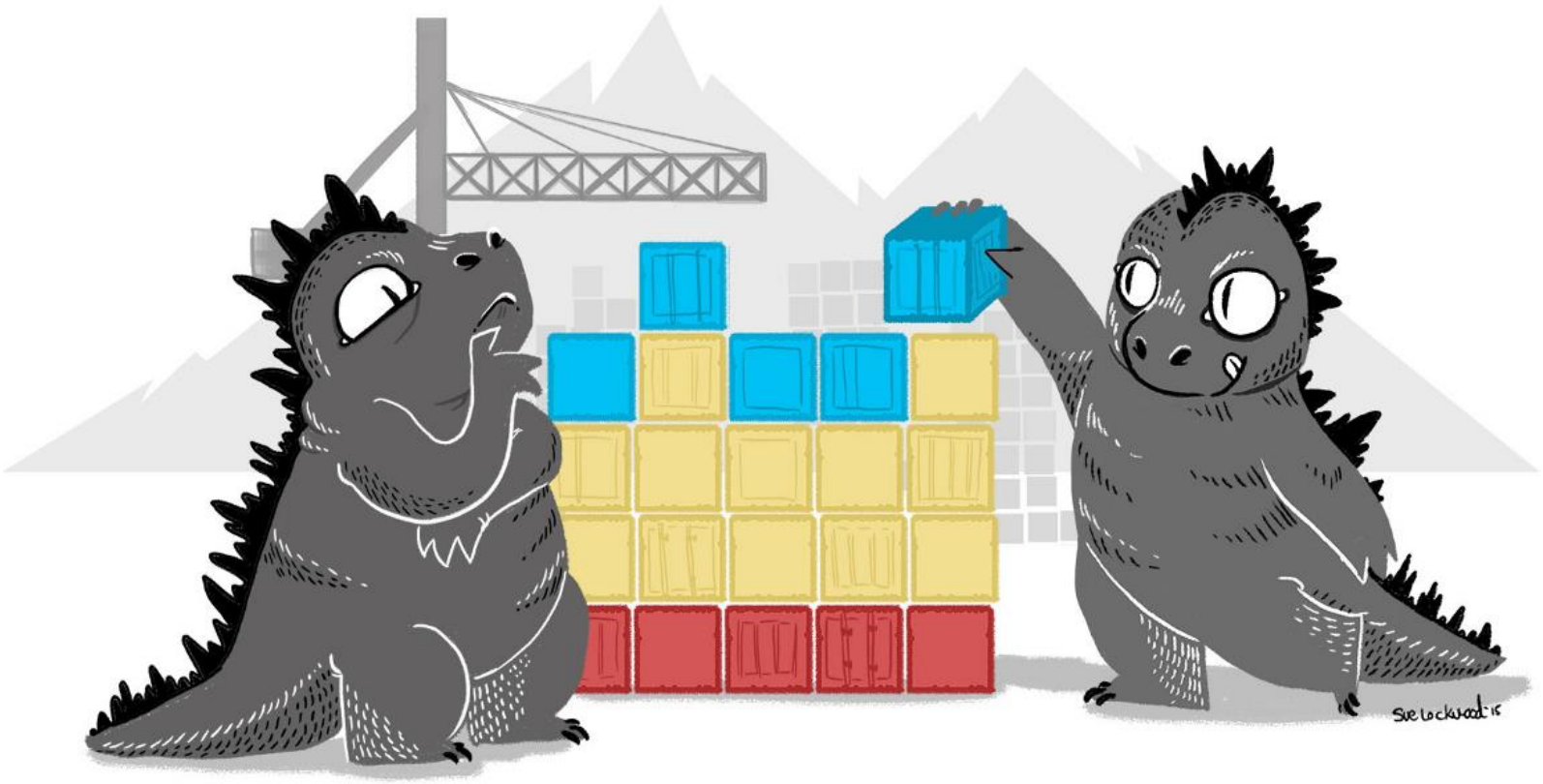


Curso: Análise e Desenvolvimento de Sistemas

Disciplina: Análise e Projeto Orientado a Objetos

Profª Drª Marcia Cassitas Hino

1º Semestre/2024

Universidade Tuiuti do Paraná

Projeto OO

- O que define uma orientação a objetos é a maneira de se entender o problema, como ele é “particionado”.

Universidade Tuiuti do Paraná

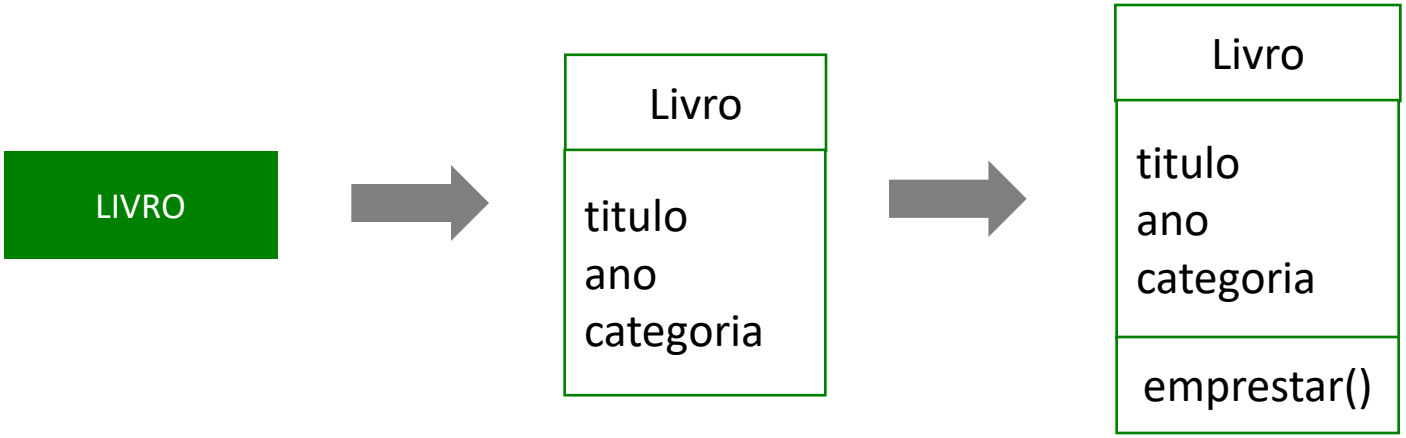
Projeto OO

- Objetivo de um projeto OO é de projetar sistemas usando **objetos** e **classes** de objetos.

- Características de um projeto OO:
  - Os projetistas pensam em termos de **coisas**, em vez de funções.
  - ***A funcionalidade do sistema é expressa em termos de serviços oferecidos pelos objetos.***
    - Objetos são abstrações do mundo real ou entidades do sistema que se auto gerenciam.
    - Objetos se comunicam por passagem de mensagem.

Projeto OO

- Na fase de análise de um projeto OO, busca-se identificar objetos e descrevê-los.



Projeto OO

- Vantagens de um projeto OO:
  - Facilidade de manutenção, onde os objetos podem ser entendidos como entidades independentes;
  - Os objetos são componentes potencialmente reutilizáveis;
  - Para vários sistemas existe um mapeamento nítido das entidades do mundo real para objetos no sistema.

Unified Modeling Language (UML)

- Linguagem visual utilizada para **modelar sistemas** baseados no **paradigma de orientação a objetos**.
- Aplicada a todos os domínios de aplicação (ou problema).
- Reconhecida internacionalmente pela indústria de *software*.

UML

- A UML é uma linguagem de modelagem, uma notação, cujo objetivo é auxiliar a definição de características do sistema, tais como requisitos, comportamentos, estrutura lógica e a dinâmica dos processos.

- É uma linguagem visual aprovada pela OMG (*Object Management Group*) que usa elementos gráficos e textuais com sintaxe (desenho padrão) e semântica (significado e objetivo de utilização).

O que a UML não é?

- Não é uma linguagem de programação.
- Não é uma metodologia ou processo de desenvolvimento.

UML

- Surgiu, no início da década de 90, da união de três métodos de modelagem.

Grady Booch

Método Booch

- Diagrama de objetos
- Diagrama de classes
- Diagrama de estados
- Diagrama de processos

Ivar Jacobson

Objectory* (OOSE) *Process

- Diagrama de caso de uso
- Subsistemas

James Rumbaugh

***Object Modeling Technique* (OMT)**

- Diagrama de classe
- Diagrama de estado

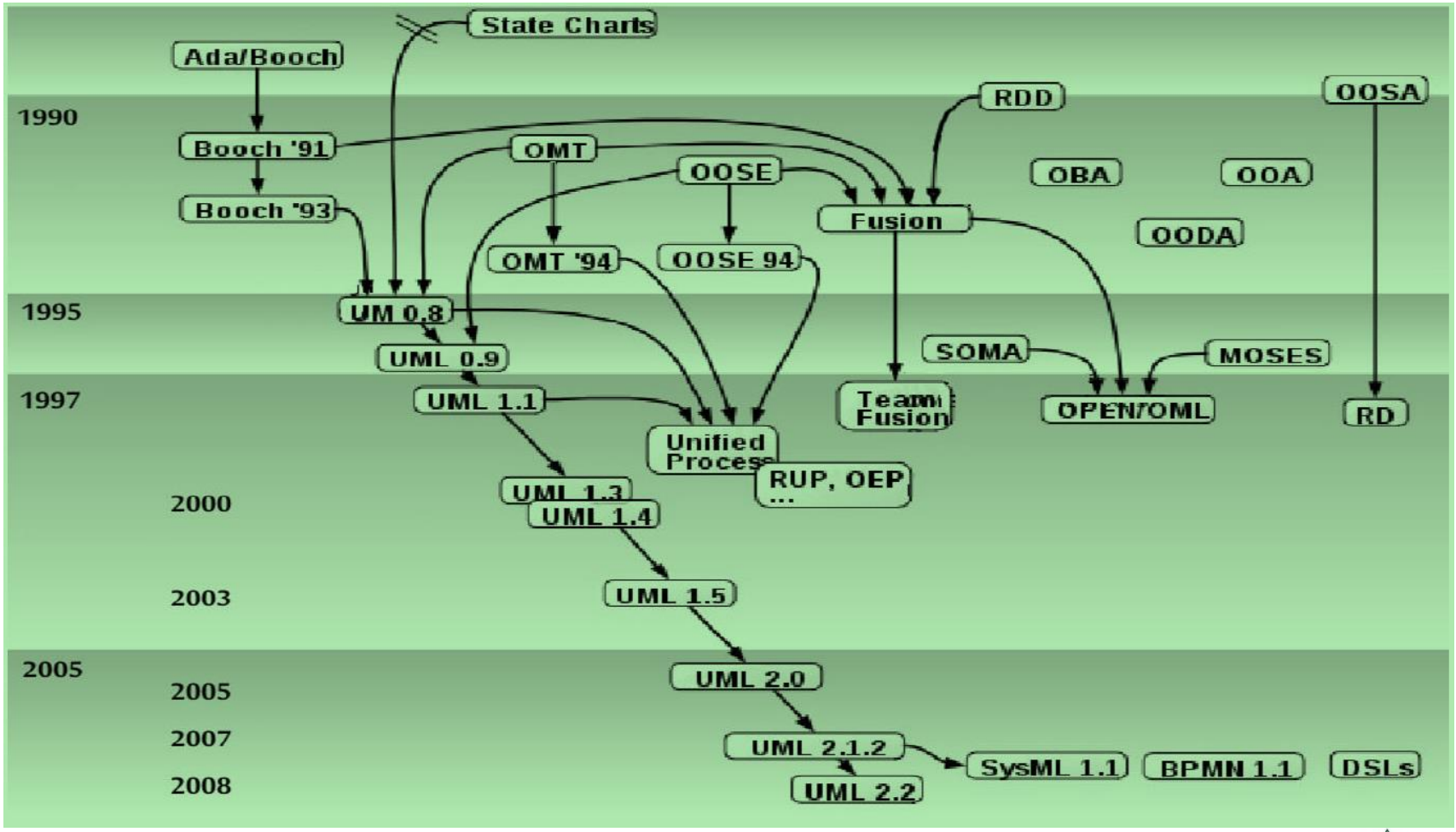
Harel (STATECHARTS)

- Diagrama de atividades

Coleman (FUSION)

- Grafo de interação de objetos

UML



UML

- 1995: Booch, Jacobson e Rumbaugh começaram a unificar suas notações.
- 1996: Primeira versão (beta) da UML foi liberada.
- 1996/97: Grandes empresas passaram a contribuir, surgindo a “UML Partners” com empresas como HP, IBM, Microsoft, Oracle, e outras.
- 1997: UML foi adotada pela OMG (*Object Management Group*) como linguagem padrão de modelagem.

UML

Incorpora noções do desenvolvimento de software totalmente visual e se baseia em diagramas que são modelados e classificados em visões de abstração.

UML

É uma linguagem não proprietária da 3º geração da linguagem de modelagem e é independente da linguagem de programação.

Notação UML

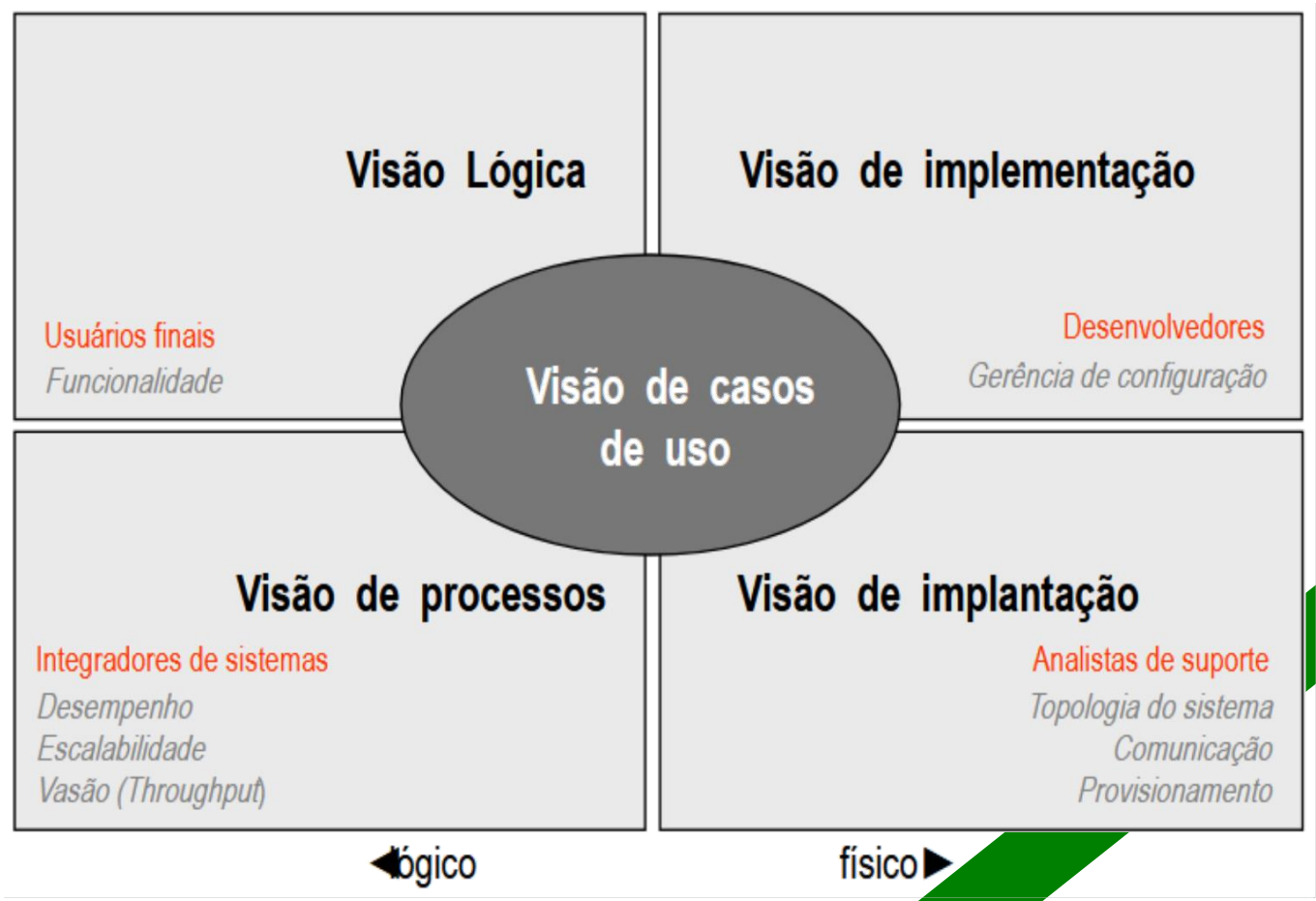
- É uma **união de sintaxe gráfica de vários métodos**, com certo número de símbolos removidos (porque eram confusos, supérfluos ou pouco usados) e com outros símbolos adicionados.

- O resultado é uma única, comum e ampla **linguagem de modelagem** utilizável por desenvolvedores de software orientado a objetos.

Visões da UML

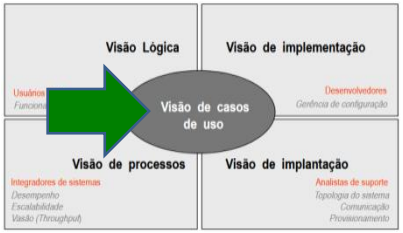
- Mostram diferentes aspectos do sistema que está sendo modelado.
- A visão não é um gráfico, mas uma abstração consistindo em uma série de diagramas.
- Cada visão mostrará aspectos particulares do sistema, dando enfoque a ângulos e níveis de abstrações diferentes.

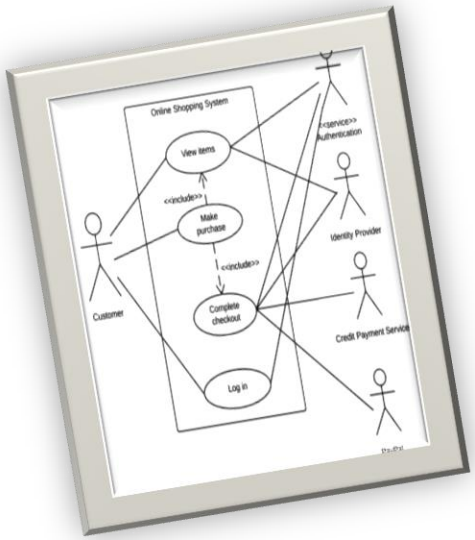
VISÃO 4+1



Universidade Tuiuti do Paraná

VISÃO DE CASO DE USO



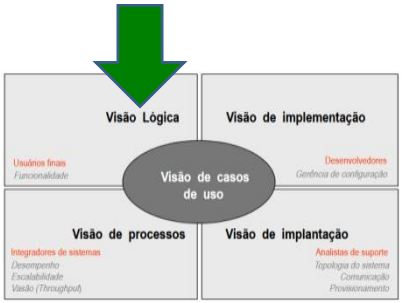


- Abrange a descrição do comportamento do sistema conforme esse é visto pelos **usuários finais** do sistema (analistas, pessoal de teste...).
- Utiliza-se da criação de casos de uso, diagrama de interação, diagrama de estado e diagrama de atividade.

A partir de um ponto de vista externo, descreve como um conjunto de interações entre o sistema e agentes externos ao sistema.

Universidade Tuiuti do Paraná

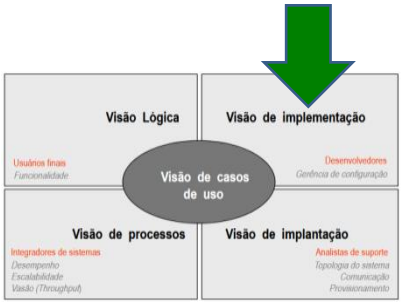
VISÃO LÓGICA



- Abrange as classes, interfaces e colaboração que formam o vocabulário do problema e de sua solução.

- Proporciona uma perspectiva de suporte para os **requisitos funcionais** de sistemas, ou seja, serviços que o sistema deverá fornecer.

VISÃO DE IMPLEMENTAÇÃO

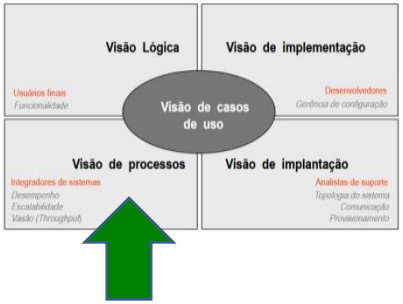


- Descreve como a funcionalidade do sistema será implementada.

Foco no gerenciamento de versões, agrupamento de módulos e subsistemas.

Universidade Tuiuti do Paraná

VISÃO DE PROCESSOS

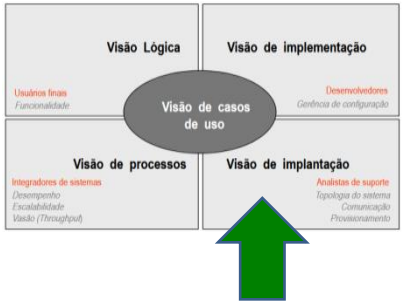


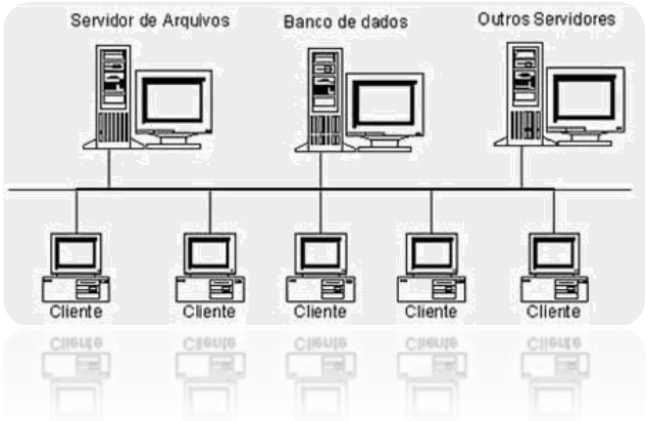
- Abrange os *threads* e os processos que formam o mecanismo de concorrência e de sincronização do sistema.
- Essa visa cuida principalmente da questão referente ao desempenho, à escalabilidade e ao *throughput* (taxa de transferência de dados de uma rede).

Foco nas características de paralelismo, sincronização e desempenho do sistema.

Universidade Tuiuti do Paraná

VISÃO DE IMPLANTAÇÃO





- Essa visão envolve principalmente o gerenciamento de configurações das versões do sistema.

- Abrange componentes e os arquivos utilizados para montagem e fornecimento do sistema físico, contemplando topologia, instalação, desempenho e protocolos de comunicação.

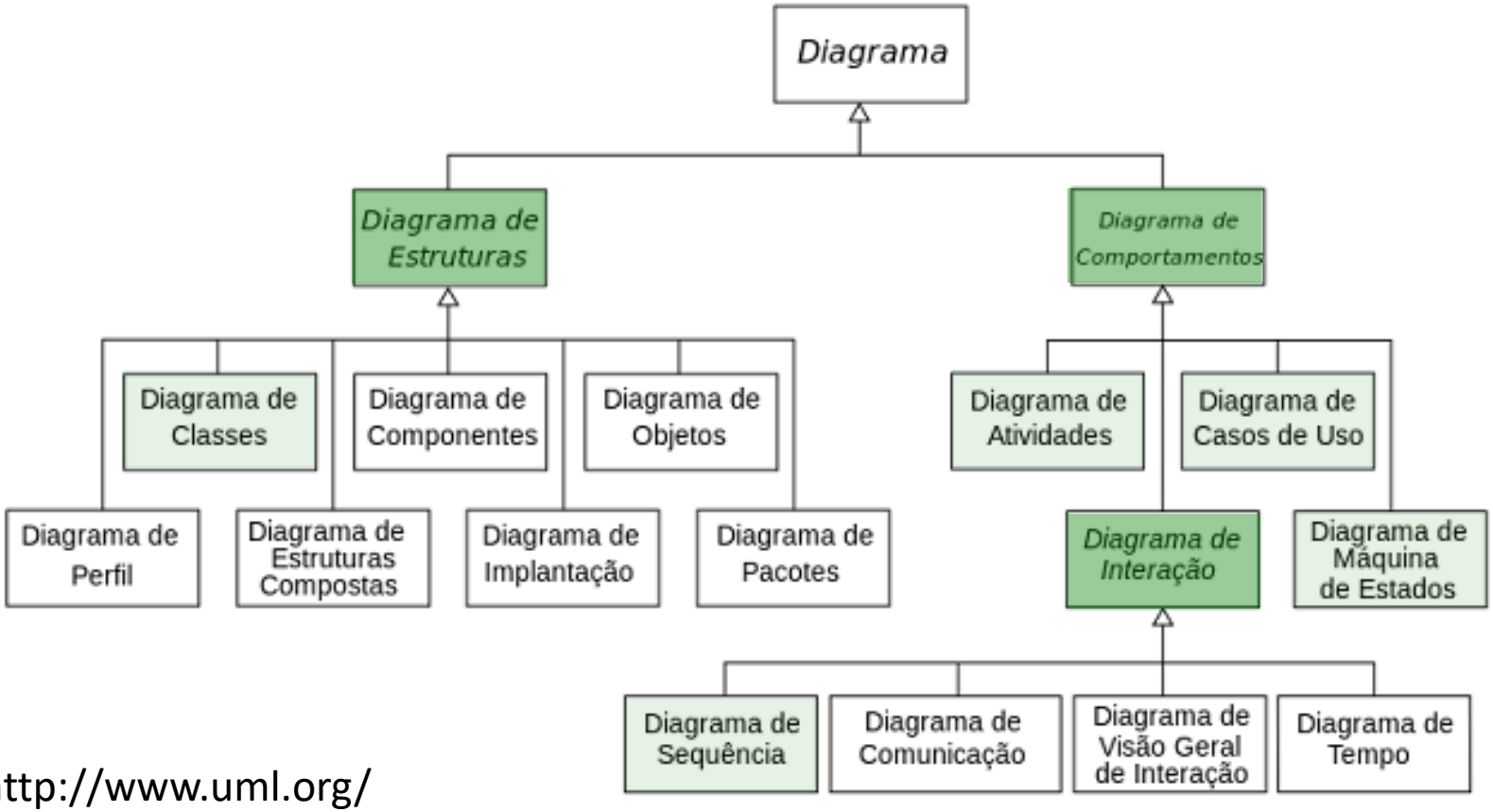
Nem todos os sistemas precisam ter todas as visões.

- Sistemas pequenos não necessitam da visão de implantação.
- Visão de implantação pode ser ignorada quando a arquitetura é de processador único.

Alguns sistemas necessitam visões adicionais:

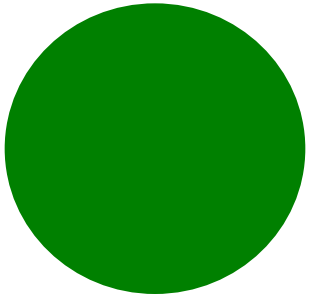
- Visão de dados.
- Visão de segurança das informações.

DIAGRAMAS



http://www.uml.org/
http://www.omg.org/

Categoria	Diagramas	Descrição
Diagramas Dinâmicos	Casos de uso	Expressam a funcionalidade de um sistema
	Atividades	Representam o fluxo de atividades dos processos de negócio
	Interatividade	Apresenta um fluxo de atividades, mostrando como elas trabalham em uma sequência de eventos
	Sequências	Define a ordem e a troca das mensagens entre objetos
	Comunicação	Representa o diagrama anterior de colaboração
	Máquina de estados	Representa as ações ocorridas em resposta ao recebimento de eventos
	Temporal	Mostra mudança de estado de um objeto
Diagramas Estruturais	Classes	Apresenta elementos conectados por relacionamentos
	Objetos	Apresenta objetos e valores de dados
	Componentes	Mostra dependências entre componentes de software
	Pacotes	Usado para organizar elementos de modelos e mostrar dependências entre eles
	Implantação	Mostra a arquitetura do sistema em tempo de execução, as plataformas de software, etc.
	Estrutura composta	Usado para mostrar a composição de uma estrutura complexa

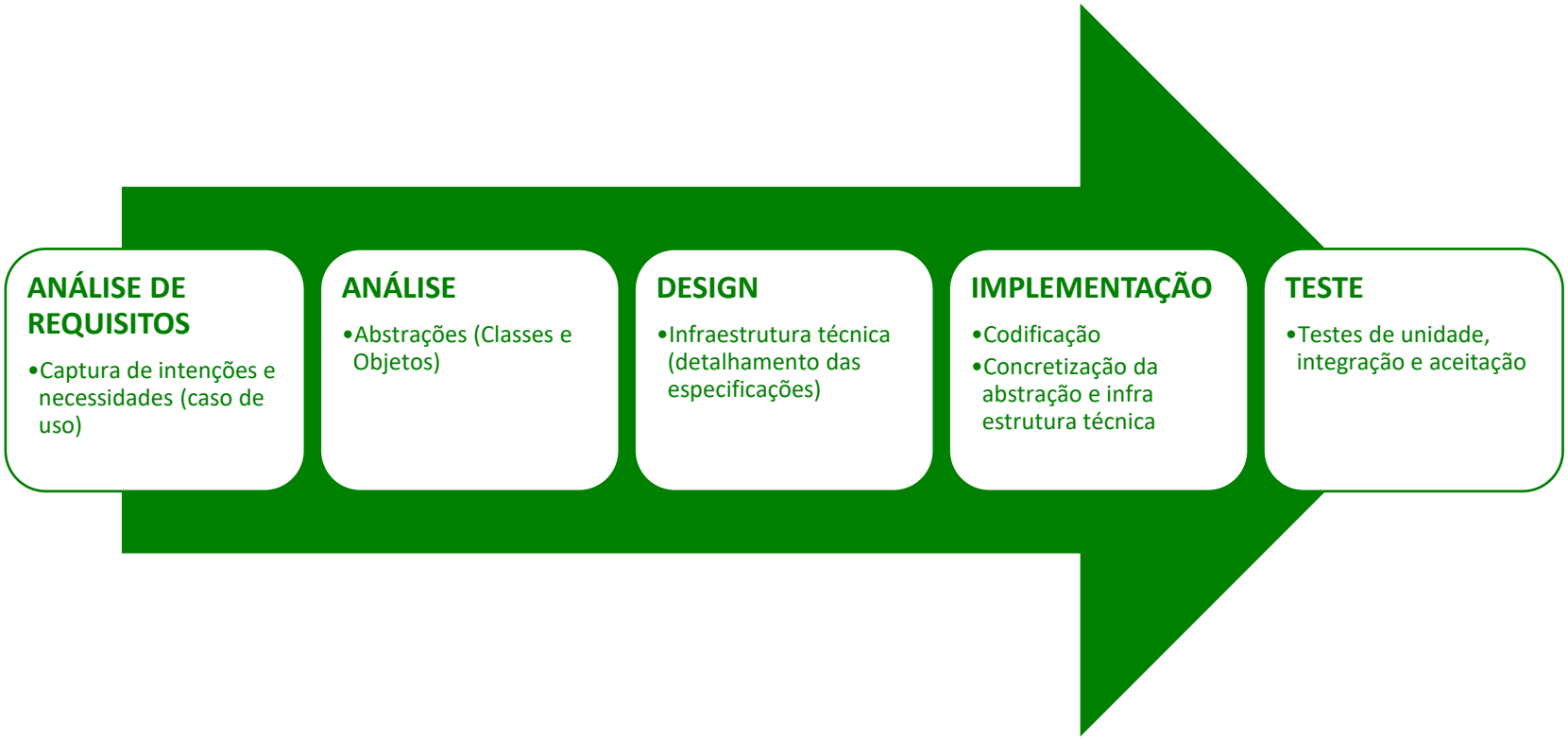


Fases de desenvolvimento

O desenvolvimento de um sistema em UML divide-se em cinco fases:

- Análise de requisitos
- Análise
- Design
- Implementação (programação)
- Testes

Fases de desenvolvimento



Fase 1: Análise de Requisitos

1. Compreende o problema e levanta necessidades (requisitos funcionais e não funcionais);
2. Não deve ter questões técnicas;
3. Deve ter envolvimento do usuário;
4. Desejável que tenha ordem de prioridade (com base em valor agregado, adição de valor);
5. Gera documento de requisitos;
6. Atendimento dos requisitos é uma das formas de medir a qualidade do sistema;
7. Etapa mais importante em termos de retorno de investimento;
8. Serve como termo de consentimento.

Fase 2: Análise

1. Detalhamento dos requisitos, gerando suas especificações;
2. Não considera o ambiente tecnológico;
3. Gera modelos e estruturas de classes.

Geração de modelos que devem ser validados, para assegurar que estão alinhados com os requisitos e que as necessidades serão atendidas.

Fase 2: Análise

- Prototipagem é uma técnica utilizada para validar requisitos.
- Opcional mas frequente.
- Complementa a construção de modelos de sistemas.
- Modifica e refina os modelos existentes.

Fase 3: Design

- Define-se o “como”, incluindo arquitetura do sistema, interface gráfica, linguagem de programação, banco de dados, etc.;
- Gera descrição computacional;
- Gera-se:
  - o projeto de arquitetura: Distribuição das classes de objetos em subsistemas e componentes
  - o projeto detalhado: Colaboração entre os objetos moldando as funcionalidades, projeto de interface e de banco de dados (utiliza o diagrama de classes, de casos de uso, interação, estados e atividades)

Fase 4: Implementação

1. Traduz a descrição computacional em código executável;
2. Gera código.

Universidade Tuiuti do Paraná

Fase 5: Teste

1. Valida-se a funcionalidade gerada com a especificação feita;

2. Gera relatório de testes com os erros detectados, para se realizar as correções devidas.

FERRAMENTAS DE MODELAGEM

- Astah Community - http://astah.net/download
- Together - http://www.borland.com/products/downloads/download_together.html
- IBM RationalRose - http://www.ibm.com/software/rational
- Omondo – Plugin para Eclipse – http://www.omondo.com/

Rational Unified Process (RUP)

- É uma metodologia para desenvolvimento de processos que utiliza abordagem de orientação a objetos em sua concepção.
- Também conhecida como *Unified Process Model*.
- Usa como base a notação UML para ilustrar os processos.
- É um método proprietário de engenharia de software criado pela *Rational Software Corporation*, posteriormente adquirida pela IBM.
- Bastante adaptável.
- Aplicável a grandes projetos, com grandes equipes.

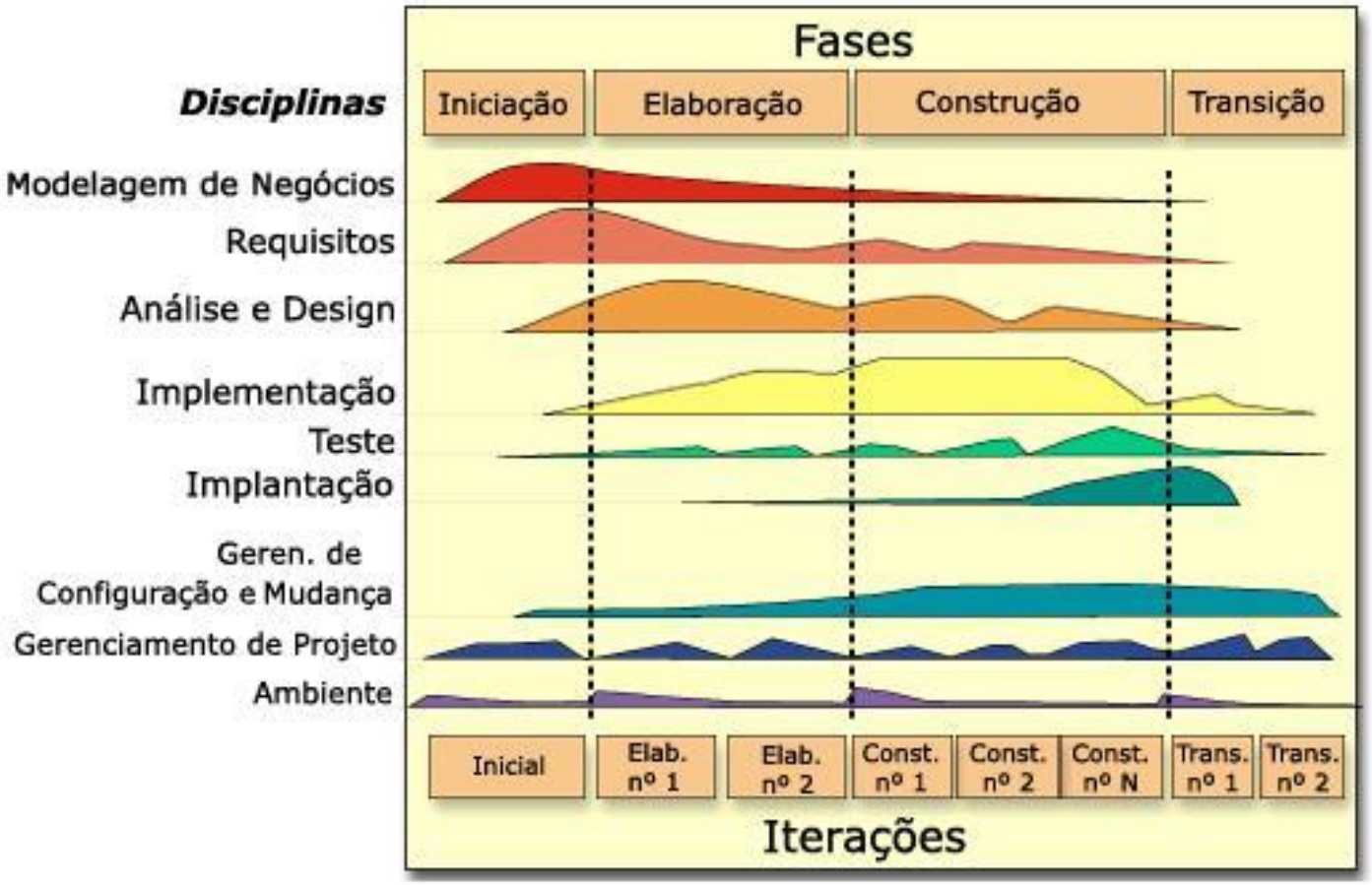
Processo de desenvolvimento:

- A UML é totalmente independente do processo.
- Não se limita ao ciclo de vida de desenvolvimento.

Porém para melhor aproveitamento da UML, é esperado se considerar um processo com:

1. Orientação a caso de uso
2. Centrado na arquitetura
3. Interativo e incremental

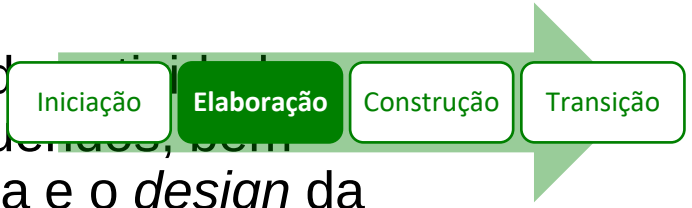
Rational Unified Process (RUP)



FASES RUP

- **Iniciação**
  Quando se especifica
  da visão do sistema e
  se define o escopo do sistema.

- **Elaboração**
  Quando se faz o planejamento d
  necessárias e dos recursos requ
  como a especificação do sistema e o *design* da
  sua arquitetura. Foco na arquitetura.



FASES RUP

- **Construção**
  Desenvolvimento do produto como uma série de interações incrementais. Desenvolver o sistema.

- **Transição**
  Fornecimento do produto para o usuário (fabricação, distribuição e treinamento). Foco na implantação.

Obrigada!