

# Tópicos em Engenharia de Software (Optativa III) - AULA 2

Prof. Andrêza Leite

[andreza.lba@gmail.com](mailto:andreza.lba@gmail.com)

(81 )9801-6619

# Engenharia de Software



# Objetivo da aula

- Depois desta aula você terá uma revisão sobre o que é a engenharia de software, os seus objetivos e conceitos básicos.

# Leitura recomendada

- Capítulo 1 (Software and Software Engineering) do livro Software Engineering — A Practitioner's Approach (de Roger Pressman)
- Capítulo 1 (Introduction) do livro Software Engineering (de Ian Sommerville)

# O que é a Engenharia de Software?

- Estudo ou aplicação de abordagens sistemáticas, econômicas e quantificáveis para o desenvolvimento, operação e manutenção de software de qualidade.
- Engenheiros de software devem adotar uma abordagem sistemática e organizada para seu trabalho e usar ferramentas e técnicas/métodos apropriados dependendo do problema a ser solucionado, das restrições de desenvolvimento e dos recursos disponíveis

# O que é software?

- Programas de computador e documentação associada
- Produtos de software podem ser desenvolvidos para um cliente particular ou podem ser desenvolvidos para um mercado geral

# Objetivos da Engenharia de Software

- Controle sobre o desenvolvimento de software dentro de **custos**, **prazos** e níveis de **qualidade** desejados
- Produtividade no desenvolvimento, operação e manutenção de software
- Qualidade versus Produtividade
- Permitir que profissionais tenham controle sobre o desenvolvimento de software dentro de custos, prazos e níveis de qualidade desejados

# Características da Engenharia de Software

- A Engenharia de Software se refere a software (sistemas) desenvolvidos por grupos ao invés de indivíduos
- usa princípios de engenharia ao invés de arte, e
- inclui tanto aspectos técnicos quanto não técnicos



# O que é um software de qualidade?

- O software que satisfaz os requisitos solicitados pelo usuário. Deve ser fácil de manter, ter boa performance, ser confiável e fácil de usar
- Alguns atributos de qualidade
  - **Manutenibilidade**
    - O software deve evoluir para atender os requisitos que mudam
  - **Eficiência**
    - O software não deve desperdiçar os recursos do sistema
  - **Usabilidade**
    - O software deve ser fácil de usar pelos usuários para os quais ele foi projetado

# Qualidade de Software (um exemplo para o Varejo)

- **Correto**
  - A loja não pode deixar de cobrar por produtos comprados pelo consumidor
- **Robusto e altamente disponível**
  - A loja não pode parar de vender
- **Eficiente**
  - O consumidor não pode esperar
  - A empresa quer investir pouco em recursos computacionais (CPU, memória, rede)

# Qualidade de Software (um exemplo para o Varejo)

- Amigável e fácil de usar
  - A empresa quer investir pouco em treinamento
- Altamente extensível e adaptável
  - A empresa tem sempre novos requisitos (para ontem!)
  - A empresa quer o software customizado do seu jeito (interface, teclado, idioma, moeda, etc.)
- Reusável
  - Várias empresas precisam usar partes de um mesmo sistema

# Qualidade de Software

## (um exemplo para o Varejo)

- Aberto, compatível, de fácil integração com outros sistemas
  - A empresa já tem controle de estoque, fidelização, etc.
- Portável e independente de plataforma (hw e sw)
  - A empresa opta por uma determinada plataforma
- Baixo custo de instalação e atualização
  - A empresa tem um grande número de PDVs

# Produtividade

- Custo de desenvolvimento reduzido
  - A empresa consumidora quer investir pouco em software
  - A empresa produtora tem que oferecer “software barato”
- Tempo de desenvolvimento reduzido
  - Suporte rápido às necessidades do mercado

# “Software Barato”

*Nem tanto resultado de baixos custos de desenvolvimento, mas principalmente da distribuição dos custos entre vários clientes.*

*Reuso, extensibilidade e adaptabilidade são essenciais para viabilizar tal distribuição.*

# Importância da Engenharia de Software

- Qualidade de software e produtividade garantem:
  - Disponibilidade de serviços essenciais
  - Segurança de pessoas
  - Competitividade das empresas
    - Produtores
    - Consumidores

# Mas, na realidade, temos a Crise de Software...

- 25% dos projetos são cancelados
- o tempo de desenvolvimento é bem maior do que o estimado
- 75% dos sistemas não funcionam como planejado
- a manutenção e reutilização são difíceis e custosas
- os problemas são proporcionais a complexidade dos sistemas



# Causas da Crise de Software

- **Essências**
  - Complexidade dos sistemas
  - Dificuldade de formalização
- **Acidentes**
  - Má qualidade dos métodos, linguagens, ferramentas, processos, e modelos de ciclo de vida
  - Falta de qualificação técnica

# Elementos e Atividades da Engenharia de Software

- Elementos

- Modelos do ciclo de vida do software
- Linguagens
- Métodos
- Ferramentas
- Processos

- Atividades

- Modelagem do negócio
- Elicitação de requisitos
- Análise e Projeto
- Implementação
- Testes
- Distribuição
- Planejamento
- Gerenciamento
- Gerência de Configuração e Mudanças
- Manutenção

# Atividades e Artefatos da Engenharia de Software

- Atividades

- Modelagem do negócio
- Elicitação de requisitos
- Análise e Projeto
- Implementação
- Testes
- Distribuição
- Planejamento
- Gerenciamento
- Gerência de Configuração e Mudanças
- Manutenção

- Artefatos

- Plano de Negócios
- Plano de Projeto
- Plano de Riscos
- Documento de Requisitos
- Mapeamentos A&P
- Documento de Caso de Uso
- Documento de Arquitetura
- Classes
- Documento de Testes
- Documento de Validação
- Manual do Sistema

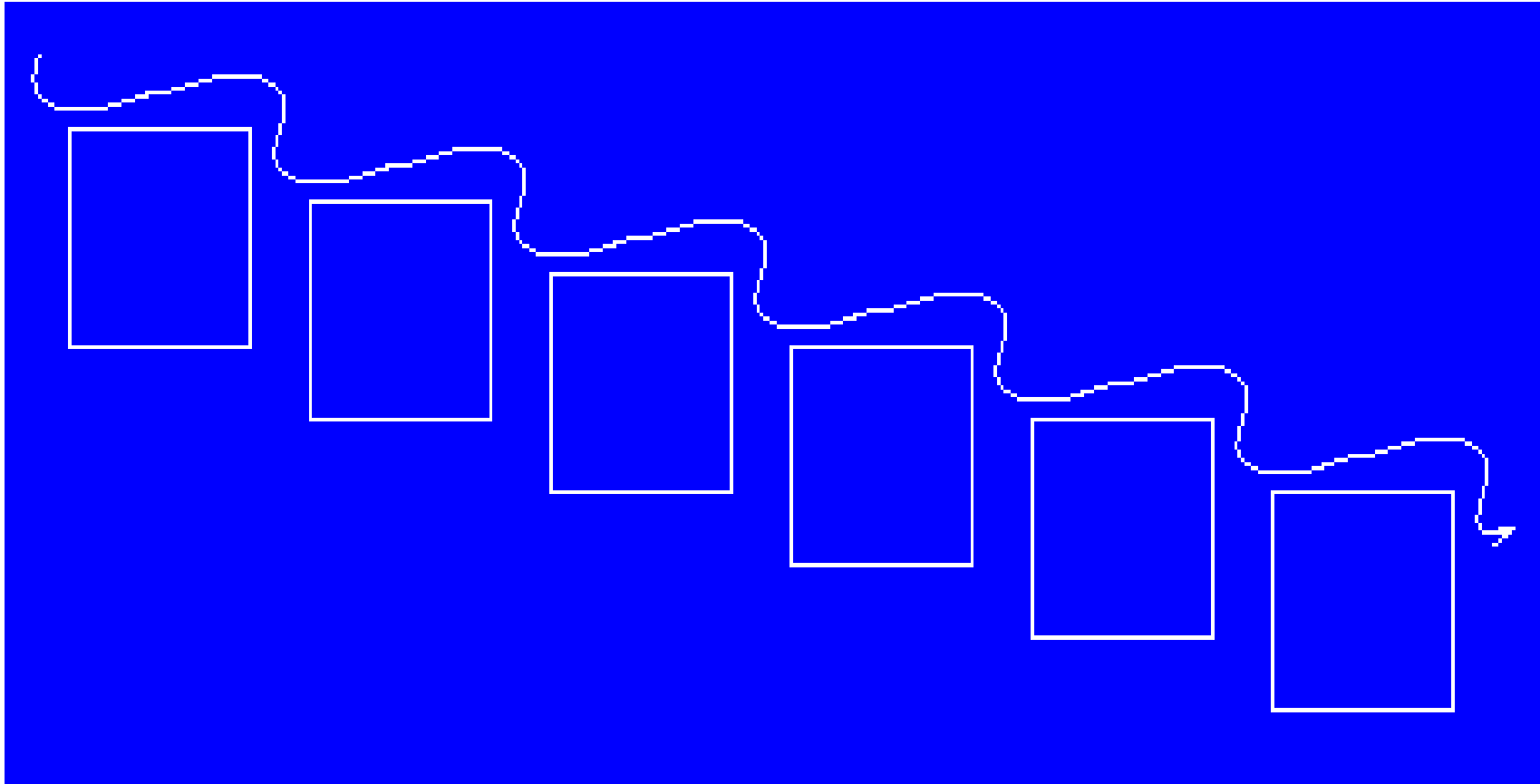
# O que é um modelo de ciclo de vida de processo de software?

- Uma representação abstrata e simplificada do processo de desenvolvimento software, tipicamente mostrando as principais atividades e dados usados na produção e manutenção de software

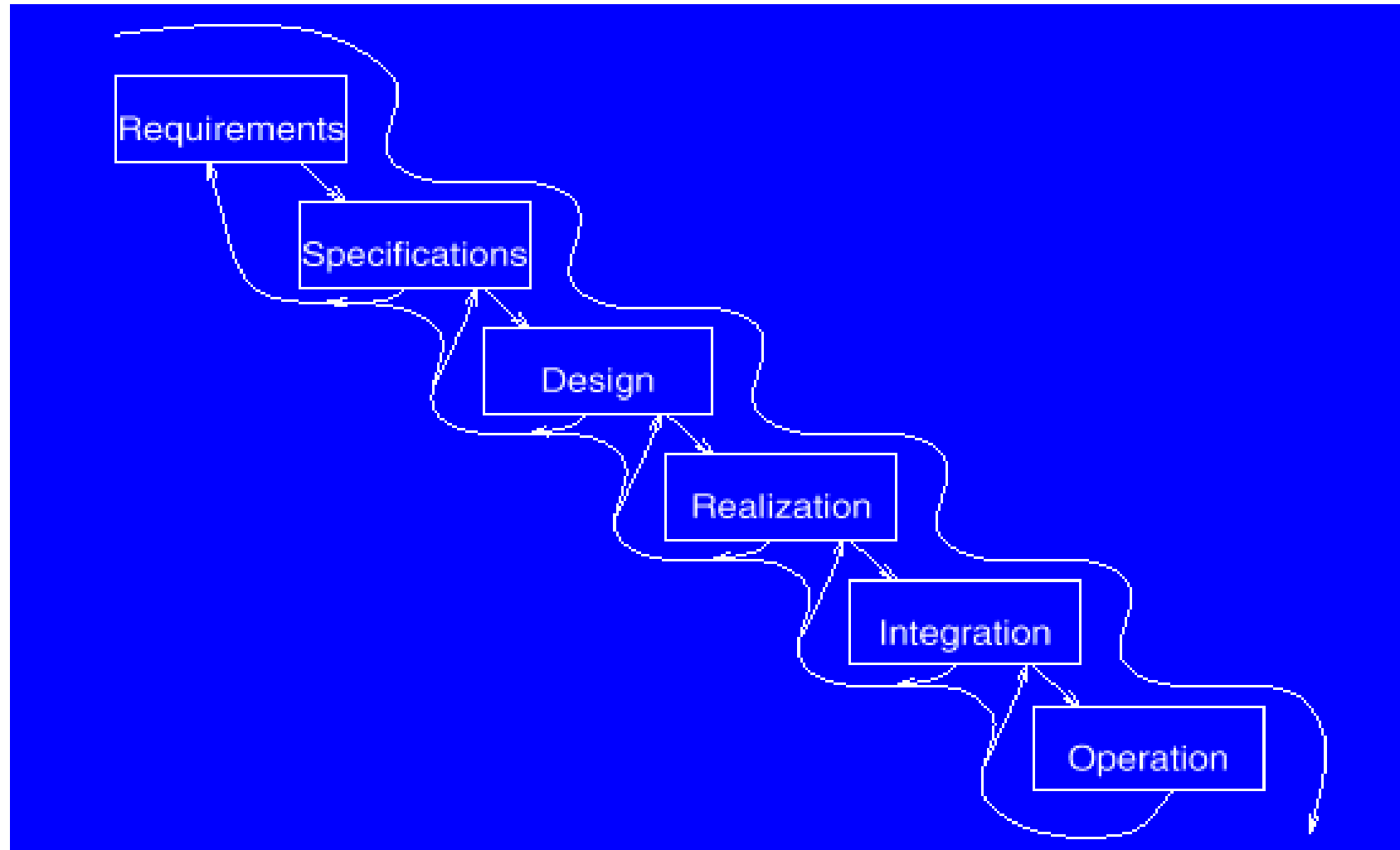
# Modelos do Ciclo de Vida de Software

- Cascata
- Modelos Iterativos
  - Espiral
  - Incremental
- ...

# Modelo Cascata



# Modelo Cascata na Prática



# Modelos Iterativos

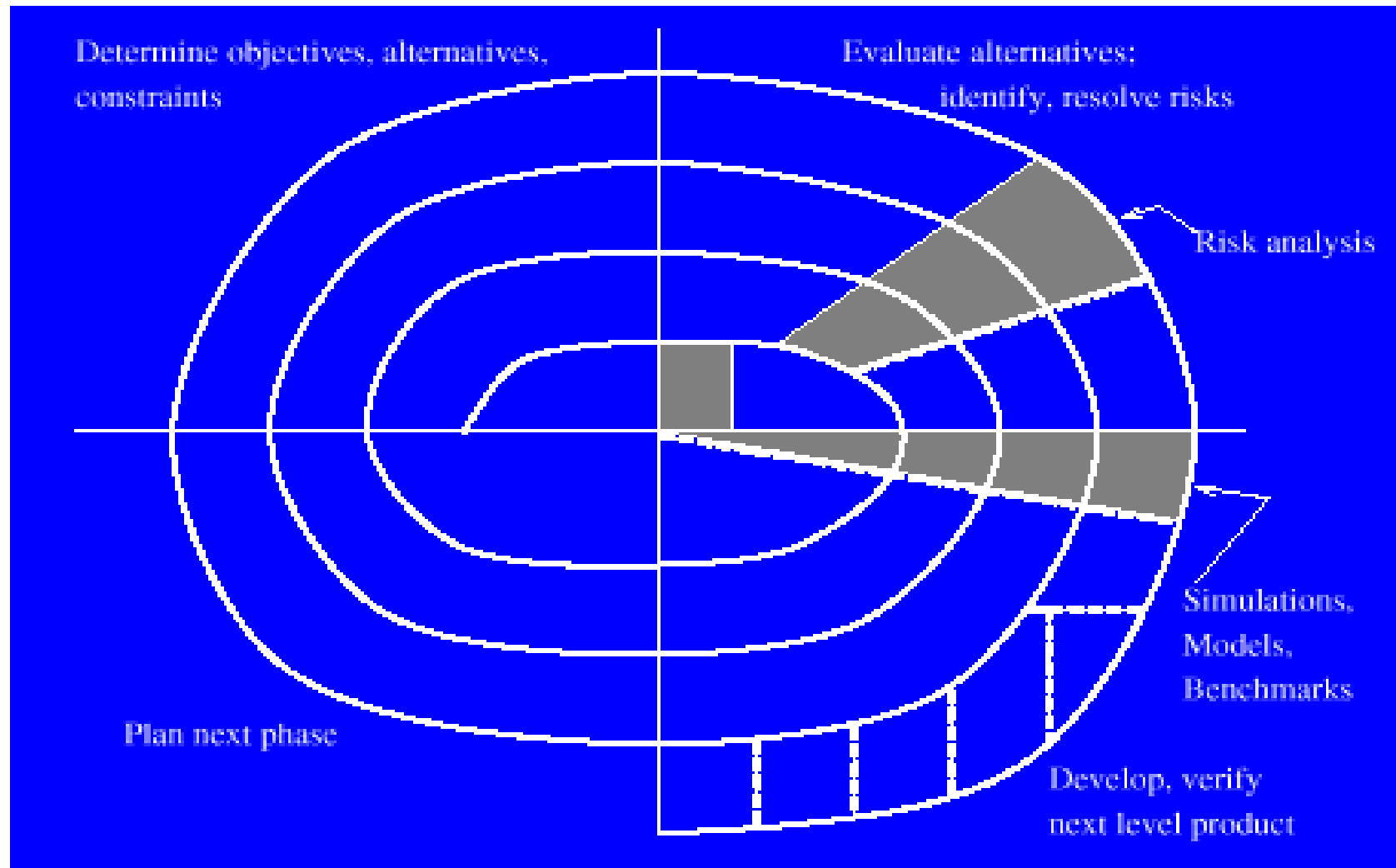
- Requisitos de sistema SEMPRE evoluem durante curso de um projeto. Assim a iteração do processo sempre faz parte do desenvolvimento de grandes sistemas
- Iterações podem ser aplicadas a quaisquer dos modelos de ciclo de vida
- Duas abordagens (relacionadas)
  - Desenvolvimento espiral
  - Desenvolvimento incremental



# Desenvolvimento Espiral

- Acrescenta aspectos gerenciais ao processo de desenvolvimento de software.
  - análise de riscos em intervalos regulares do processo de desenvolvimento de software
  - planejamento
  - controle
  - tomada de decisão
- O processo é representado como uma espiral em vez de uma seqüência de atividades
- Cada volta na espiral representa uma fase no processo
- Não há fases fixas como especificação ou projeto - voltas na espiral são escolhidas dependendo do que é requerido
- Riscos são avaliados explicitamente e resolvidos ao longo do processo

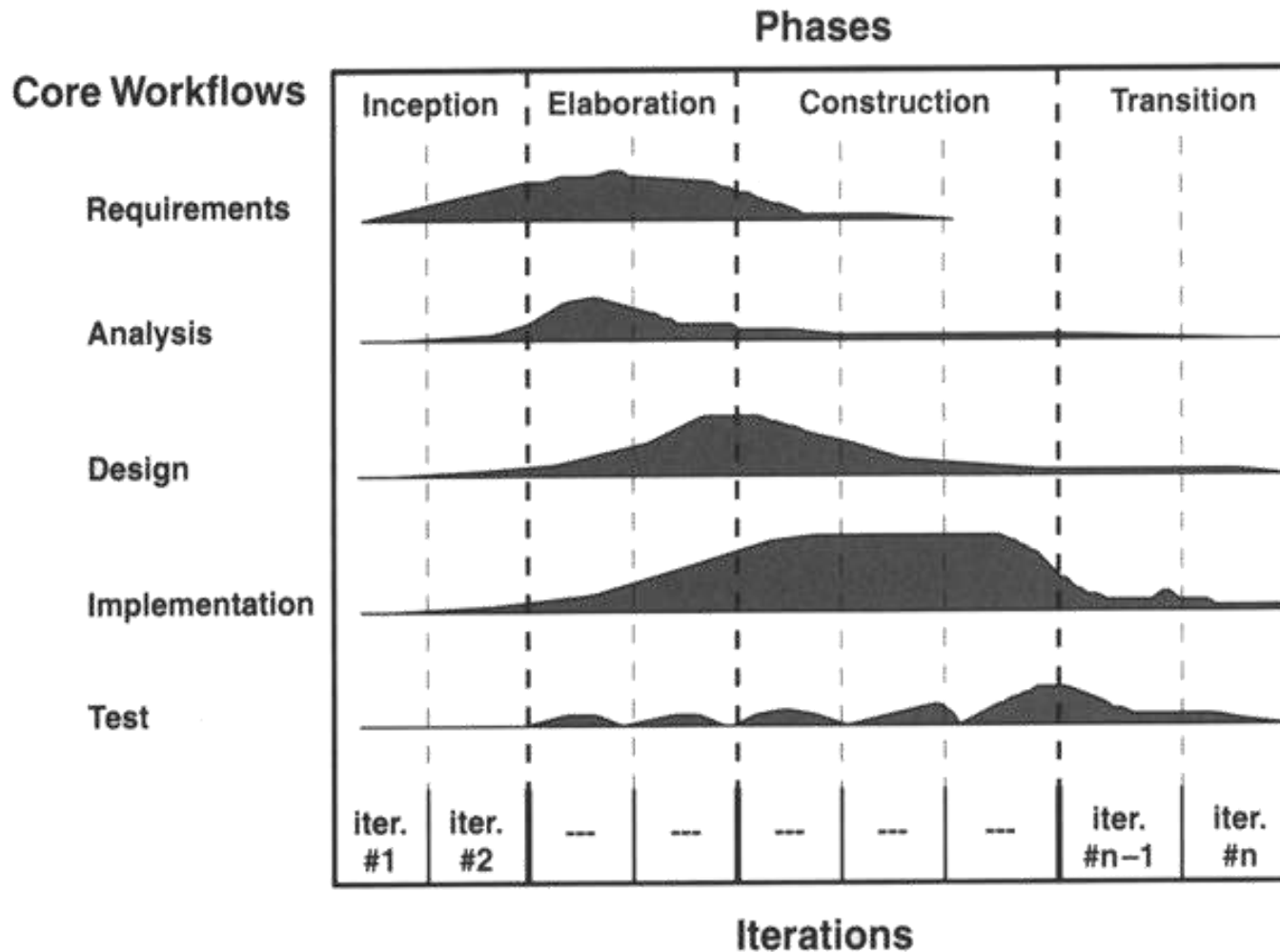
# Desenvolvimento Espiral



# Desenvolvimento Incremental

- Em vez de entregar o sistema como um todo, o desenvolvimento e a entrega são divididos em incrementos, com cada incremento entregando parte da funcionalidade requerida
- Requisitos dos usuários são priorizados e os requisitos de mais alta prioridade são incluídos nas iterações iniciais
- Uma vez que o desenvolvimento de um incremento é iniciado, os requisitos são "congelados". Embora os requisitos possam continuar a evoluir para incrementos posteriores

# Desenvolvimento Iterativo e Incremental (do RUP)



# Desenvolvimento Iterativo e Incremental (do RUP)

- Concepção: ênfase no escopo do sistema;
- Elaboração: ênfase na arquitetura;
- Construção: ênfase no desenvolvimento;
- Transição: ênfase na implantação.

# Linguagem

- Notação com sintaxe e semântica bem definidas
  - com representação gráfica ou textual
- Usada para descrever os artefatos gerados durante o desenvolvimento de software
- Exemplos: UML, Java

# Método

- Descrição sistemática de como deve-se realizar uma determinada atividade ou tarefa
- A descrição é normalmente feita através de padrões e guias

# Ferramenta CASE

- Provê suporte computacional a um determinado método ou linguagem
- Ambiente de desenvolvimento: conjunto de ferramentas integradas (CASE)
- Exemplos: Astah, Eclipse



# Processo

- Conjunto de atividades
  - bem definidas
  - com responsáveis
  - com artefatos de entrada e saída
  - com dependências entre as mesmas e ordem de execução
  - com modelo de ciclo de vida

# Processo de software

- Um conjunto de atividades cujo objetivo é o desenvolvimento ou a evolução do software
- Conjunto coerente de atividades para especificação, projeto, implementação e teste de sistemas de software

# Metodologia

- Conjunto de métodos + processo

# Pontos principais

- Engenharia de software é uma disciplina de engenharia que está envolvida com todos os aspectos da produção de software
- Produtos de software consistem de programas desenvolvidos e documentação associada. Alguns atributos de qualidade do produto são manutenibilidade, eficiência e usabilidade
- O processo de software consiste nas atividades que são envolvidas no desenvolvimento de produtos de software

# Pontos principais

- Métodos são formas organizadas de produzir software. Eles incluem sugestões para o processo a ser seguido, as notações a serem usadas, regras que governam as descrições do sistema que são produzidas e diretrizes de projeto
- Ferramentas CASE são sistemas de software que são projetados para suportar as atividades rotineiras no processo de software, como edição de diagramas de projeto e verificação de consistência dos diagramas

# Cenas do próximo capítulo

- Metodologias existentes
  - Workshops
  - Vamos organizar?!