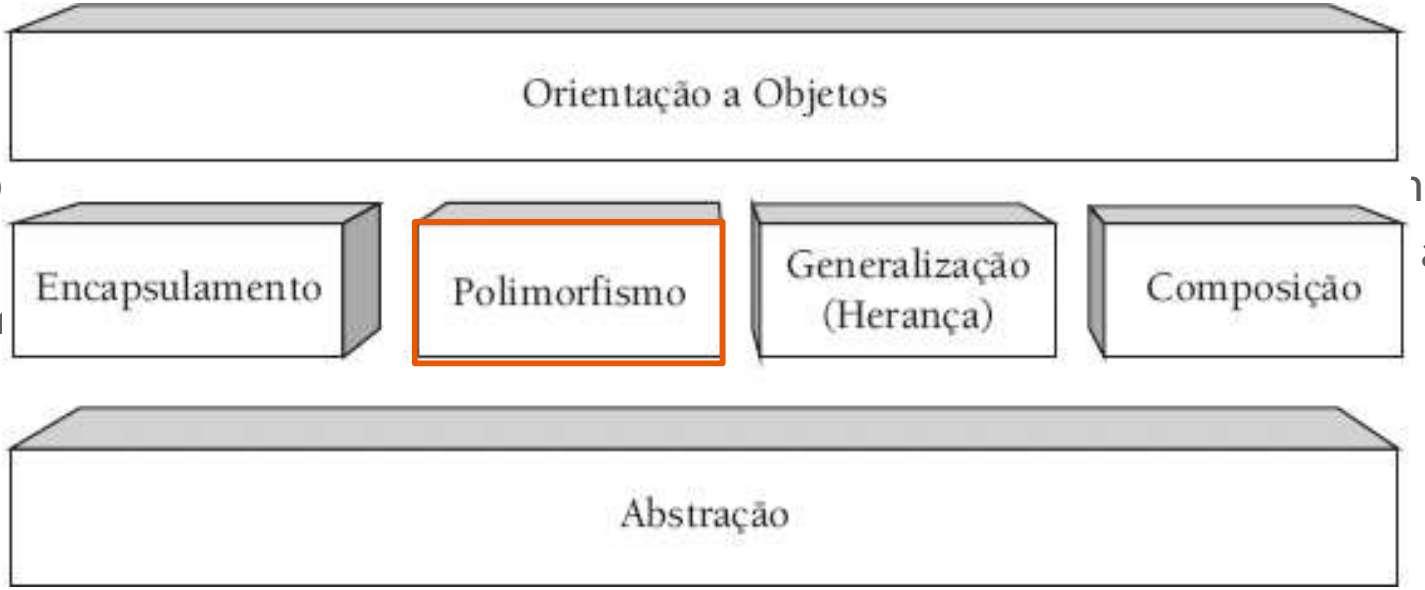


A **ab** ...os
atem ... ao
mesn

Encapsulamento

O mecanismo de **encapsulamento** é uma forma de restringir o acesso ao comportamento interno de um objeto.

O Papel da abstração na orientação a objetos

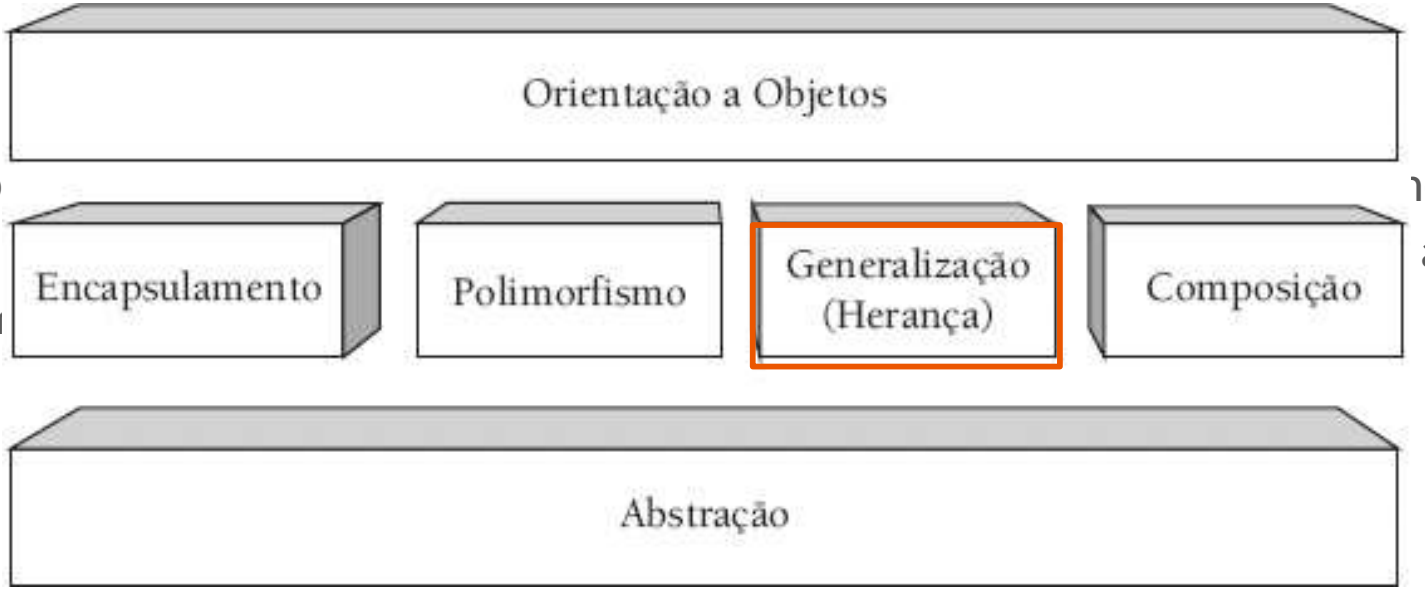


A **ab** ...os
atem ...ao
mesn

Polimorfismo

Polimorfismo é o princípio pelo qual duas ou mais classes derivadas da mesma superclasse podem invocar métodos que têm a mesma assinatura, mas comportamentos distintos.

O Papel da abstração na orientação a objetos



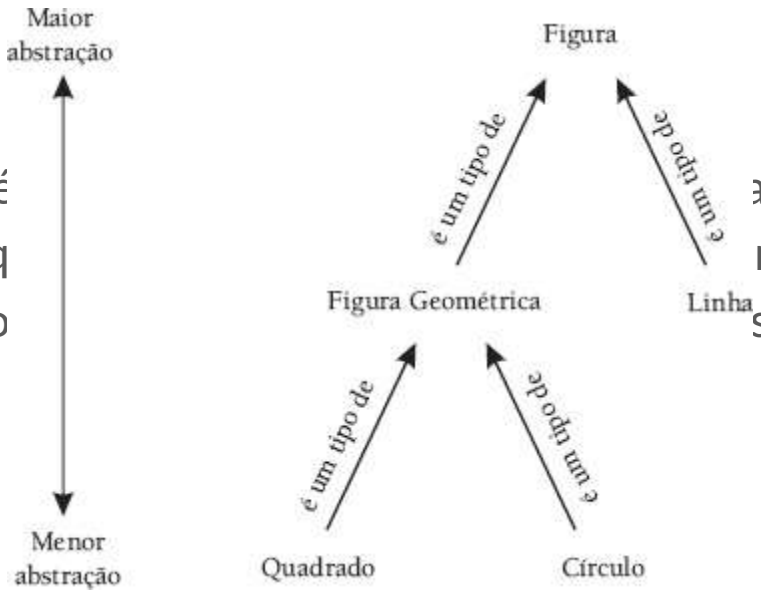
A **ab** ...os
atem ...ao
mesn

Generalização (Herança)

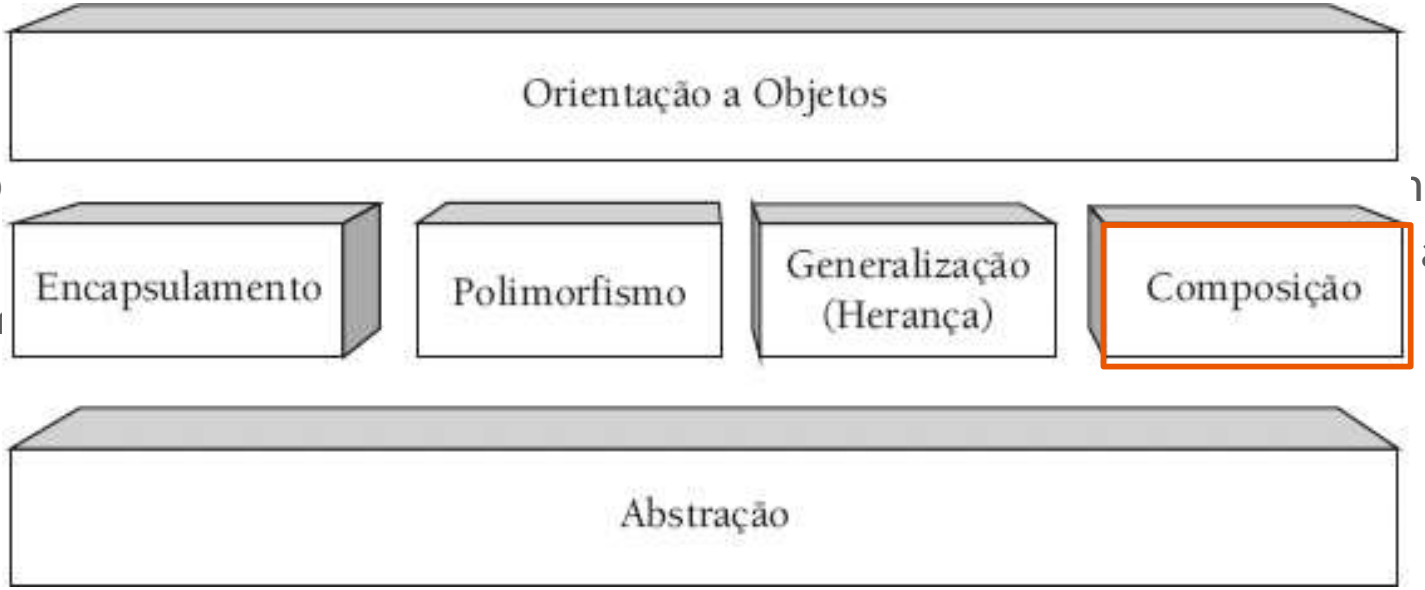
A generalização é outra forma de abstração utilizada na orientação a objetos. Declara que as características e o comportamento comuns a um conjunto de objetos podem ser abstraídos em uma classe.

Generalização (Herança)

A generalização é [illegible]ada na orientação a objetos. Declara q[illegible]mento comuns a um conjunto de objeto[illegible]se.



O Papel da abstração na orientação a objetos



A **ab** ...os
atem ...ao
mesn

Composição

“Objetos compostos por outros objetos”

UML e Visões de um sistema

UML

A construção da UML - *Unified Modeling Language* (Linguagem de Modelagem Unificada) teve muitos contribuintes, mas os principais atores no processo foram Grady Booch, James Rumbaugh e Ivar Jacobson. Esses três pesquisadores costumam ser chamados de “os três amigos”.

UML

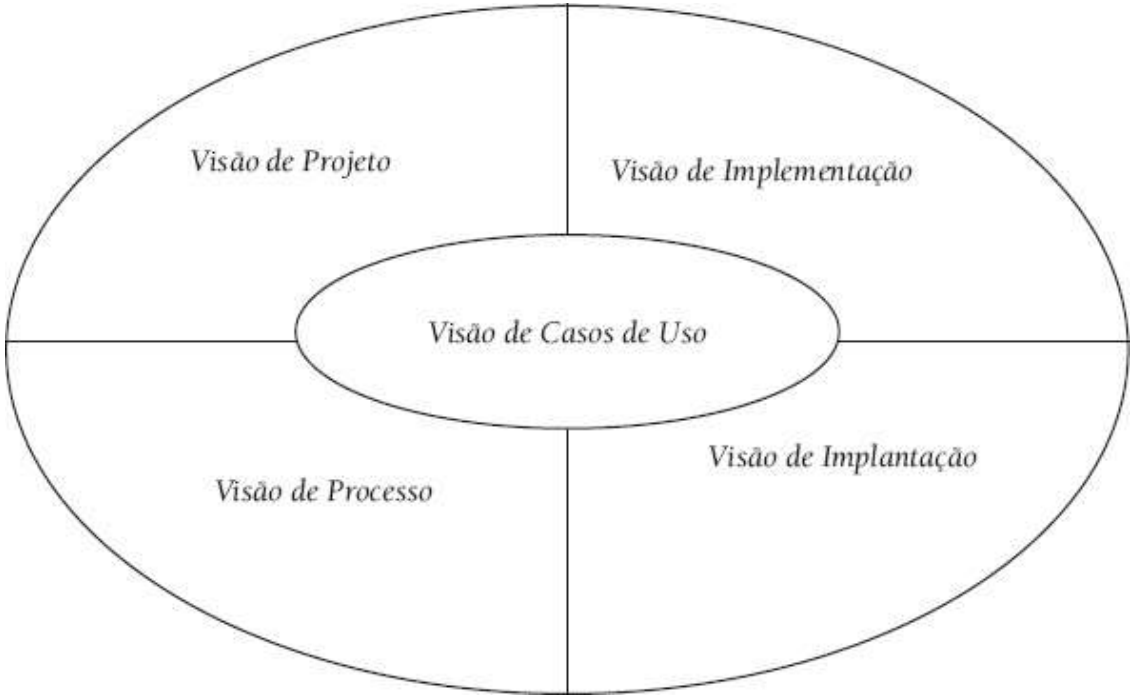
No processo de definição inicial da UML, esses pesquisadores buscaram aproveitar o melhor das características das notações preexistentes, principalmente das técnicas que haviam proposto anteriormente (essas técnicas eram conhecidas pelos nomes Booch Method, OMT e OOSE).

Visões de um sistema

O desenvolvimento de um sistema de software complexo demanda que seus desenvolvedores tenham a possibilidade de examinar e estudar esse sistema a partir de perspectivas diversas.

Os autores da UML sugerem que um sistema pode ser descrito por cinco visões interdependentes desse sistema.

Visões de um sistema



Visão de Casos de Uso

Descreve o sistema de um ponto de vista externo como um conjunto de interações entre o sistema e os agentes externos ao sistema. Esta visão é criada em um estágio inicial e direciona o desenvolvimento das outras visões do sistema.

Visão de Projeto

Enfatiza as características do sistema que dão suporte, tanto estrutural quanto comportamental, às funcionalidades externamente visíveis do sistema.

Visão de Implementação

Abrange o gerenciamento de versões do sistema, construídas pelo agrupamento de módulos (componentes) e subsistemas.

Visão de Implantação

Corresponde à distribuição física do sistema em seus subsistemas e à conexão entre essas partes.

Visão de Processo

Esta visão enfatiza as características de concorrência (paralelismo), sincronização e desempenho do sistema.

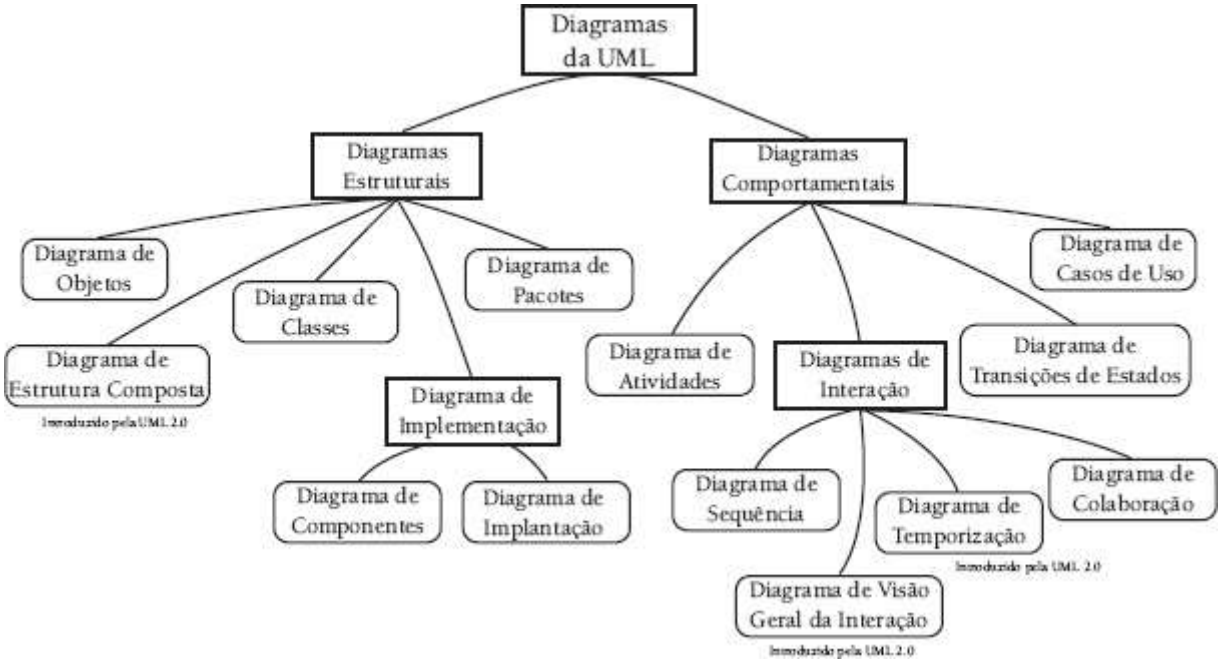
Diagramas da UML

Um processo de desenvolvimento que utilize a UML como linguagem de suporte à modelagem envolve a criação de diversos documentos. Esses documentos podem ser textuais ou gráficos. Na terminologia da UML, eles são denominados **artefatos de software**, ou simplesmente artefatos. São os artefatos que compõem as visões do sistema.

Diagramas da UML

Os artefatos gráficos produzidos durante o desenvolvimento de um sistema de software orientado a objetos (SSOO) podem ser definidos pela utilização dos diagramas da UML. Os 13 diagramas da UML 2.0

Diagramas da UML



Bibliografia

BEZERRA, Eduardo. Princípios de Análise e Projeto de Sistemas com UML. 2. ed. Rio de Janeiro: Elsevier, 2006.

LARMAN, Craig. Utilizando UML e padrões. 3. ed. Porto Alegre: Bookman, 2007.

WAZLAWICK, Raul. Análise e Projetos de Sistemas de Informação orientados a objetos. 2. ed. Rio de Janeiro: Campus, 2011

Bibliografia

BLAHA, Michael; RUMBAUGH, James. Modelagem e projetos baseados em objetos com UML 2. 2. ed. Rio de Janeiro: Elsevier, 2006

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. UML: guia do usuário. 2. ed. Rio de Janeiro: Campus, 2006.

GAMMA, Erich. Padrões de projeto: soluções reutilizáveis de software orientado a objetos. Porto Alegre: Bookman, 2000.

HUMPHREY, Watts S. A discipline for software engineering. 7. ed. Massachusetts: Addison-Wesley, 1997.

SOMMERVILLE, Ian. Engenharia de Software. 8. ed. São Paulo: Prentice Hall, 2007.