# ENGENHARIA DE SOFTWARE

Prof. Tiago M. Calixto

# OBJETIVO DA AULA

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

# 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

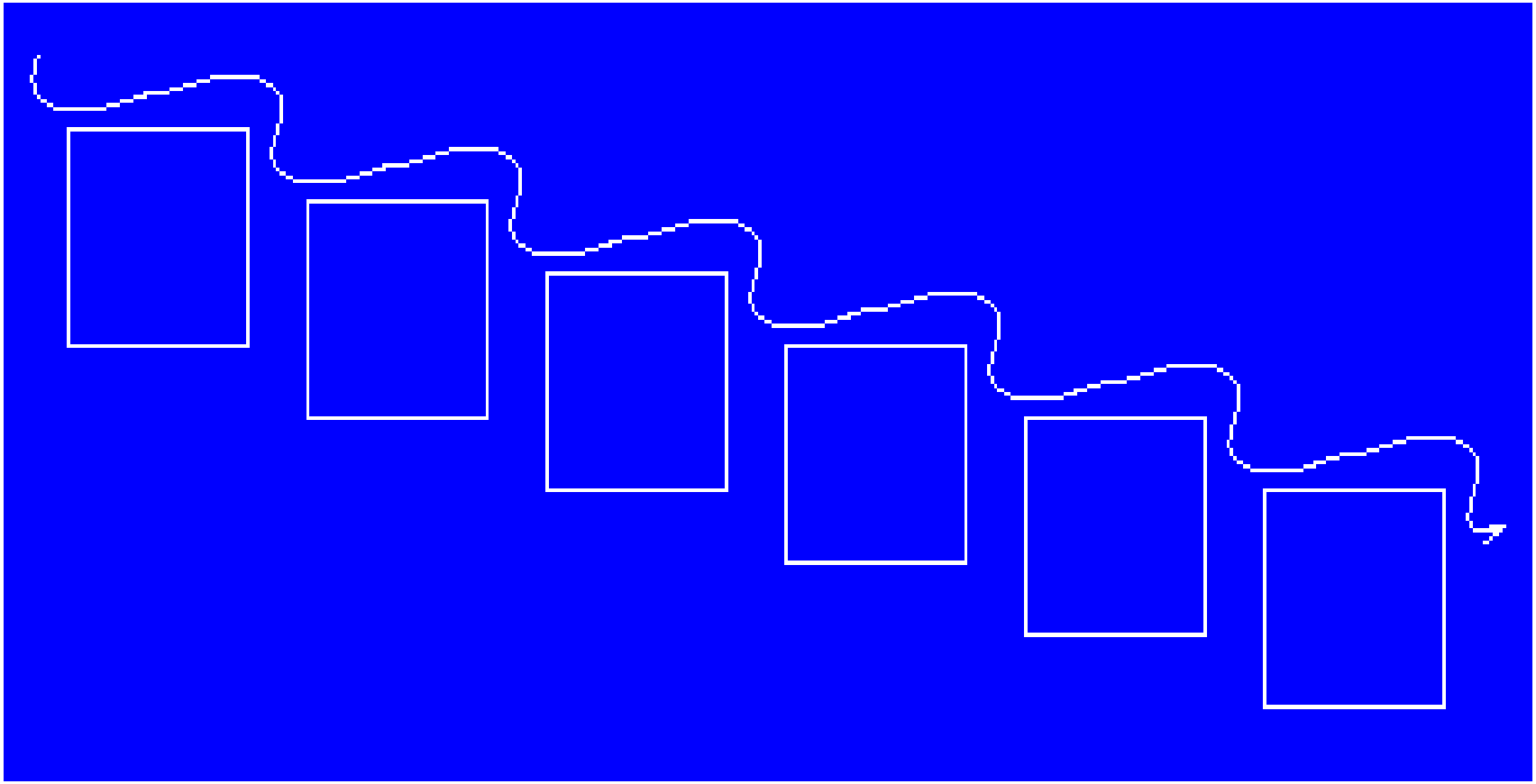
# O QUE É UM MODELO DE CICLO DE VIDA DE PROCESSO DE SOFTWARE?

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

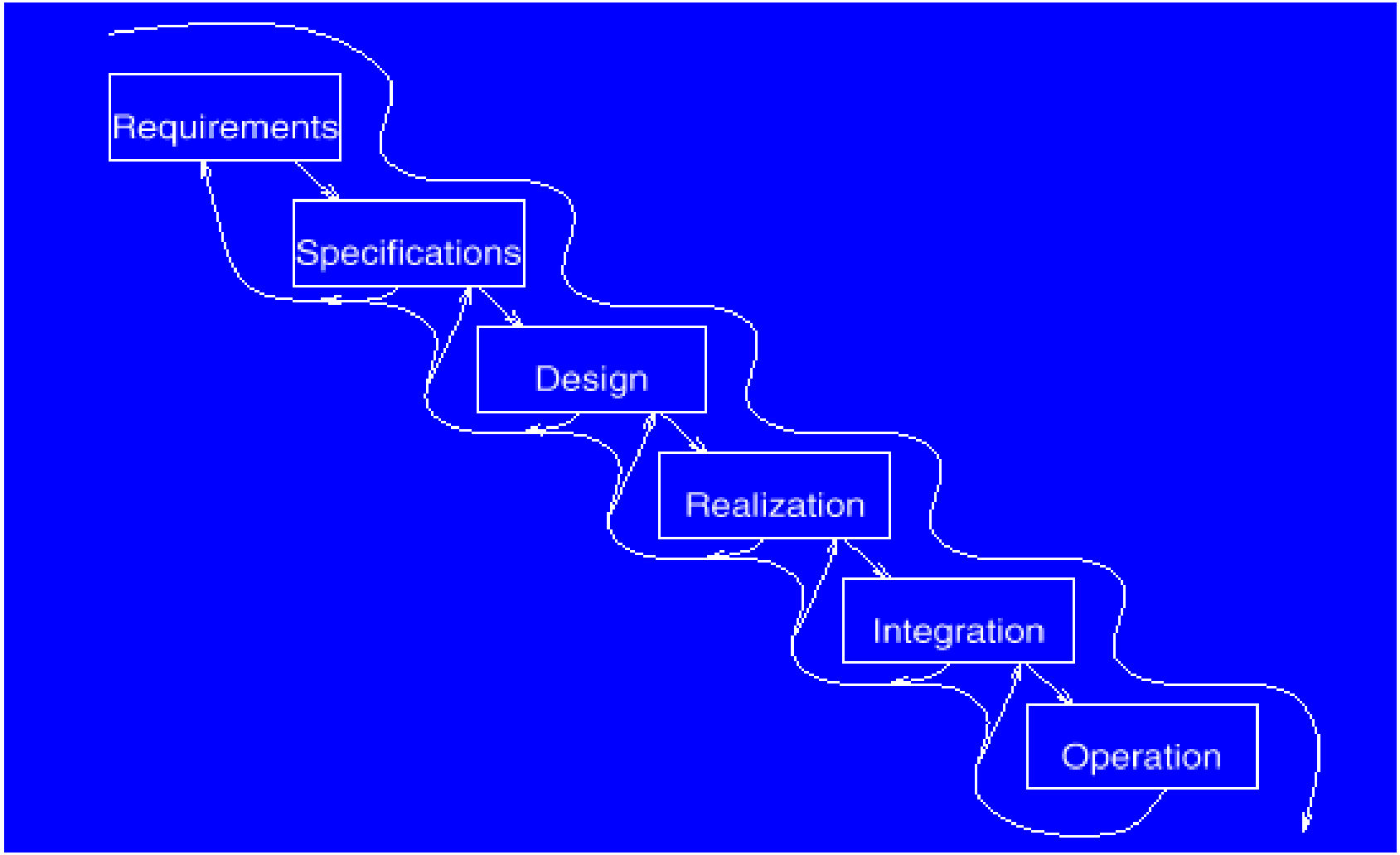
# Modelos do ciclo de vida de software

- Cascata
- Modelos Iterativos
  - Espiral
  - Incremental (ex: do RUP)
- ...

# MODELO CASCATA



# MODELO CASCATA NA PRÁTICA



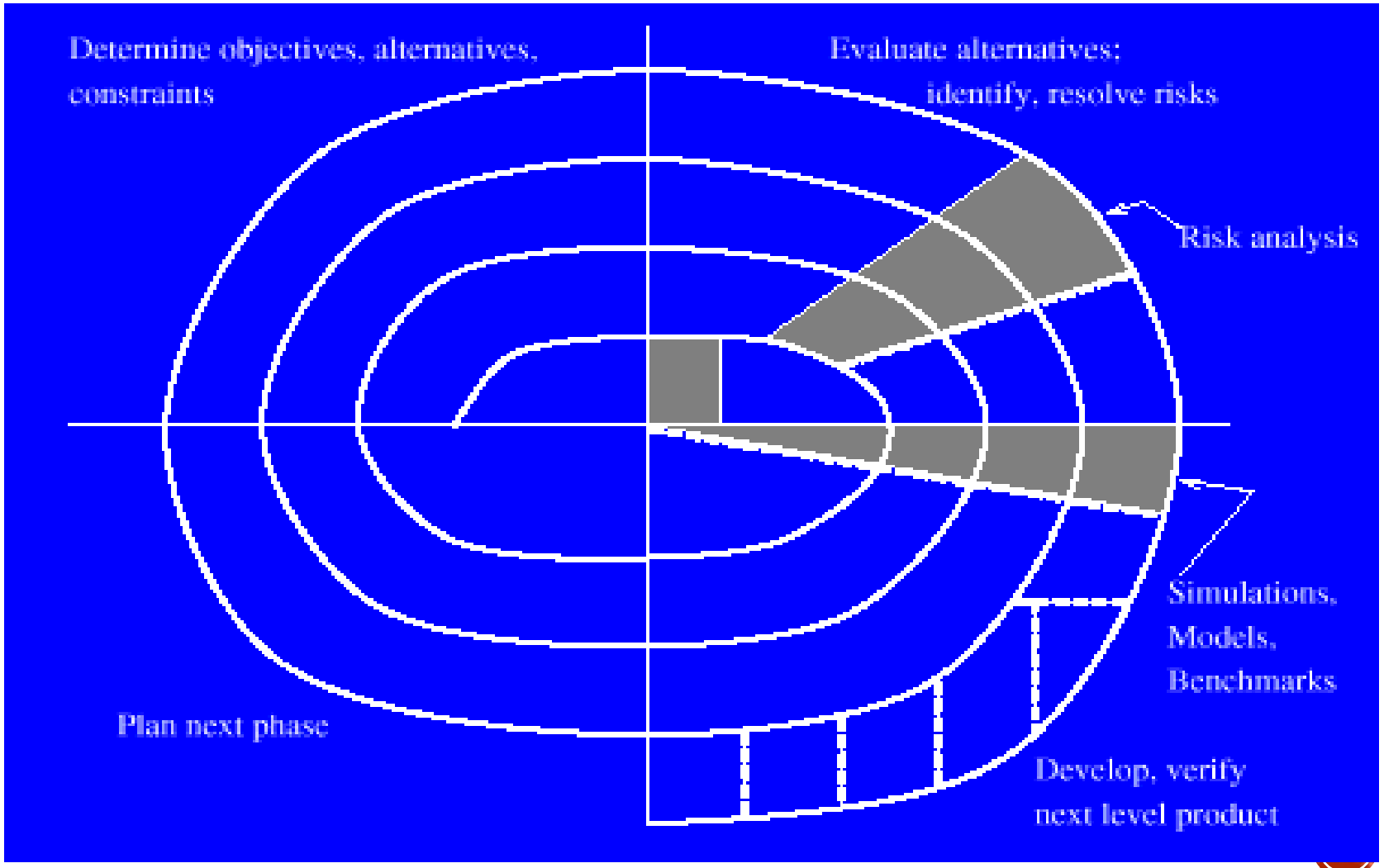
# MODELOS ITERATIVOS

- 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 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

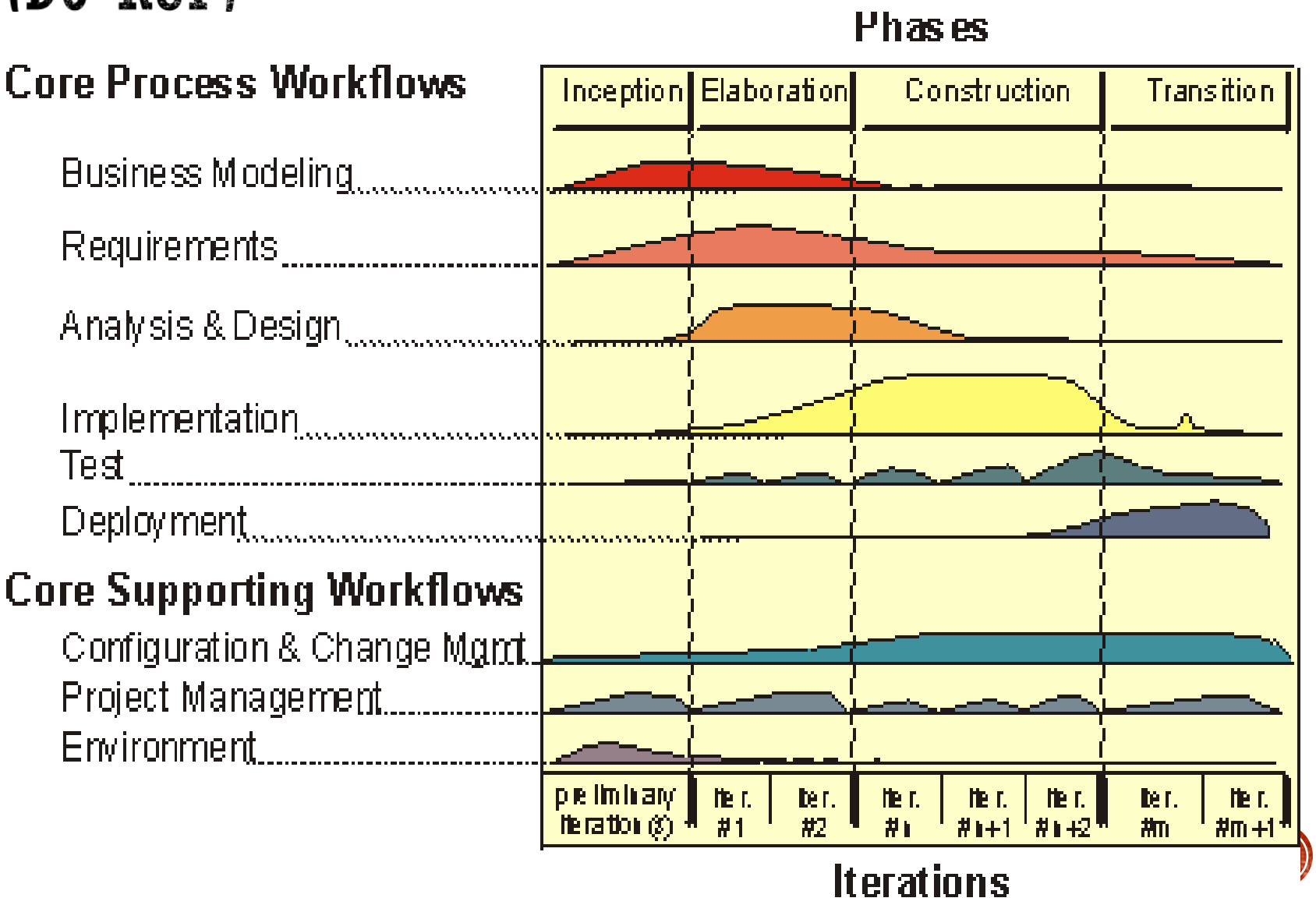
# DESENVOLVIMENTO ESPIRAL



# 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)



# 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
- Exemplos: Método para descoberta das classes de análise no RUP.

# FERRAMENTA CASE

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

# 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

# LEITURA ADICIONAL

- Daniel M. Berry. Myths and Realities in Software Development.
- W. Wayt Gibbs. Software's chronic crisis. Scientific American, September 1994.
- Alan Joch. How software doesn't work. Byte, December 1995.

# E DEPOIS DE TUDO...

# MOMENTO RELAX

- Mensagem Subliminar
  - Quantidade de informação dividida pelo tempo/Espaço de Exposição.

# MENSAGEM SUBLIMINAR

M