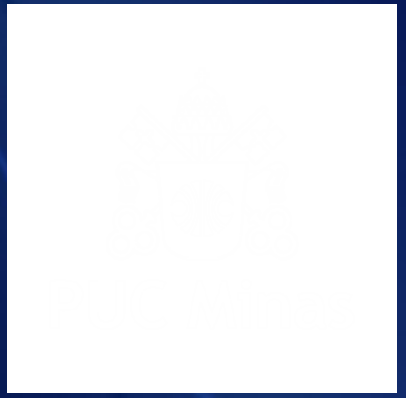
Processos de Software

Curso de Ciência da Computação
Disciplina de Engenharia de Software I
Prof. Humberto Torres Marques Neto

Agosto de 2018



Objetivos

- Apresentar e discutir a evolução histórica dos modelos de processos de software
- Discutir os problemas relacionados ao uso de processos de software para propor soluções

Referências Bibliográficas

Básica:
PRESSMAN, Roger S. Engenharia de Software: uma abordagem profissional. 8 ed. McGraw-Hill, 2016.

Complementar:
SOMMERVILLE, Ian. Engenharia de software. 9. ed. São Paulo: Pearson Prentice Hall, 2011.

O que é um Processo?

Um processo é uma sequência de passos realizados com um propósito. Ou seja, processo é o que você faz. O processo integra pessoas, ferramentas e procedimentos. Processo é o que pessoas fazem, utilizando procedimentos, métodos, ferramentas, e equipamentos, para transformar matéria prima (*inputs*) em um produto (*outputs*) de valor para os seus clientes.” (PAULK, Mark C. The capability maturity model: guidelines for improving the software process. Boston: Addison Wesley, 2003.)

O que é um Processo de Desenvolvimento de Software?

"Um processo de desenvolvimento de software pode ser definido como um conjunto de atividades, métodos, práticas e transformações que pessoas utilizam para desenvolver ou dar manutenção em softwares ou em seus produtos associados (projetos, manuais, código, etc.)" (PAULK, Mark C. The capability maturity model: guidelines for improving the software process. Boston: Addison Wesley, 2003.)

"… *processo de software é definido* como uma metodologia para as atividades, ações e tarefas necessárias para desenvolver um software de alta qualidade." (PRESSMAN, 2011. p. 51)

Processo é sinônimo de Engenharia de Software?

"Um processo de software define a abordagem adotada conforme um software é elaborado pela engenharia. Mas, a engenharia de software também engloba tecnologias que fazem parte do processo – métodos técnicos e ferramentas automatizadas." (PRESSMAN, 2011. p. 51)

Ou seja, é resposta é "sim e não"!

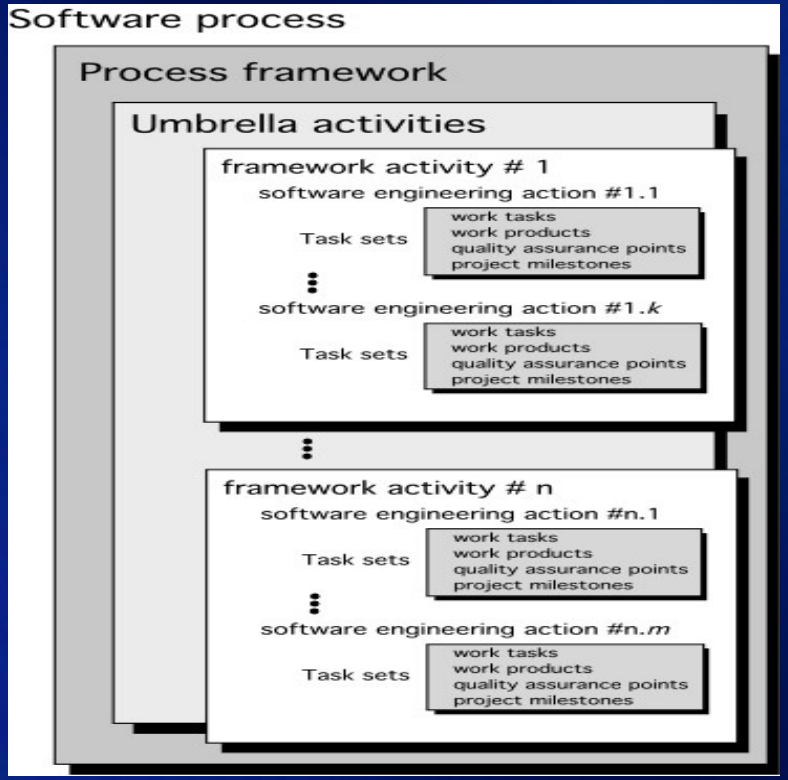
Modelo de Processo Genérico (1/2)

- Atividades metodológicas
  - Comunicação
  - Planejamento
  - Modelagem
  - Construção
  - Entrega

Modelo de Processo Genérico (2/2)

- Atividades de apoio
  - Controle e acompanhamento do projeto
  - Administração de riscos
  - Garantia de qualidade
  - Revisões técnicas
  - Medição
  - Gerenciamento de configuração de software
  - Gerenciamento da reusuabilidade
  - Preparo e produção de artefatos

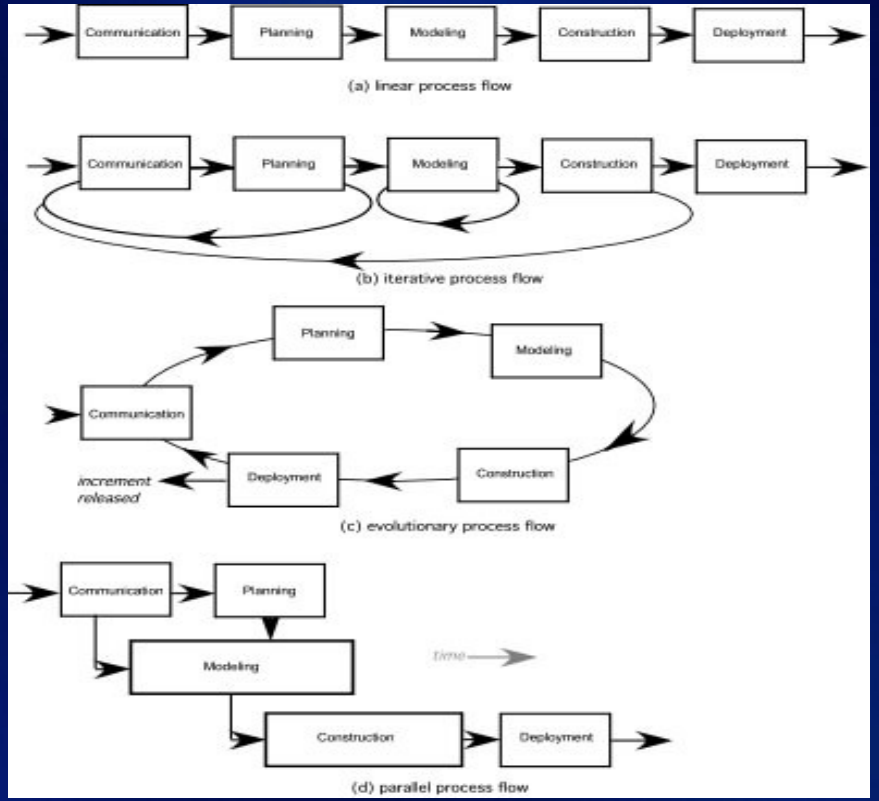
Modelo de Processo Genérico



PRESSMAN, 2011, p.53

Fluxo de Processo

- Linear
- Iterativo
- Evolucionário
- Paralelo



PRESSMAN, 2011, p.54

Modelo de Processo Genérico

- As atividades do processo devem ser definidas, basicamente, de acordo com a complexidade e com os evolvidos no projeto.
- Ou seja, projetos diferentes demandam conjuntos de tarefas diferentes. A equipe de software escolhe o conjunto de tarefas fundamentada no problema e nas características do projeto.

Padrões de Processos

- Descreve um problema relacionado ao processo que pode ser encontrado em um trabalho de engenharia de software
- Identifica o ambiente onde problema é normalmente encontrado
- Sugere soluções para o problema

Tipos de Padrões de Processos (1/3)

- Padrão de estágio
  - Define um problema associado a uma atividade metodológica
  - *Exemplo*: Estabelecendo Comunicação

Tipos de Padrões de Processos (2/3)

- Padrão de tarefas
  - Define um problema associado a uma ação de engenharia de software ou tarefa de trabalho relevante para a sua prática
  - *Exemplo*: Levantamento de Necessidades

Tipos de Padrões de Processos (3/3)

- Padrão de fases
  - Define a sequência das atividades metodológicas que ocorrem dentro do processo
  - *Exemplo*: Modelo Espiral ou Prototipação

Veja um exemplo na página 57 do livro do PRESSMAN, 2011

Avaliação e Aperfeiçoamento de Processos

- SCAMPI (*Standard CMMI Assesment Method for Process Improvement*)
- CBA IPI (*CMM – Based appraisal for Internal Processo Impovement*)
- SPICE (ISO/IEC15504)
- ISO 9001:2000 para Software

Modelos de Processo Prescritivo (1/2)

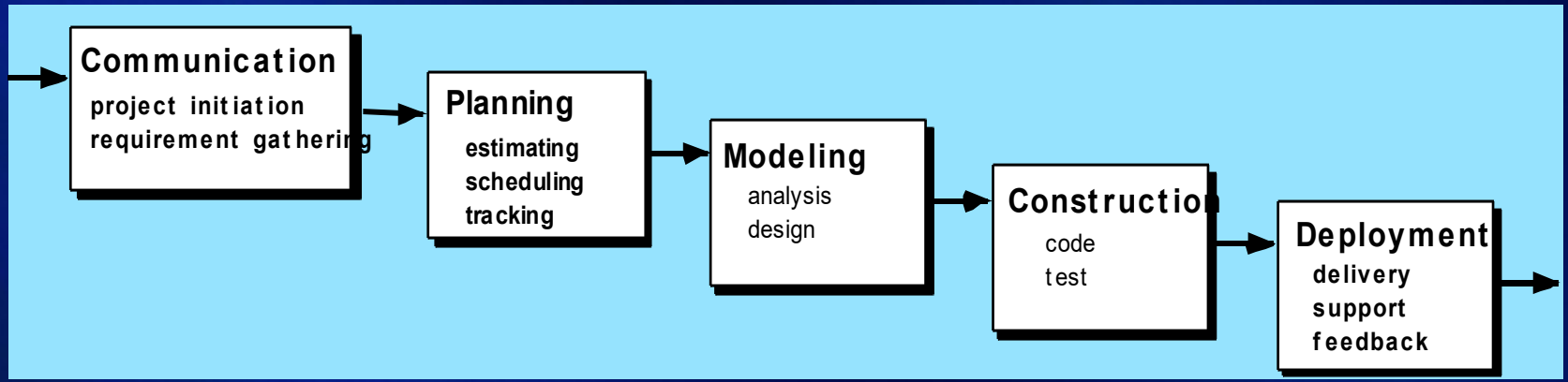
- Esses modelos defendem uma abordagem ordenada para a engenharia de software
- Tentam trazer ordem ao "caos" (*code and fix*)
- Prescrevem um conjunto de elementos do processo
- Também são conhecidos como *modelos tradicionais*

Modelos de Processo Prescritivo (2/2)

- Modelo em cascata
- Modelos de processo incremental
- Modelos de processo evolucionário
  - Prototipação
  - Modelo espiral
- Modelos concorrentes

Modelo em cascata (*Waterfall Model*)

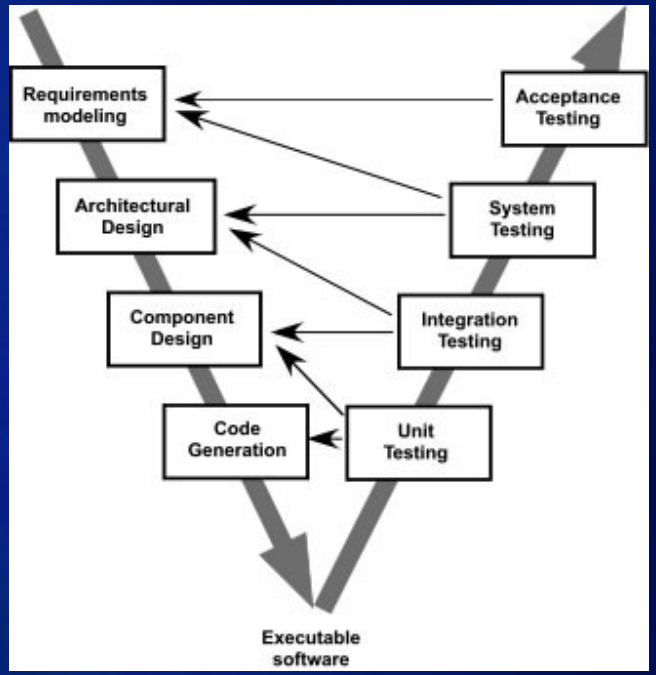
- Também conhecido como *Ciclo de Vida Clássico*



PRESSMAN, 2011, p.60

Modelo V

- Variação na representação do modelo cascata



PRESSMAN, 2011, p.60

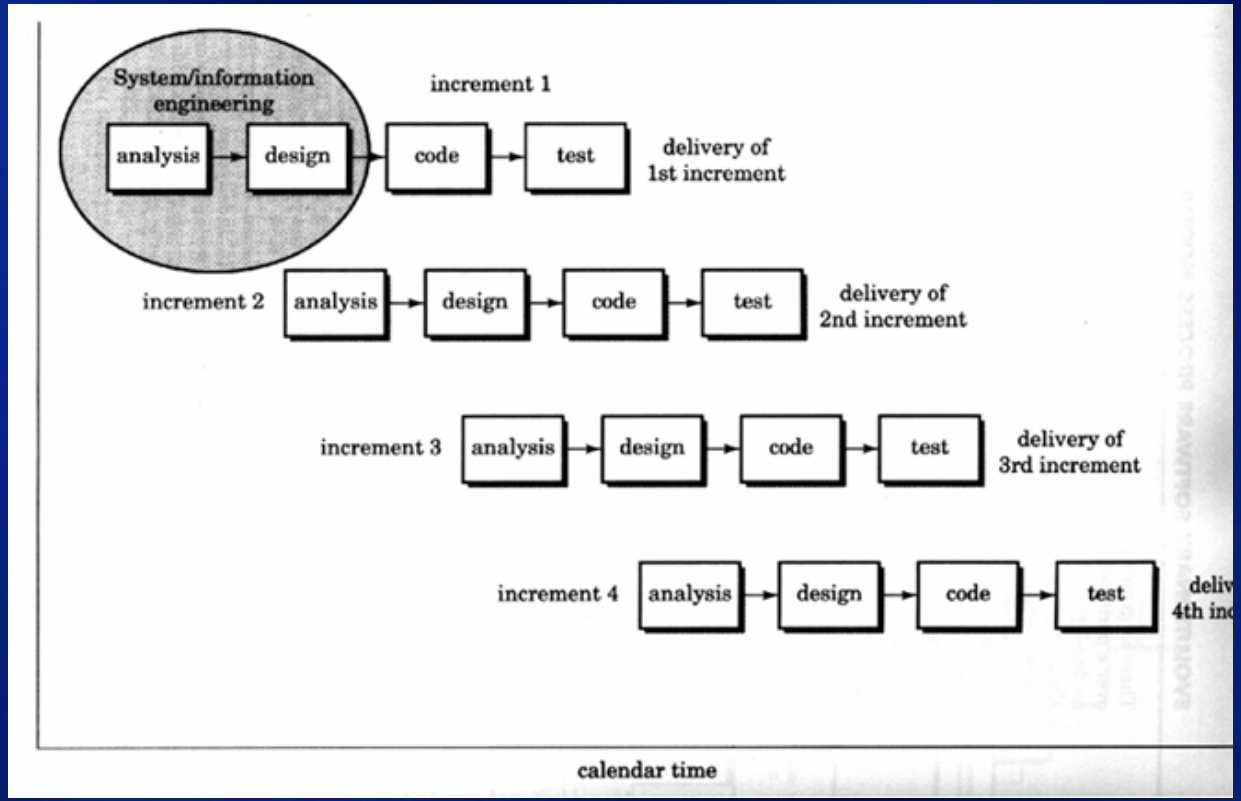
Modelo em cascata: alguns problemas

- Adia a identificação de riscos, tornando extremamente caro desfazer algum erro cometido nas fases iniciais
- Atrasa e agrupa os testes do sistema, o que torna a integração um processo difícil
- Torna difícil a entrega parcial de produtos
- Difícil de lidar com mudanças de requisitos

Modelos de processo incremental

- Combina elementos dos fluxos de processos lineares e paralelos
- Cada sequência linear produz um incremento executável do software
- Cada incremento é um núcleo, uma base para o próximo → *produto essencial*

Modelos de processo incremental



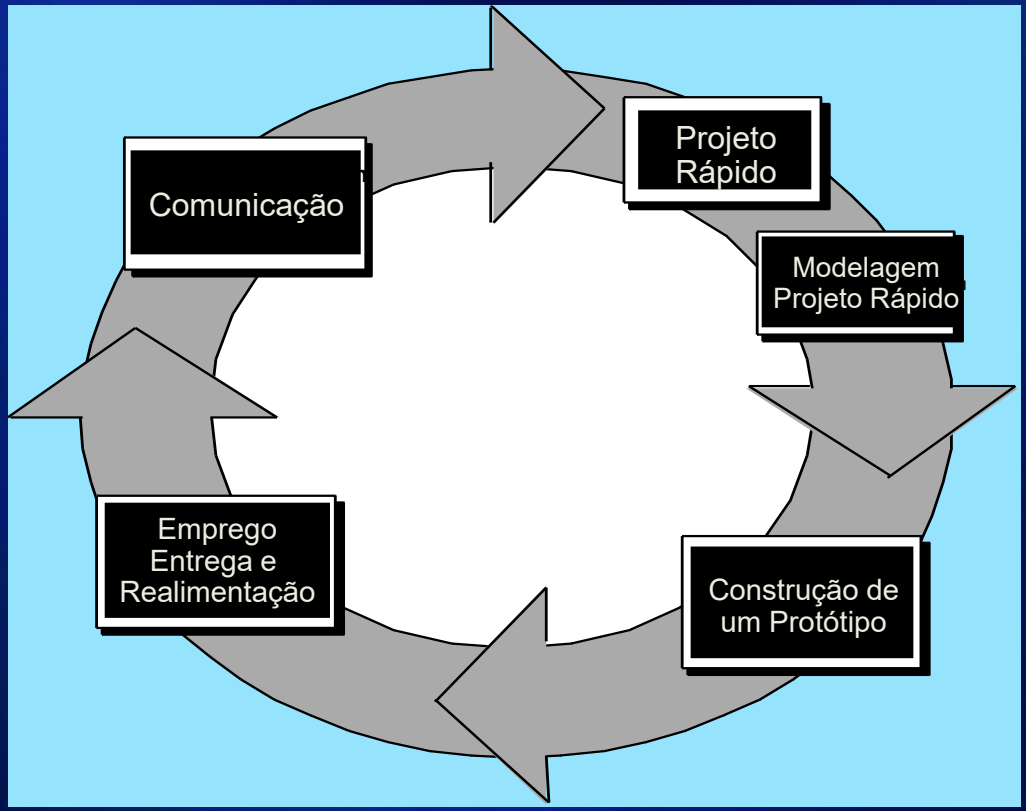
Modelos de processo incremental

- *Exemplo 1*: um editor de texto que é implementado só com as funções básicas em sua primeira versão
- *Exemplo 2*: um Sistema de Gestão Empresarial para substituição de um Sistema Legado

Modelos de processo evolucionário

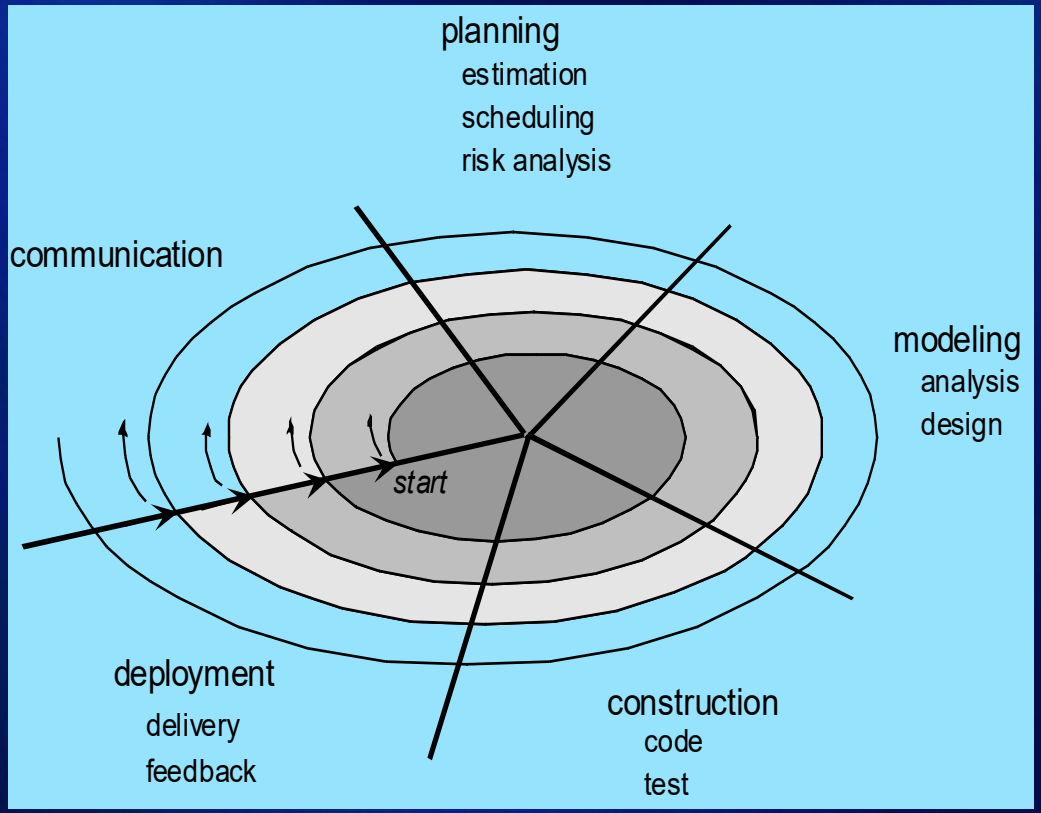
- Modelos evolucionários são iterativos ou incrementais
- Apresentam características que possibilitam desenvolver versões cada vez mais completas do software
- Modelos comuns: Prototipação e Modelo Espiral

Prototipação



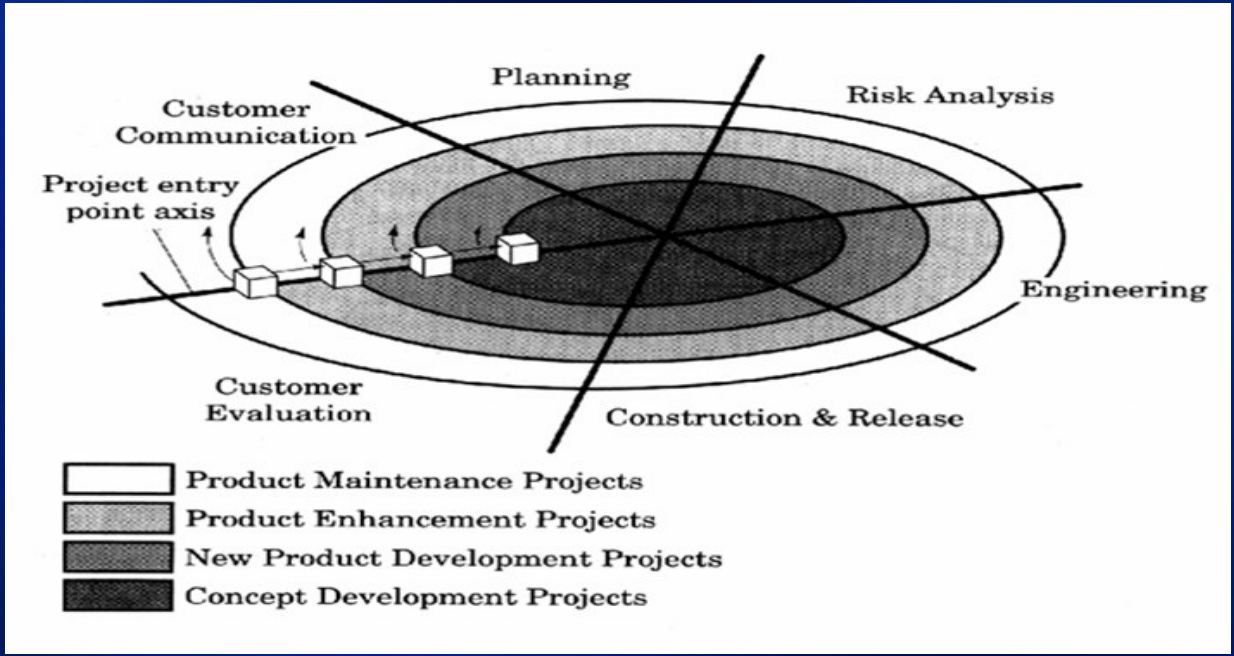
PRESSMAN, 2011, p.63

Modelo Espiral



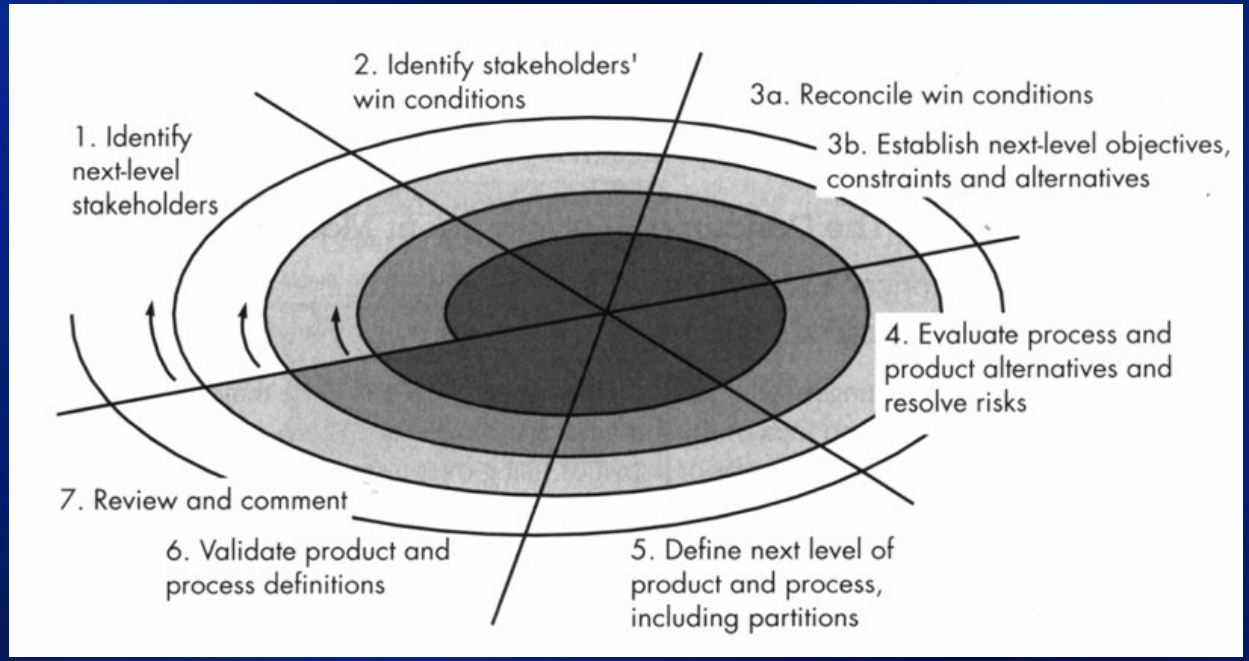
PRESSMAN, 2011, p.65

Modelo Espiral (outra versão)



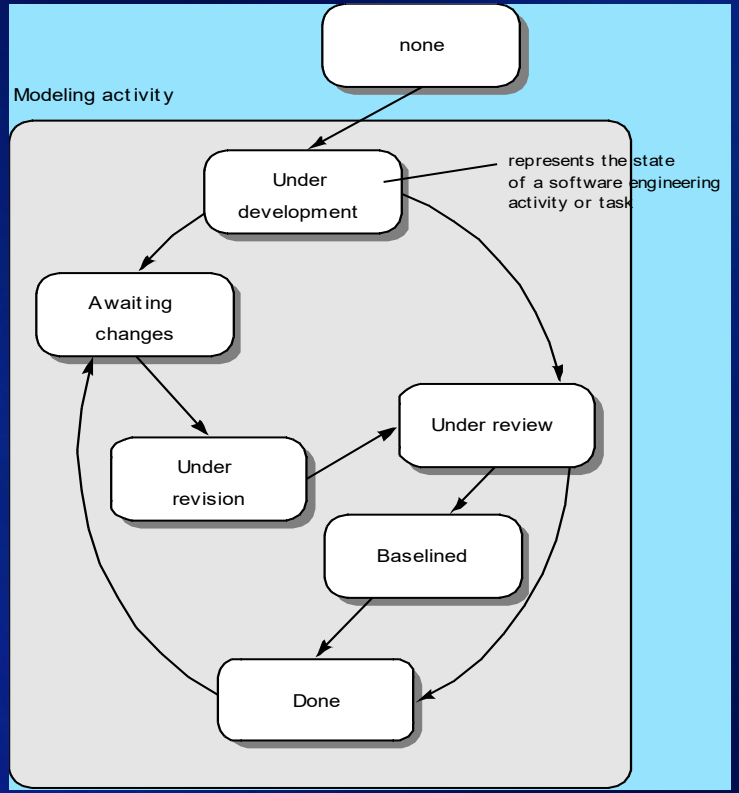
PRESSMAN, 2001, p.37

Modelo Espiral “WinWin”



PRESSMAN, 2001, p.39

Modelos Concorrentes



PRESSMAN, 2011, p.67

Modelo Baseado em Componentes

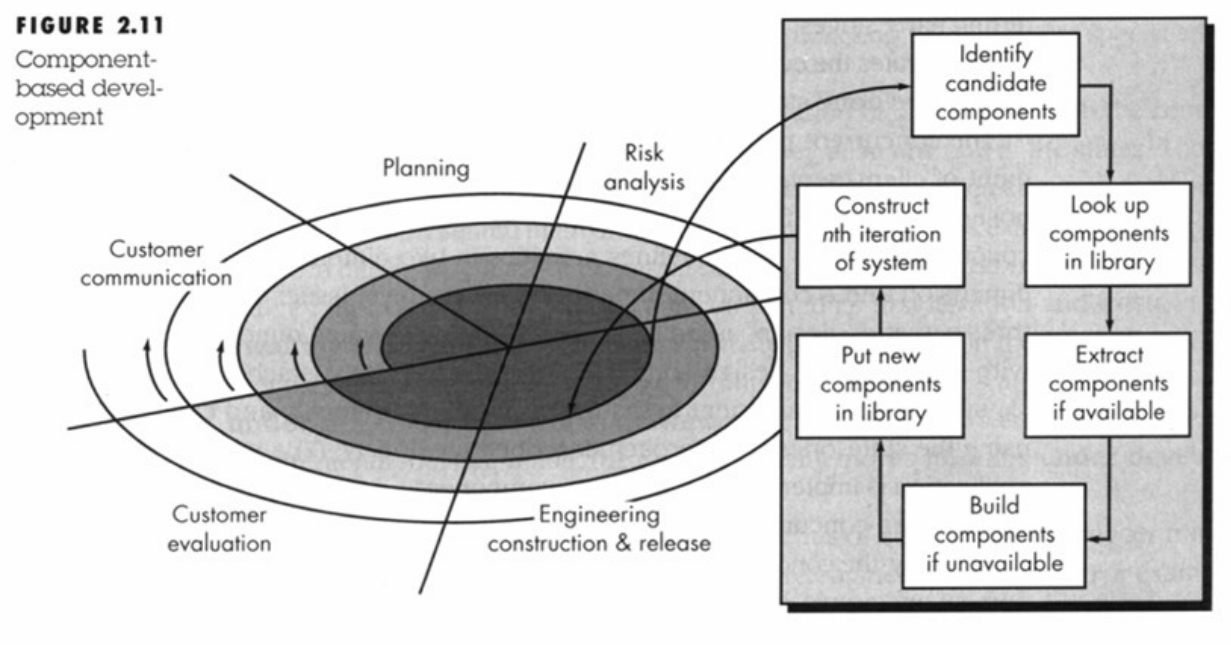


FIGURE 2.11
Component-based development

PRESSMAN, 2001, p.42

Processo WebE

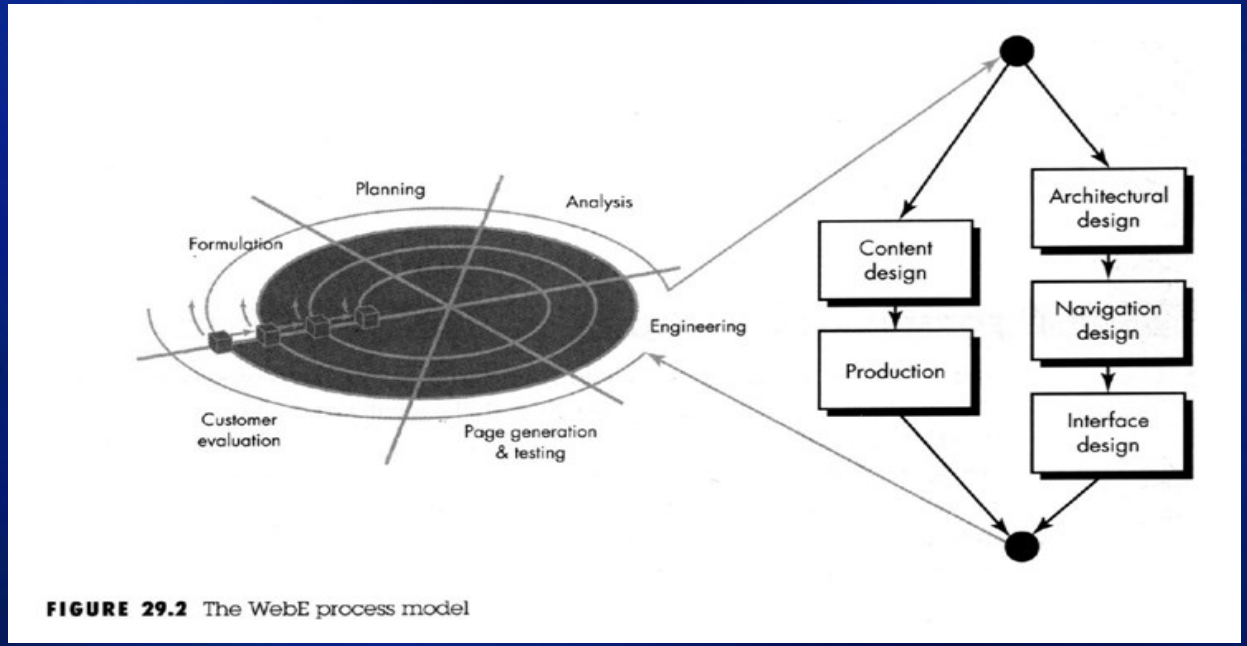


FIGURE 29.2 The WebE process model

PRESSMAN, 2001, p.775

Modelos de Processo Especializado

- Desenvolvimento baseado em componentes
  - Importância do reuso de software
- Modelo de métodos formais
  - Ênfase na especificação matemática dos requisitos de software

Processo de Software Pessoal (PSP) - 1/2

- Proposto por Watts Humphrey
- O processo de software deve se adequar às pessoas e não o contrário!
- O PSP (*Personal Software Process*) enfatiza a importância de se aprender com os erros
- É um processo desafiador e exige um nível de comprometimento bem alto

Processo de Software Pessoal (PSP) - 2/2

- O PSP (*Personal Software Process*) possui cinco atividades estruturais:
  - Planejamento
  - Projeto de alto nível
  - Revisão de projeto de alto nível
  - Desenvolvimento
  - Autópsia (análise *postmortem*)

Processo de Software em Equipe (TSP)

- Também proposto pelo Humphrey
- Baseado em lições aprendidas com o PSP
- O objetivo do TSP (*Team Software Process*) é criar equipes “autodirigidas” para produzir software de alta qualidade

Objetivos do TSP (1/2)

- Promover a criação de equipes autodirigidas que planejem e acompanhem seu próprio trabalho, estabeleçam metas e sejam proprietárias de seus processos e planos
- Mostrar aos gerentes como treinar e motivar suas equipes, mantendo alto desempenho

Objetivos do TSP (2/2)

- Acelerar o aperfeiçoamento dos processos de software
- Fornecer orientação para melhorias a organizações com elevado grau de maturidade
- Facilitar o ensino universitário de habilidades de trabalho em equipe de nível industrial

Atividades Metodológicas do TSP

- O TSP define cinco atividades metodológicas:
  - Lançamento do projeto
  - Projeto de alto nível
  - Implementação
  - Integração e testes
  - Autópsia (análise *postmortem*)

Processo de Manutenção de Sistemas (1/2)

- Identificação do problema
- Análise do impacto da alteração demandada pelo problema
- Planejamento da forma de abordagem do problema

Processo de Manutenção de Sistemas (2/2)

- Atualização da documentação existente, levando em consideração a transição de versões
- Implementação das alterações
- Implantação das alterações
- Treinamento

Algumas Formas de Organização de Equipe

- Estrutura centralizada
- Estrutura descentralizada
- Terceirização

Como formar uma equipe motivada para atuar em um processo de software?

Processos e Equipes no Modelo Cascata

- Modelo monolítico (*Analista-Projetista-Desenvolvedor*)
  - Não é escalável
  - Reflete modelo em cascata (waterfall)
  - Requisitos “perfeitos” são transformados em desenhos imutáveis que são implementados em códigos por programadores mecânicos
  - Antítese do modelo iterativo e incremental

Processos e Equipes no Modelo Iterativo

- Modelo hierárquico
  - Arquiteto de Sistemas com visão macro
  - Analista de Sistemas com visão micro
  - Desenvolvedores (engenheiros de aplicação) realizam visão micro

Centro de gravidade de projetos OO bem realizados oscila entre gerente de projeto, arquiteto de sistemas, analistas de sistemas e desenvolvedores

Desafios do Processo de Software (1/2)

- Relacionados às PESSOAS
  - "Inteligência compartilhada"
  - Integração
  - Comunicação
- Relacionados aos PROBLEMAS
  - "Dividir para conquistar"

Desafios do Processo de Software (2/2)

- Relacionados aos PROCESSOS
  - Visão integrada
  - Ciclo de vida do processo de desenvolvimento de sistemas
- Relacionados ao PROJETO
  - Gerenciamento, controle e resultado

Gerenciamento de Riscos

- Como identificar os riscos que envolvem o processo de desenvolvimento de um software?
  - Tamanho do projeto
  - Interferência no negócio
  - Características do cliente / usuário
  - Definição do processo (metodologia)
  - Ambiente de desenvolvimento (tecnologia)
  - Experiência da equipe

Riscos em Processos de Software

Acompanhamento de Projetos de Software

- Como dividir e organizar as atividades que devem ser realizadas?
- Quais as principais ferramentas utilizadas para acompanhar o desenvolvimento de um processo de desenvolvimento de software?

Acompanhamento de Projetos de Software

- Alguns critérios que devem ser considerados para relacionar pessoas às tarefas / atividades
  - Conhecimento da tecnologia
  - Habilidade e experiência para lidar com a tecnologia
  - Grau de dificuldade da tarefa / atividade
  - Interdependência das tarefas / atividades
  - Prazo para realização da tarefa / atividade

Processo Unificado

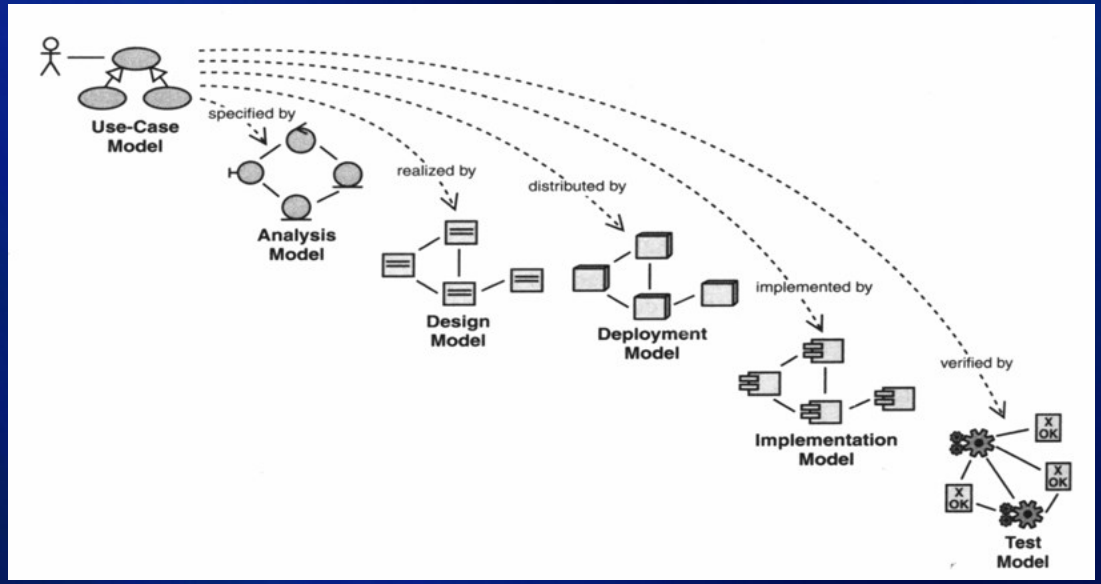
Processo Unificado (UP - *Unified Process*)

JACOBSON, Ivar, BOOCH, Grady, RUMBAUGH, James. The unified software development process. Addison Wesley, 1998.

- Baseado na construção de software a partir de componentes interconectados através de interfaces bem definidas
- Utiliza a UML (*Unified Modeling Language*)

Características chave do UP (1/2)

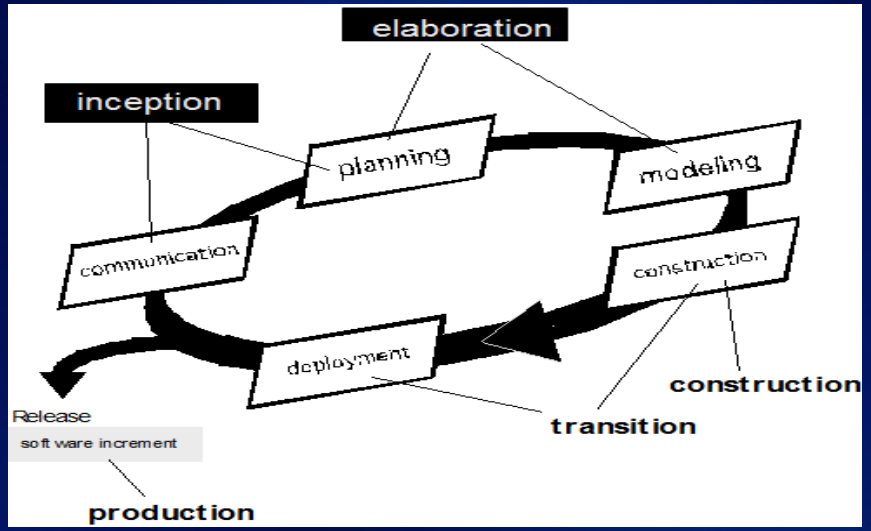
- Dirigido por caso de uso



JACOBSON, 1998, p.10

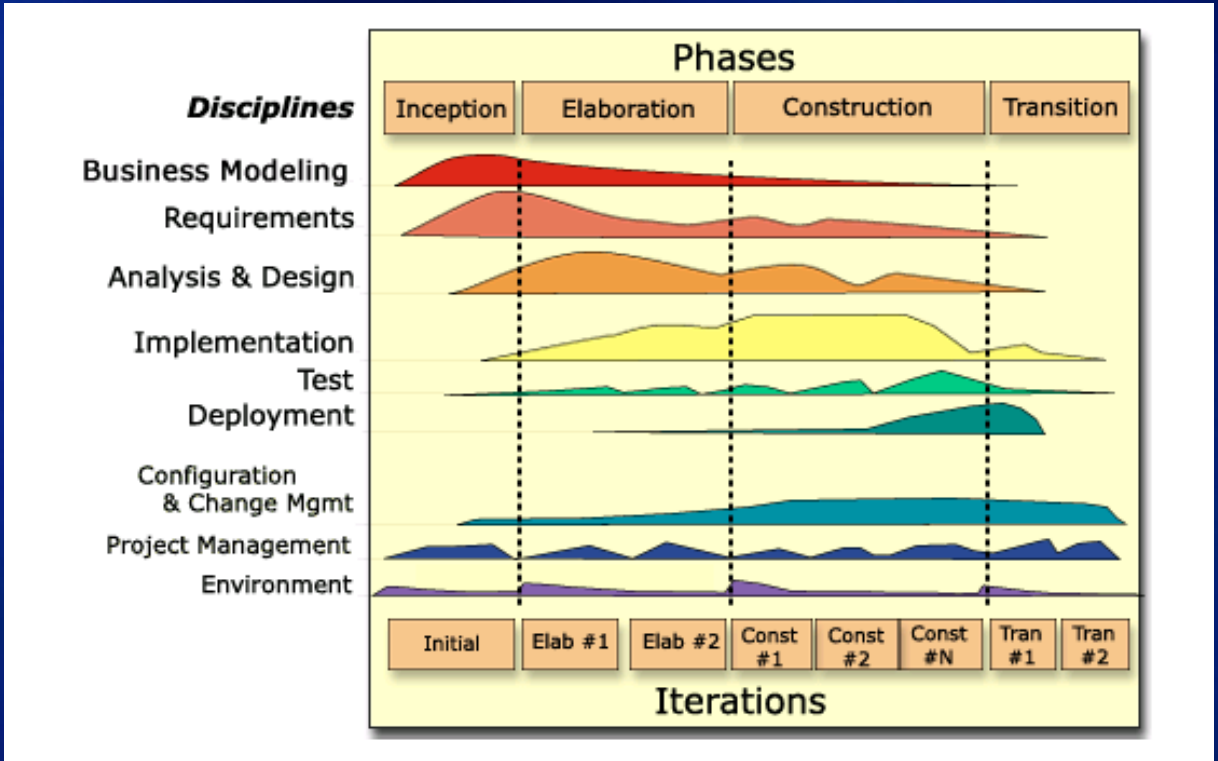
Características chave do UP (2/2)

- Centrado na arquitetura
- Iterativo e incremental



PRESSMAN, 2011, p. 73

RUP (*Rational Unified Process*)



Fases do RUP (1/4)

- Concepção
  - Definição dos casos de uso mais críticos, os quais representam as funções chave do sistema
  - Delimita-se o escopo do produto a ser desenvolvido, identifica-se e reduz-se principalmente os riscos críticos

Fases do RUP (2/4)

- Elaboração
  - Descrição arquitetural do software
  - Procura-se também definir a maioria dos casos de uso, capturando a maioria dos requisitos do software
  - No final desta fase deve-se estar apto a planejar a fase de construção em detalhes

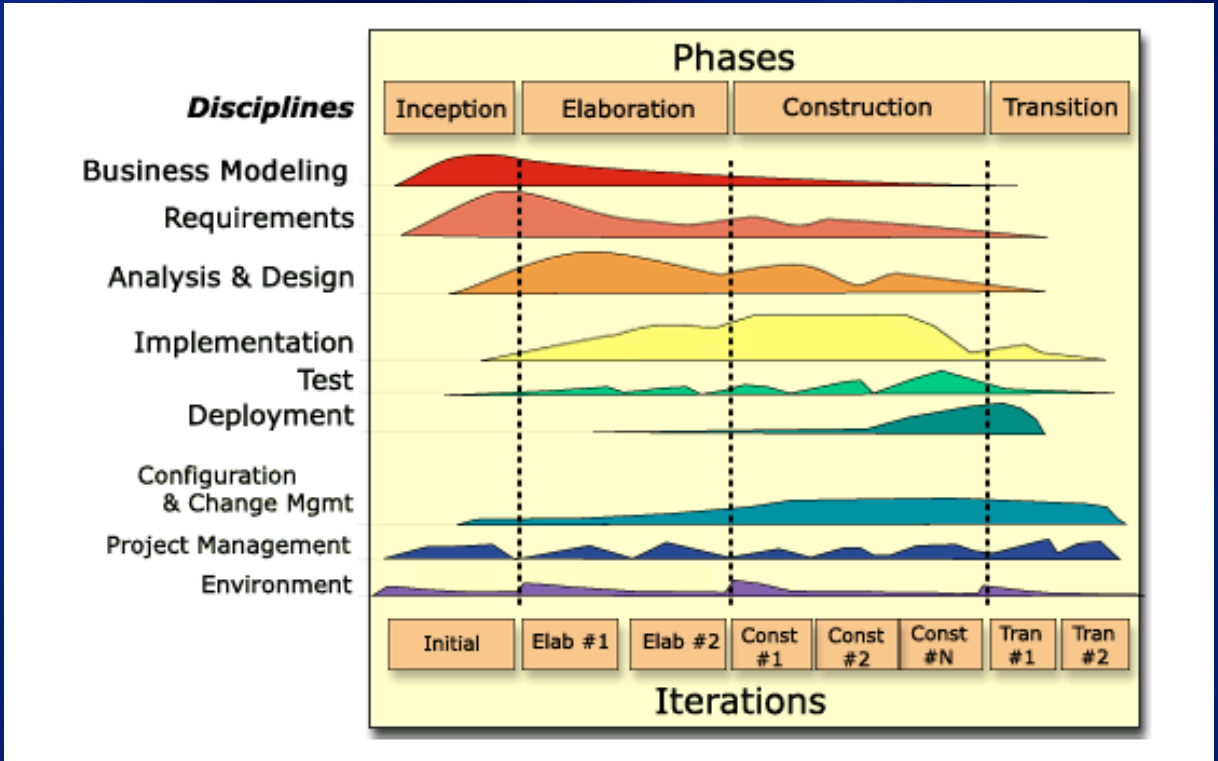
Fases do RUP (3/4)

- Construção
  - O software deve ser construído completamente, ou seja, deve-se adicionar a musculatura ao esqueleto (arquitetura)
  - Visa-se a capacidade operacional do software

Fases do RUP (4/4)

- Transição
  - Esta fase envolve a realização de testes com o usuário, corrigir defeitos encontrados e realizar treinamento

RUP (*Rational Unified Process*)



Número de iterações

- Projeto muito simples
  - Concepção: 1
    - ✓ Protótipo GUI ou prova de conceito
  - Elaboração: 1
    - ✓ Protótipo arquitetural

Número de iterações

- Projeto muito simples
  - Construção: 1
    - ✓ Montagem do produto
  - Transição: 1
    - ✓ Finalização do produto

Número de iterações

- Projetos maiores
  - Concepção: 1
    - ✓ Protótipo GUI ou prova de conceito
  - Elaboração: 2
    - ✓ Protótipo arquitetural
    - ✓ Finalização da arquitetura

Número de iterações

- Projetos maiores
  - Construção: 2
    - ✓ Montagem parcial do produto
    - ✓ Maturação e montagem final do produto
  - Transição: 1
    - ✓ Finalização do produto

Número de iterações

- Projetos maiores ainda com muitas tecnologias desconhecidas
  - Concepção: +1 iteração
    - ✓ Para mais protótipos
  - Elaboração: +1 iteração
    - ✓ Para mais explorações arquiteturais

Número de iterações

- Projetos maiores ainda com muitas tecnologias desconhecidas
  - Construção: +1 iteração
    - ✓ Devido ao tamanho do produto
  - Transição: +1 iteração
    - ✓ Para mais feedback operacional

Variação nas iterações (1/2)

- Modelo de domínio completamente diferente
  - Consolidação de conceitos
  - Mais iterações na concepção

- Nova arquitetura precisa ser montada ou existem muitos riscos
  - Mais iterações na elaboração

Variação nas iterações (2/2)

- Produto grande e complexo
  - Mais iterações na construção

- Organização com pouca experiência em projetos iterativos
  - Três iterações que produzem software [0,1,1,1]

Consideração sobre as iterações

- Projetos normalmente possuem entre 6 e 8 iterações em projetos típicos em empresas com experiência em projetos iterativos
- Tamanho de uma iteração depende do número de pessoas e da cultura de desenvolvimento
- Iterações menores que 1 mês devem ser definidas cuidadosamente e não devem ser maiores que 3 meses

Estrutura do RUP