Engenharia de Software

Natália Schots

Aula 3

Agenda

- Processo de Desenvolvimento de Software
- Modelos de Ciclos de Vida
  - Cascata
  - RAD
  - Incremental
  - Prototipação
  - Iterativo

Na aula passada...

Introdução à ES

- "Aplicação de uma **abordagem sistemática, disciplinada e quantificativa** para o desenvolvimento, operação e manutenção de software, isto é, a aplicação da engenharia ao software" (IEEE, 1993)
- Elementos básicos
  - Processos
  - Métodos
  - Ferramentas
- Histórico da ES

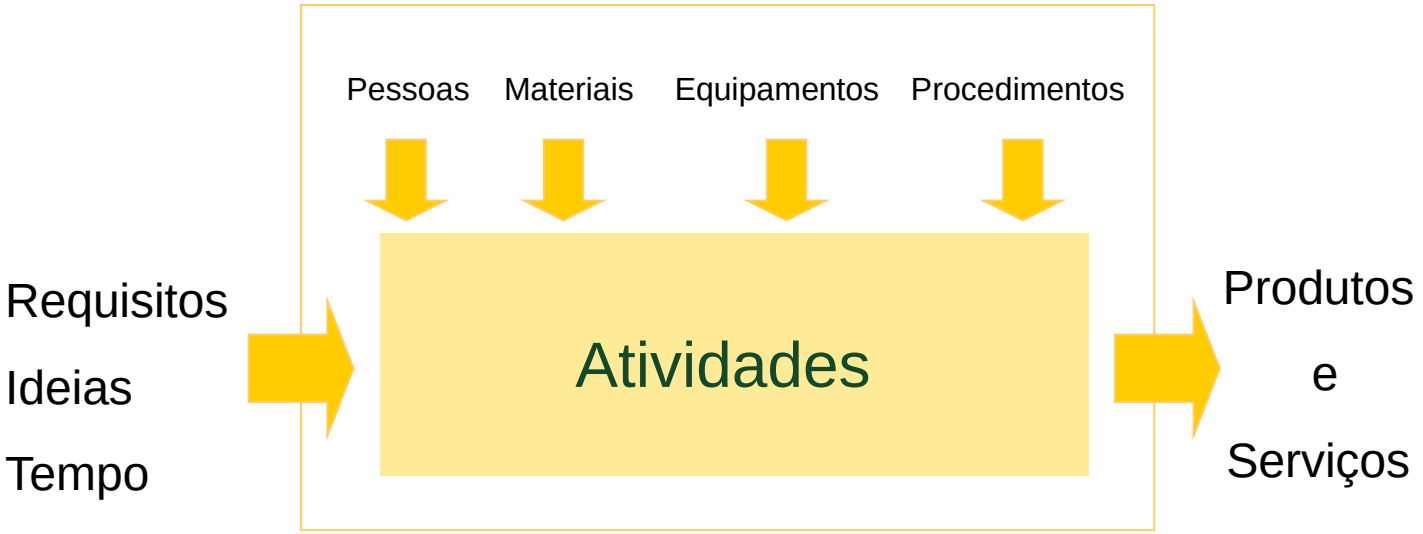
Introdução à ES

- Mitos da Engenharia de Software
  - Maus exemplos e aplicações
- Princípios de Hooker
  - **Princípio #1**: tem que existir uma razão
  - **Princípio #2**: *Keep it simple, sir*! (KISS)
  - **Princípio #3**: Mantenha o estilo
  - **Princípio #4**: O que é produzido por você é consumido por outros
  - **Princípio #5**: Esteja pronto para o futuro
  - **Princípio #6**: Planeje para reutilização
  - **Princípio #7**: Pense!

Processo do desenvolvimento de software

Processo

- Um conjunto de atividades inter-relacionadas ou interativas, que transforma insumos (entradas) em produtos (saídas) (ISO 9000, 2000)



[FLORAC e CARLETON, 1999]

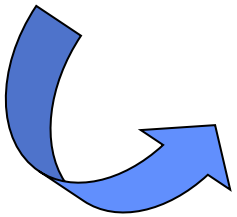
Características de um processo

- Tecnologicamente competitivos, adaptáveis e adequados com relação ao tempo
- Capazes de produzir produtos que atingem as necessidades do cliente e do negócio
- Adequados à cultura organizacional

[ROCHA, 2013]

Processo imaturo

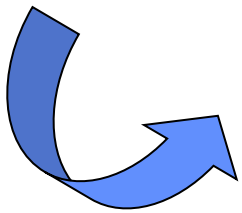
- **Características**
  - *Ad hoc* - Improvisado
  - Fortemente dependente dos profissionais
  - Indisciplinado



- **Consequências**
  - Pouca produtividade
  - Qualidade de difícil previsão
  - Alto custo de manutenção
  - Risco na adoção de novas tecnologias

Processo maduro

- **Características**
  - Processo conhecido por todos
  - Apoio visível da alta administração
  - Auditagem da fidelidade ao processo
  - Medidas do produto e do processo
  - Adoção disciplinada de tecnologias



- **Consequências**
  - Papéis e responsabilidades claramente definidos
  - Acompanhamento da qualidade do produto e da satisfação do cliente
  - Expectativas para custos, cronograma, funcionalidades e qualidade do produto é usualmente alcançada

Dimensões de um processo (1/2)

[Campos, 2014]

Dimensões de um processo (2/2)

- Um processo de desenvolvimento de software define:
  - Um ciclo de vida para o software e um paradigma
  - Os métodos que serão utilizados durante o desenvolvimento
  - As ferramentas que apoiarão estes métodos
  - Os papéis das pessoas envolvidas no desenvolvimento

Principais atividades (1/5)

- 3 atividades básicas
  1. Entender o problema e avaliar alternativas de solução
  2. Desenvolver a solução escolhida
  3. Implantar a solução

- Problema-exemplo:
  - Controle das finanças domésticas

Principais atividades (2/5)

- Exemplo:
  1. Entender o problema e avaliar alternativas de solução
     - Deve-se: controlar as receitas e despesas de cada mês, registrar o saldo disponível calculado mês a mês, categorizar os gastos por categoria...
     - Alternativas de solução:
       a. Planilha eletrônica (Excel) local
       b. Planilha eletrônica (Excel) em pasta compartilhada
       c. Planilha eletrônica (Google Drive) compartilhada
       d. Aplicativo no celular
     - Prós e contras de cada alternativa
     - Escolha da melhor solução

Principais atividades (3/5)

– Exemplo (cont.):

2. Desenvolver a solução escolhida

			Janeiro	Fevereiro	Março	Abril	Maio	Junho	Julho	Agosto	Setembro	Outubro	Novembro	Dezembro	Total
Receitas	Recebimentos	Salário	1200,34												1200,34
		Férias													0,00
		13º salário													0,00
		Restituição IRPF													0,00
		Aposentadoria													0,00
		Aluguel													0,00
		Outros 1													0,00
		Outros 2													0,00
		Outros 3													0,00
		Outros 4													0,00
		Outros 5													0,00
		Total	1200,34	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	1200,34
	Alimentação	Supermercado	120,00												120,00
		Feira / Sacolão													0,00
		Padaria	57,00												57,00
		Outros													0,00
		Outros 1													0,00
		Outros 2													0,00
		Total	177,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	177,00
	Moradia	Aluguel / Prestação	350,00												350,00
		Condomínio													0,00
		Água	28,00												28,00
		Serviço de limpeza													0,00
		Luz	78,00												78,00
		Gás													0,00
		IPTU													0,00
		Reforma													0,00
		Outros 1													0,00

Principais atividades (4/5)

- Exemplo (cont.):
  3. Implantar a solução

Principais atividades (5/5)

Atividades básicas	Tarefas específicas	Produtos gerados
Entender o problema ou a oportunidade	Estudo de viabilidade	Anteprojeto
	Análise dos requisitos	Documento de requisitos
Desenvolver a solução	Modelagem do sistema	Diversos diagramas (ex. Diagrama de classes, Diagrama de sequência...)
	Projeto do sistema	Especificação do sistema
	Codificação	Código-fonte Documentação do produto
Implantar a solução	Entrega do sistema	Sistema em operação
	Manutenção do sistema*	Melhoria do sistema

* OBS.: Em muitas empresas, a manutenção é vista como um processo à parte do processo de desenvolvimento de software

Estudo de Viabilidade (1/2)

- Deve-se estimar as dimensões do problema
  - Perspectivas
  - Visão sociotécnica
  - Partes envolvidas (*stakeholders*)

- Análise do sistema atual
  - Identificar áreas envolvidas
  - Identificar fatores críticos
  - Identificar vantagens e desvantagens do(s) sistema(s) existente(s)

Estudo de Viabilidade (2/2)

- Estudo preliminar das possíveis soluções
  - Necessidades de informação dos usuários
  - Requisitos de recursos
  - Custos
  - Benefícios
  - Viabilidade da solução

- **Principal produto:** Anteprojeto do sistema

Análise dos Requisitos

- Detalha as necessidades de informação
  - Estudo sobre as necessidades de informação da organização e dos usuários finais
  - Estudo sobre as atividades, recursos e produtos de quaisquer sistemas de informação existentes
  - Estudo sobre a capacidade exigida para satisfazer as necessidades de informação dos usuários finais

- **Principal produto:** Documento de requisitos (lista de requisitos funcionais)

Modelagem do Sistema

- Detalha os requisitos do sistema identificados anteriormente
  - Cada modelo apresenta uma visão ou perspectiva diferente do sistema
  - Descreve o sistema para que o implementado
  - Documenta a estrutura e a operação do sistema

- **Principal produto:** Diversos tipos de diagramas

Projeto do Sistema

- Especifica **como** o sistema de informação atenderá as necessidades de informação dos usuários
- Envolve:
  - Projeto da interface com o usuário (telas, formulários, relatórios)
  - Projeto de dados (estrutura dos dado a processar e armazenar)
  - Projeto do processo (programas e procedimentos a serem desenvolvidos/alterados)
- **Principal produto:** Especificação do sistema

Codificação do Sistema

- É o desenvolvimento do sistema solicitado

- Envolve as atividades de:
  - Aquisição de hardware, software e serviços
  - Desenvolvimento ou modificação do software (programação)
  - Teste de programas, procedimentos e hardware
  - Documentação do sistema

- **Principal produto:** Código-fonte, documentação do produto

Entrega do Sistema

- É a preparação para que o sistema possa ser utilizado

- Envolve as atividades de:
  - Treinamento do usuário final
  - Transição do antigo sistema para o novo

- **Principal produto:** Sistema em operação

Manutenção do Sistema

- Compreende ajustes, melhorias, adaptação e expansão do sistema a partir do seu uso
- Envolve a monitoração e avaliação dos resultados e desempenho da solução em uso
- **Principal produto:** Melhorias do sistema

OBS.: Em muitas empresas, a manutenção é vista como um processo à parte do processo de desenvolvimento de software

Processos de apoio

- Atividades que auxiliam e garantem uma execução correta das atividades principais do processo de desenvolvimento de software
  - Gerência de projetos
  - Medição
  - Garantia da qualidade
  - Gerência de configuração
  - Gerência de reutilização

Norma ISO/IEC 12207 (1/2)

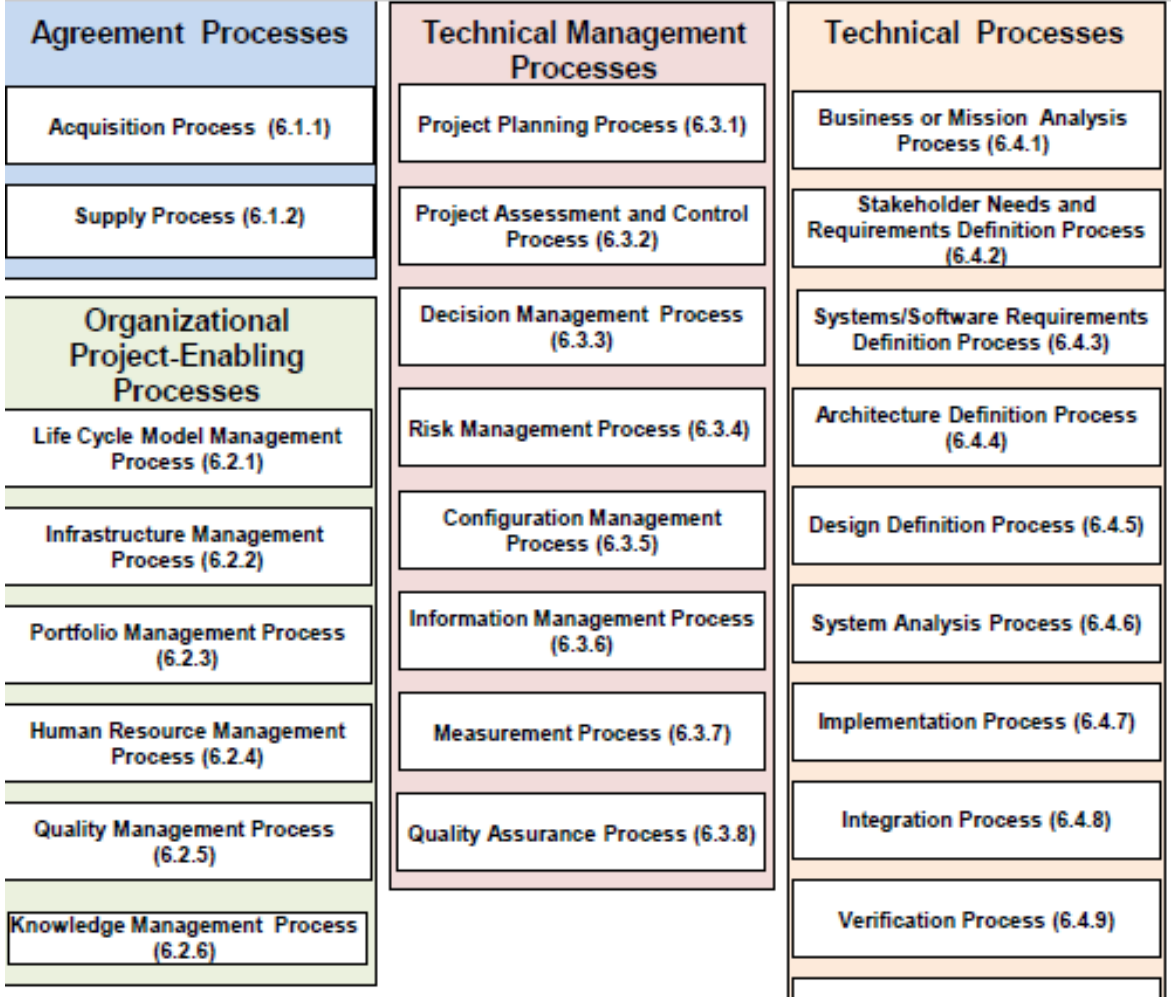
- Norma que define um *framework* para processos de ciclo de vida com terminologia bem definida
  - Usada como referência

- Contém processos, atividades e tarefas que devem ser aplicadas na aquisição, fornecimento, desenvolvimento, operação e manutenção de produtos de software

Norma ISO/IEC 12207 (2/2)

- Descreve os processos do ciclo de vida de software mas não especifica os detalhes de como implementar ou realizar as atividades e tarefas dos processos

- Não prescreve:
  - nome, formato e conteúdo da documentação
  - um modelo específico de ciclo de vida
  - um método de desenvolvimento de software



[ISO 12207, 2017]

Equipe e responsabilidades

Patrocinador

- Dois tipos:
  - Patrocinador externo: cliente e/ou usuário
  - Patrocinador interno: diretor ou gerente da área envolvida

- Funções: determina objetivos específicos, prazos, negocia planejamento e cronogramas

- Responsabilidades: participação nas principais reuniões, aprovações e avaliação dos principais resultados e produtos

Gerente do projeto (1/2)

- Pessoa diretamente ligada aos procedimentos operacionais e sistêmicos do projeto, sistema ou software em questão, com **poder de decisão**

- Funções: responsável pelo planejamento e cumprimento das atividades relacionadas ao desenvolvimento

- Responsabilidades: participação direta e efetiva no projeto, em todas as reuniões e aprovações de todos os resultados e produtos; coordenação da equipe

Gerente do projeto (2/2)

Equipe técnica

- Especialistas necessários para a execução das atividades de desenvolvimento

- Funções: executores das atividades operacionais planejadas: analistas de sistemas, arquitetos, desenvolvedores, testadores etc.

- Responsabilidades: participação direta e efetiva no projeto; realização das atividades

Usuário

- Pessoas que utilizarão o sistema de informação

- Responsabilidades: participação direta e efetiva no projeto; fornecimento das informações necessárias

Modelos de Ciclo de Vida

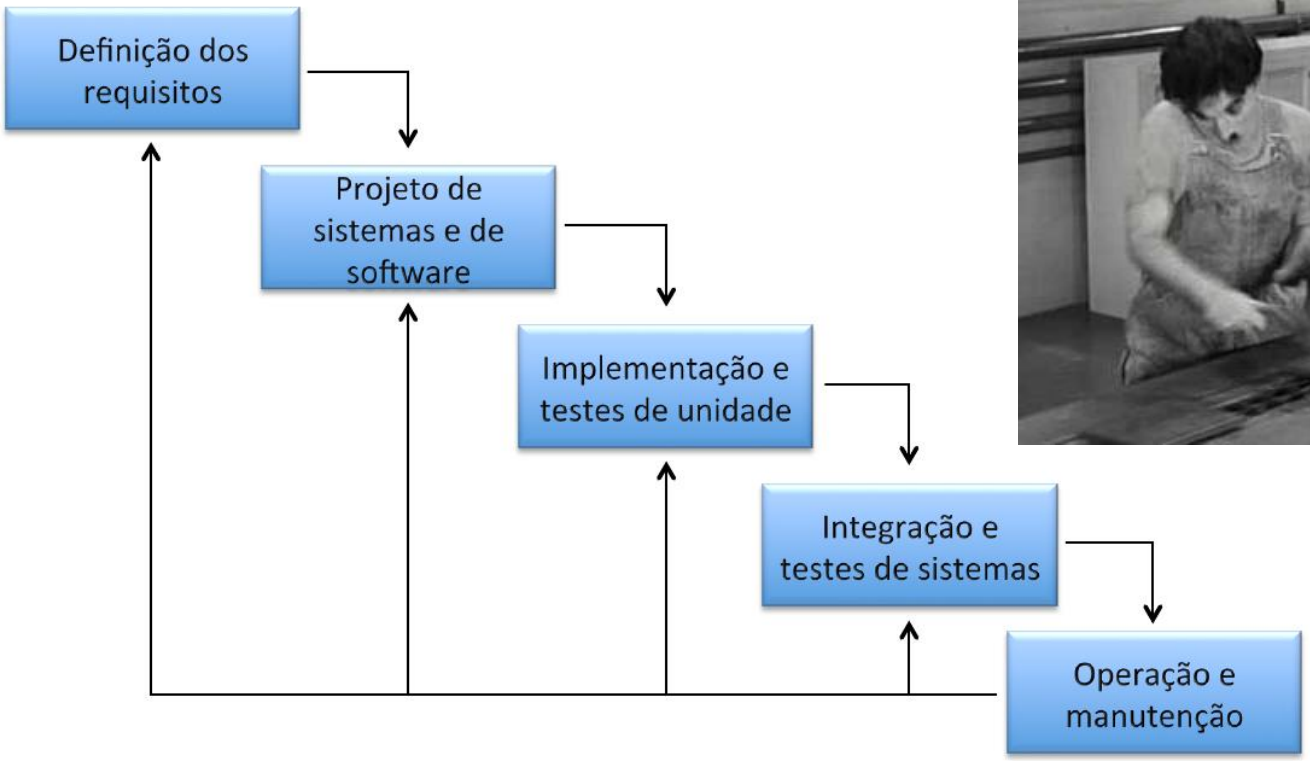
O que é?

- Também denominados “Modelos de Processo”
- É uma representação abstrata de um “esqueleto” de processo
- Considerados como “processos pré-fabricados”
  - Apresentam características predefinidas
  - Devem ser adaptados ao contexto real de uso (projeto, equipe, cliente)

Critérios para selecionar ciclo de vida

- Relacionados à equipe e aos usuários
  - Experiência dos usuários no domínio da aplicação
  - Facilidade de expressão dos usuários
  - Experiência da equipe no domínio da aplicação
  - Disponibilidade de recursos para a equipe
  - Grau de acesso aos usuários
- Relacionados ao problema
  - Grau de maturidade do domínio da aplicação
  - Complexidade do problema
  - Frequência e complexidade das mudanças nos requisitos
  - Controles e produtos que precisam ser entregues

Modelo Cascata (1/3)



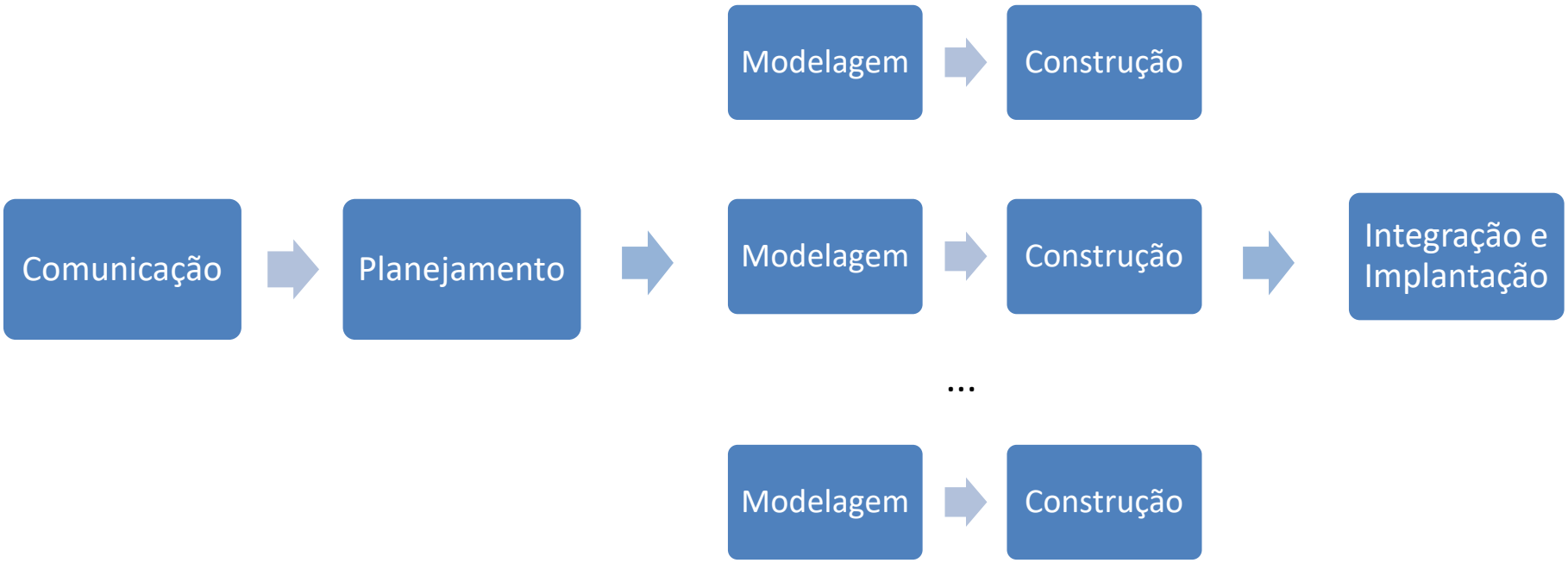
Modelo Cascata (2/3)

- Composto por uma determinada sequência de atividades
- Uma atividade começa a ser executada quando a anterior termina
- Resultado de uma etapa é utilizado na etapa seguinte
- Guiado por documentos
- Ciclo de vida mais antigo e mais utilizado

Modelo Cascata (3/3)

- Útil quando se tem requisitos estáveis e bem definidos
  - Ex.: Adicionar um novo dispositivo legal em um sistema de contabilidade
- Não lida bem com incertezas
- Fornece pouca visibilidade do estado do projeto
  - Tempo longo para a primeira entrega
  - Dificuldade na obtenção de *feedback* do cliente

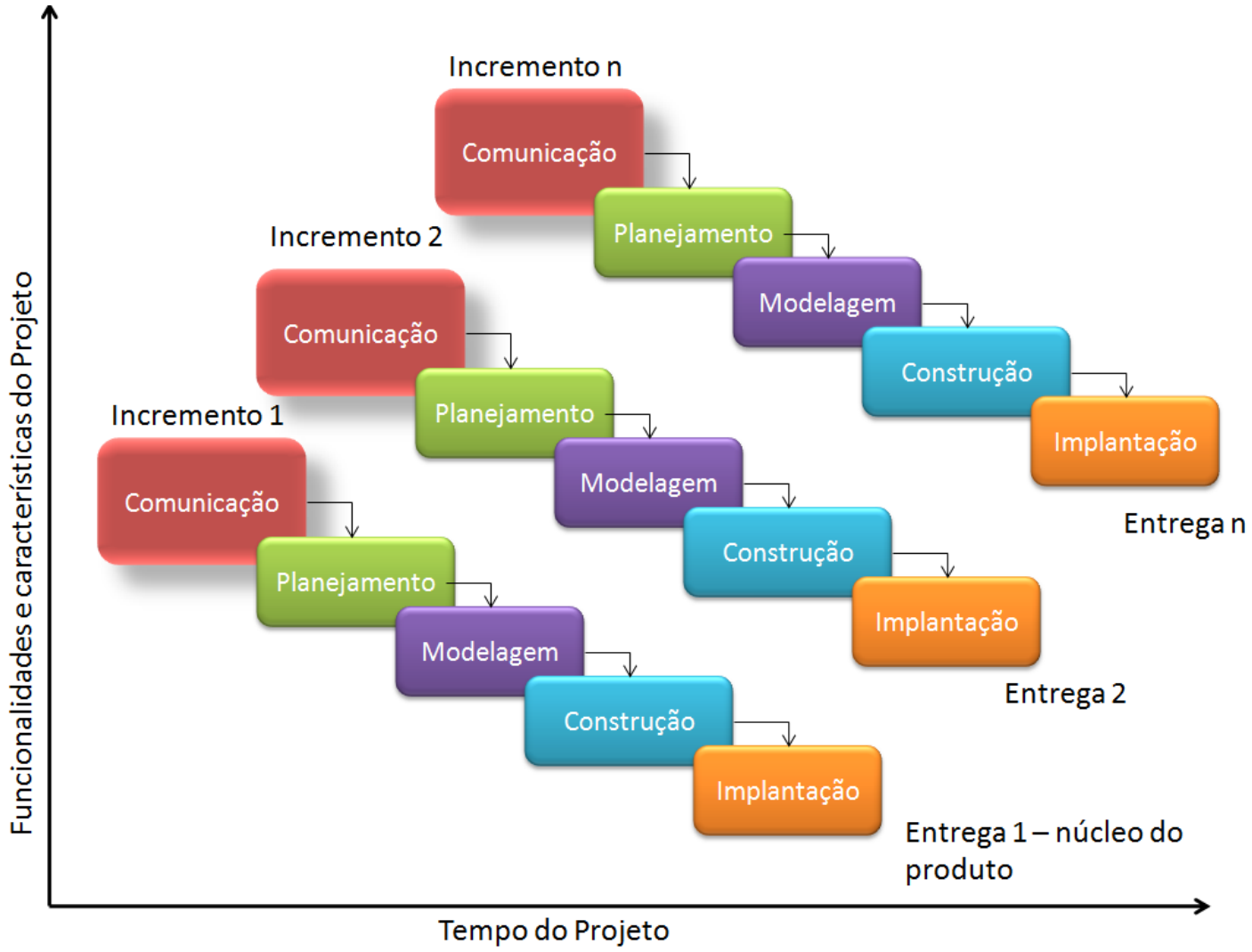
RAD (*Rapid Application Development*) (1/2)



RAD (*Rapid Application Development*) (2/2)

- Funcionamento equivalente ao cascata
- Principais diferenças
  - Visa entregar o sistema completo em 60 a 90 dias
  - Múltiplas equipes trabalham em paralelo nas etapas de modelagem e construção
  - Assume a existência de componentes reutilizáveis e geração de código
- Difícil de ser utilizado em domínios novos ou instáveis

Modelo Incremental (1/2)

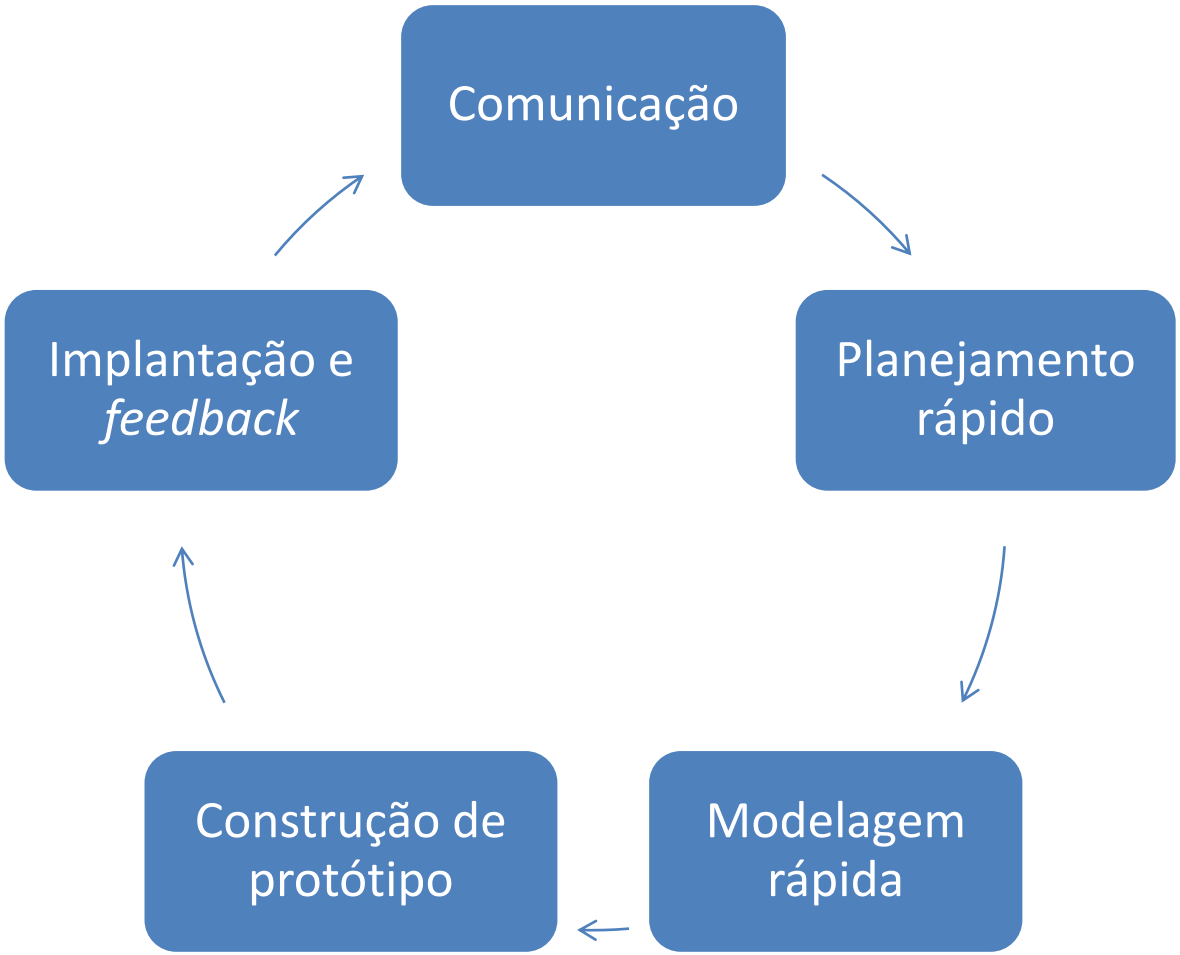


Modelo Incremental

Modelo Incremental (2/2)

- Faz entregas incrementais do software
  - Cada incremento é construído via um mini-cascata
  - Cada incremento é um software operacional
- Versões anteriores ajudam a refinar o plano
  - Feedback constante do cliente
- Diminuição da ansiedade do cliente
  - O cliente rapidamente recebe uma versão funcional do software
- É a base utilizada pelos atuais métodos ágeis

Prototipação (1/2)



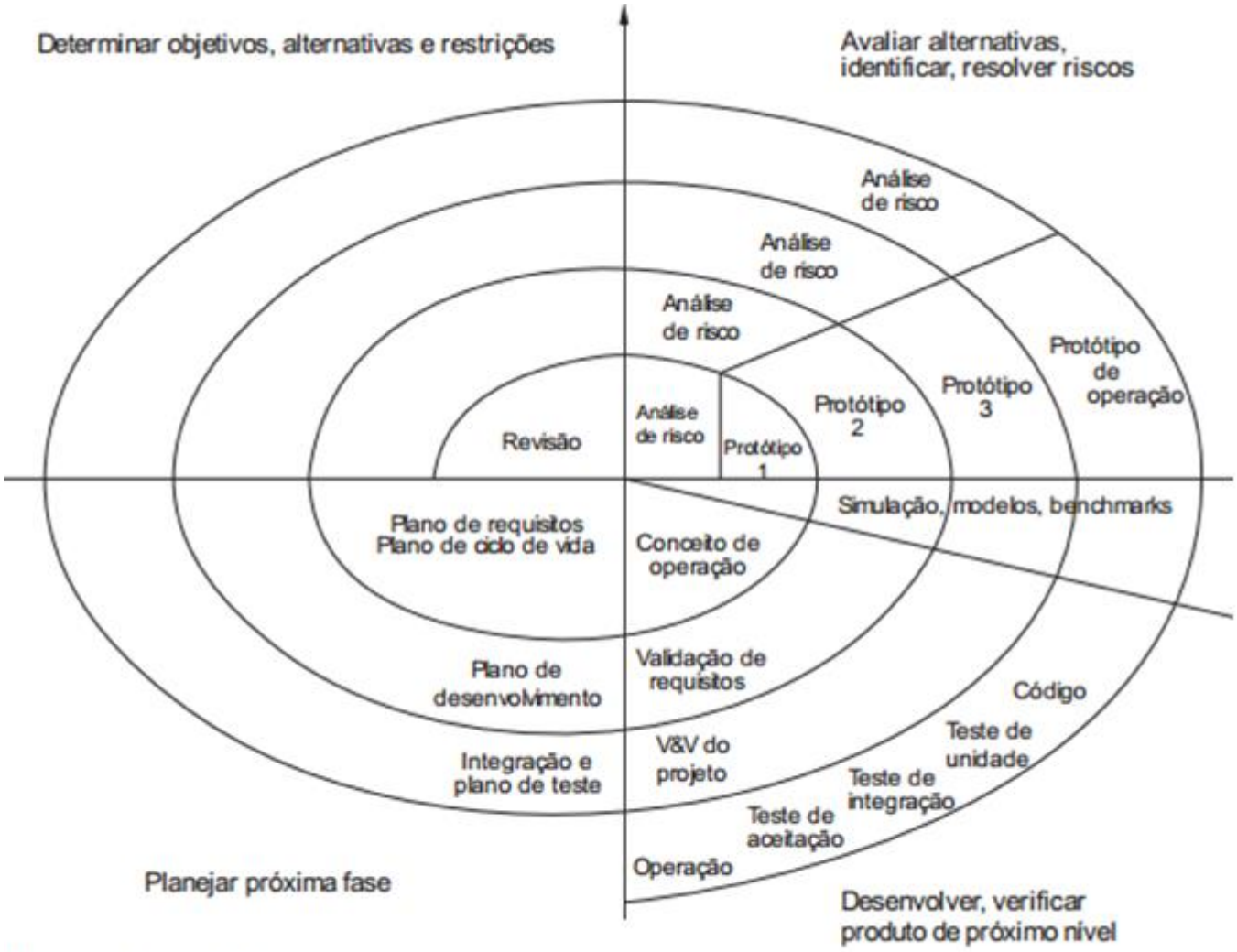
Prototipação (2/2)

- Usualmente utilizado como auxílio a outro modelo de ciclo de vida
- Útil para
  - Validar um requisito obscuro com o cliente
  - Verificar o desempenho de um algoritmo específico
- Deveria ser descartado (“jogado fora”) no final
  - Protótipos não são produtos
  - Apesar disto, alguns clientes tendem a desejar colocar protótipos em ambiente de produção

Modelos Iterativos

- É a repetição das atividades de desenvolvimento, fazendo progressos através de tentativas sucessivas de refinamento
- Utilizado quando não se conhece totalmente os requisitos no início do desenvolvimento
- Modelos ou processos que usam esse conceito:
  - Espiral
  - Processo Unificado (ex. RUP)

Espiral (1/3)



Modelo em espiral do processo de software de Boehm (©/EEE,1988).

Espiral (2/3)

- Foi proposto por Boehm em 1988
- O processo é representado como uma espiral, cada ciclo (*loop*) representando uma fase do processo de software.
  - Ex.: o *loop* mais interno pode estar relacionado à viabilidade do sistema; o próximo loop, à definição de requisitos; o próximo, ao projeto de sistema e assim por diante
- Cada *loop* está dividido em 4 setores:
  - Definição de objetivos
  - Avaliação e redução de riscos
  - Desenvolvimento e validação
  - Planejamento (da próxima fase)

Espiral (3/3)

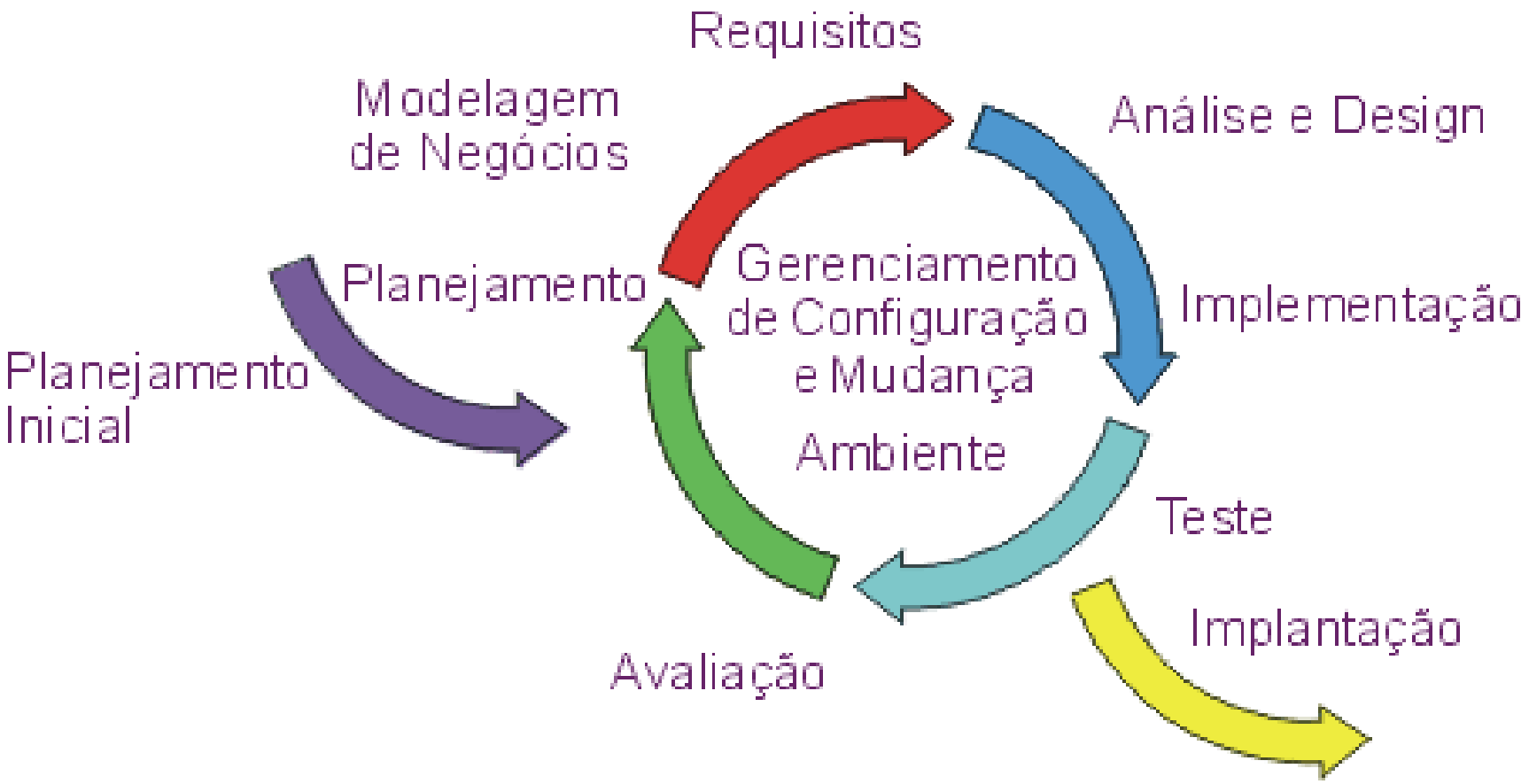
- Foco principal no gerenciamento de riscos
- A cada *loop*
  - O conhecimento aumenta
  - O planejamento é refinado
  - O produto gerado no ciclo anterior é evoluído (não é descartado)
- Cada *loop* evolui o sistema, mas não necessariamente entrega um software operacional
  - Modelo em papel
  - Protótipo
  - Versões do produto
  - etc.

Processo Unificado

- Tentativa de obter o que há de melhor em cada modelo de ciclo de vida (iterativo + evolutivo + desenvolvimento ágil)

- Exemplos:
  - RUP – *Rational Unified Process*
  - OpenUP – *Open Unified Process*
  - AUP – *Agile Unified Process*

RUP (1/6)



Modelo RUP

RUP (2/6)

- Descrito em três perspectivas:
  - Perspectiva dinâmica: apresenta as fases do modelo ao longo do tempo
  - Perspectiva estática: apresenta as atividades realizadas no processo (*workflows*)
  - Perspectiva prática: sugere boas práticas a serem utilizadas durante o processo

RUP (3/6)

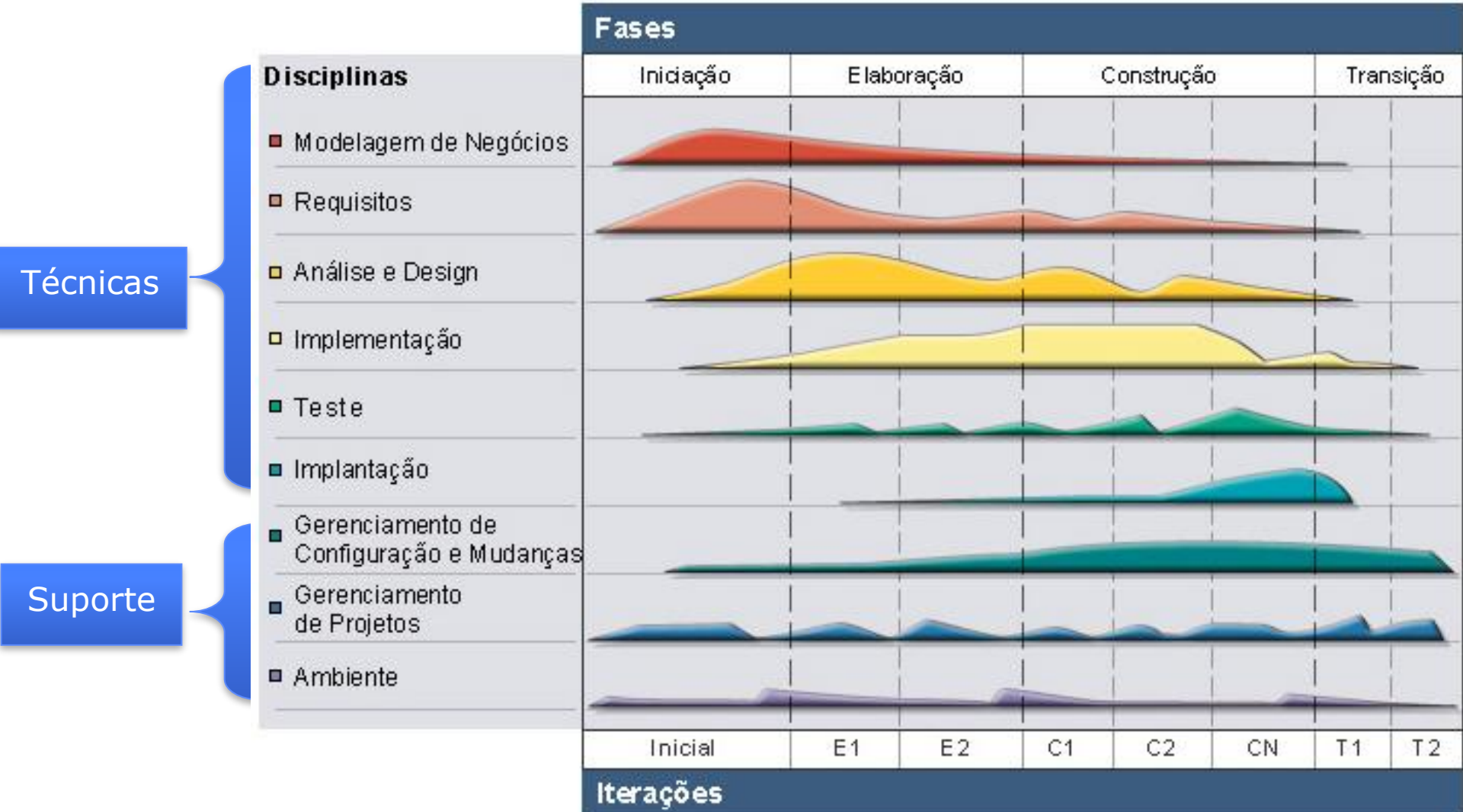
- Iterações pequenas: entre 2 a 6 semanas
- Fases:
  - Concepção: visão geral do projeto
  - Elaboração: elicitação de requisitos, em detalhes; na 1ª iteração, deve-se detalhar um ou dois requisitos de maior risco
  - Construção: implementação iterativa
  - Transição: testes finais e implantação

RUP (4/6)

- *Workflows*:
  - Modelagem de negócio (casos de uso de negócio)
  - Requisitos (casos de uso)
  - Análise e projeto (modelos de arquitetura, de componentes, de objetos e de sequência)
  - Implementação
  - Teste
  - Implantação
  - Gerenciamento de configuração e mudanças
  - Gerenciamento de projeto
  - Ambiente (ferramentas para a equipe)

RUP (5/6)

- Fases x *Workflows*



RUP (6/6)

- Práticas fundamentais recomendadas:
  - Desenvolver software iterativamente
  - Gerenciar os requisitos
  - Usar arquitetura baseada em componentes
  - Modelar o software visualmente (utilizando UML)
  - Verificar a qualidade do software
  - Controlar as mudanças do software

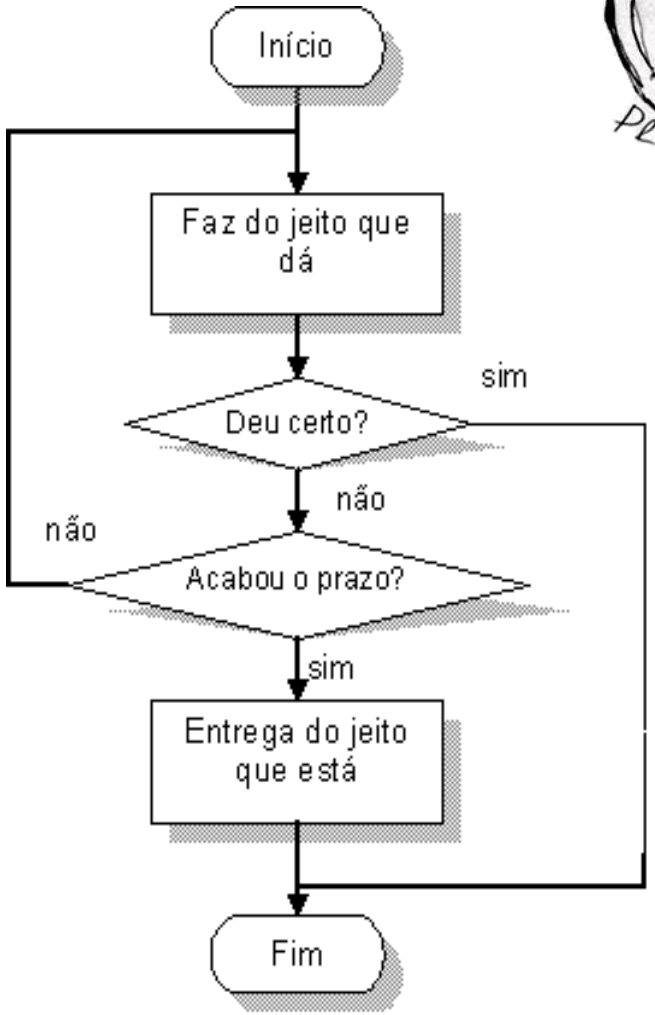
Incremental x Espiral

- Incremental

- Iterativo

Go Horse Process (GHP) (1/2)

- É uma **crítica** que satiriza o mau-uso de certos "métodos" de desenvolvimento de software



Go Horse Process (GHP) (2/2)

- Principais “princípios”:
  - Pensou, não é GHP
    - GHP não pensa, faz a primeira coisa que vem à mente
  - Quanto mais GHP você faz, mais precisará fazer
    - Para cada problema resolvido, mais uns 7 são criados
  - GHP é totalmente reativo e não tem prazo
  - Seja autêntico, GHP não respeita padrões
  - Acostume-se ao sentimento de fracasso iminente
  - GHP nem sempre é POG
    - Muitas POG’s exigem um raciocínio muito elevado

Mais informações em: https://gohorseprocess.com.br/

Referências

- Pressman, R.S.; “Engenharia de Software”; 6ª edição, Ed. McGraw-Hill, 2006
- Sommerville, I., “Engenharia de Software”, 11ª edição, Ed. Pearson, 2019
- Slides Engenharia de Software – Professor Leonardo Murta
- Slides Engenharia de Software – Professor Marcelo Schots
- Slides Processo de Software – Professora Ana Regina Rocha
- Slides Introdução à Engenharia de Software – Professor Márcio Barros
- Slides Processo de Software – Professor Gleison Santos
- FALBO, R. A.; BARCELLOS, M. P., 2011, Notas de Aulas – Processo de Software, UFES.