



Engenharia de Software: Introdução

Mestrado em Ciência da Computação
2008
Profa. Itana Gimenes



Programa

1. O processo de engenharia de software
2. UML
3. O Processo Unificado
 1. Captura de requisitos
 2. Análise de software
 3. Projeto de software
 4. Geração de código
4. Ferramentas de Apoio ao Processo de Software



Estratégias

- Desenvolvimento de um projeto de software em grupo de dois alunos
- Prova escrita
- Eventuais trabalhos que podem valer pontos
- Ferramenta: Jude



Bibliografia Básica

- PRESSMAN, R., Software Engineering: A Practitioner's Approach, McGraw-Hill Series in Computer Science.
- McDERMID, J. A. (Ed.), Software Engineer's Reference Book, Butterworth-Heinmann, 1991.
- SOMMERVILLE, I., Software Engineering, Addison-Wesley.
- BOOCH, G., RUMBAUGH, J., JACOBSON, I., The Unified Modeling Language User Guide, Addison-Wesley, Janeiro 1999.
- RUMBAUGH, J., JACOBSON, I BOOCH, G., Unified Modeling Language Reference Manual, Addison-Wesley, Dezembro, 2005.
- JACOBSON, I BOOCH, G., RUMBAUGH, J., Unified Software Development Process, Addison-Wesley, Janeiro 1999.



Motivação

- Compreender as etapas de desenvolvimento de software.
- Conhecer a diferença entre o desenvolvimento de software de pequeno e grande porte.
- Conhecer as principais técnicas de desenvolvimento de software.
- Estar apto a especificar projetos de software em UML (*Unified Modeling Language*)



Conceitos

- Software

“criação intelectual compreendendo os programas, procedimentos, regras e qualquer documentação correlata à operação de um sistema de processamento de dados.” ISO9000-3

- Produto de Software

“Conjunto completo de programas de computador, procedimentos e documentação correlata, assim como dados designados para entrega a um usuário.” ISO9000-3

- Outros termos: sistemas de software, aplicações, aplicativos.



O Papel do Software na sociedade

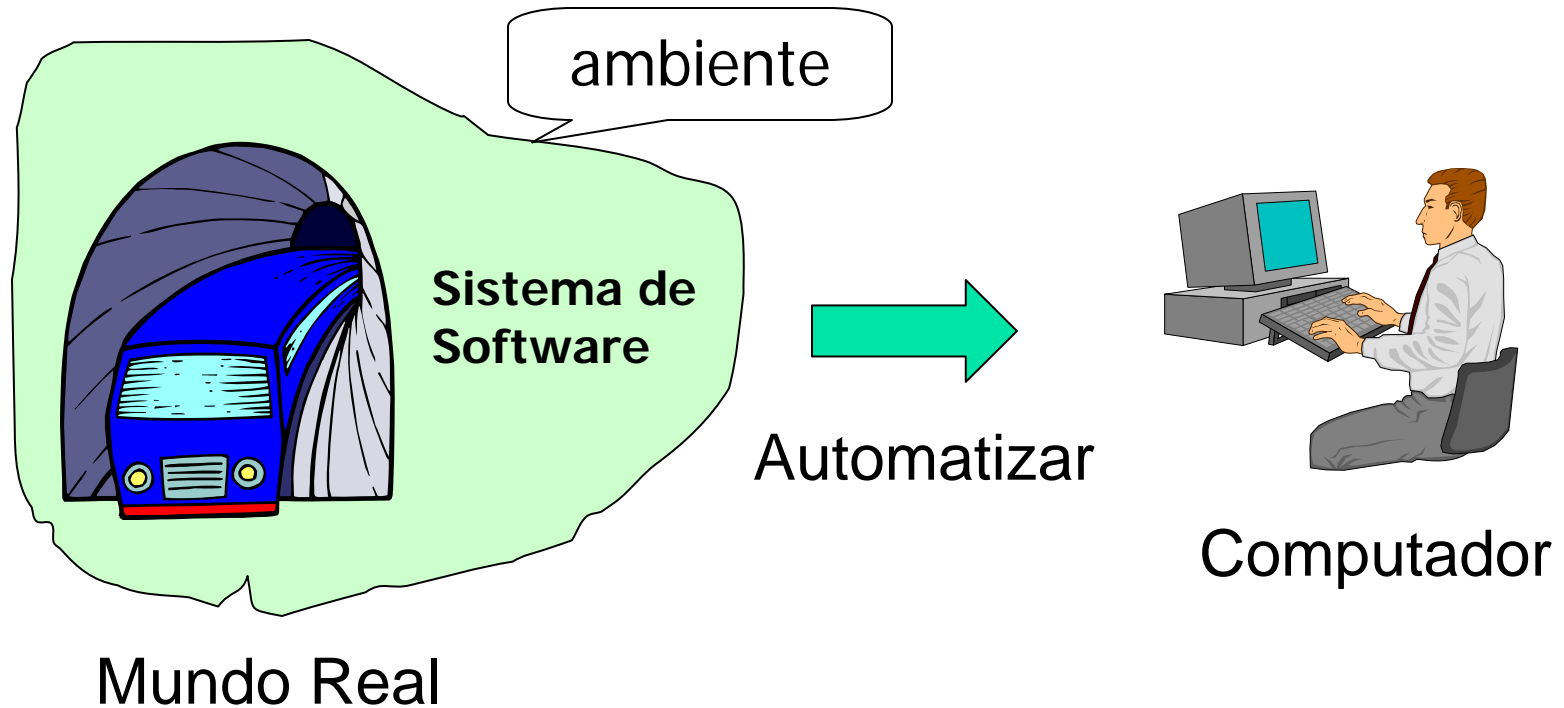
- Software está presente na grande maioria das atividades da sociedade:
 - Banco
 - Comunicação
 - Transporte
 - Escola
 - Previdência
 - Supermercado
 - Cinema
 - Bares
 - Parques de diversão



Exemplos de Riscos de Utilização de Software

- LAS (*London Ambulance System*) – o objetivo é automatizar o tratamento de chamadas de emergência e a atribuição de ambulâncias a acidentes.
- É o maior serviço de ambulância do mundo, abrangendo uma população de aproximadamente 6.8 milhões. LAS transporta mais de 5.000 pacientes por dia, recebe entre 2.000 e 2.500 chamadas diariamente.
- Falhas no sistema em 26 e 27 de Outubro de 1992 causaram problemas tais como:
 - alocação ineficiente de ambulâncias (duplicada e atrasada - múltiplas ambulâncias foram enviadas para o mesmo acidente ou o veículo mais próximo não foi enviado para o acidente mais próximo);
 - uma crescente lista de mensagens de exceção e uma enorme lista de espera;
 - um aumento do tempo de resposta proporcional ao aumento de mensagens e da lista de espera;
 - um número crescente de telefonemas repetidos (“call backs”).

Como Produzir software?





Características de Software

- Software é desenvolvido e não manufaturado no sentido clássico.
- Software não se deteriora ... não existem componente de reposição.
- Software é feito sob encomenda, ao invés de ser construído a partir de componentes.
- A evolução tecnológica afeta diretamente as facilidades que podem ser incorporadas ao software e as técnicas de construção destes.
- ! Tempo para transferência de tecnologia.



Tipos de Sistemas de Software

- Software básico
- Software para sistema em tempo real
- Software comercial
- Software para engenharia e aplicações científicas
- Software embarcado (ex. microwave)
- Software para computadores pessoais (shrink-wrap)
- Software baseado em inteligência artificial
- Software de entretenimento



Engenharia de Software

- Uma definição:
 - O estabelecimento e uso de um conjunto de princípios para se obter, **economicamente**, um software que seja **confiável** e trabalhe **eficientemente** em máquinas reais.
- Três elementos chaves:
 - métodos
 - ferramentas
 - **procedimentos (gerenciamento de projetos)**

A Evolução do Software

Os primeiros anos

- sistemas batch
- distribuição limitada
- software personalizado

A segunda era

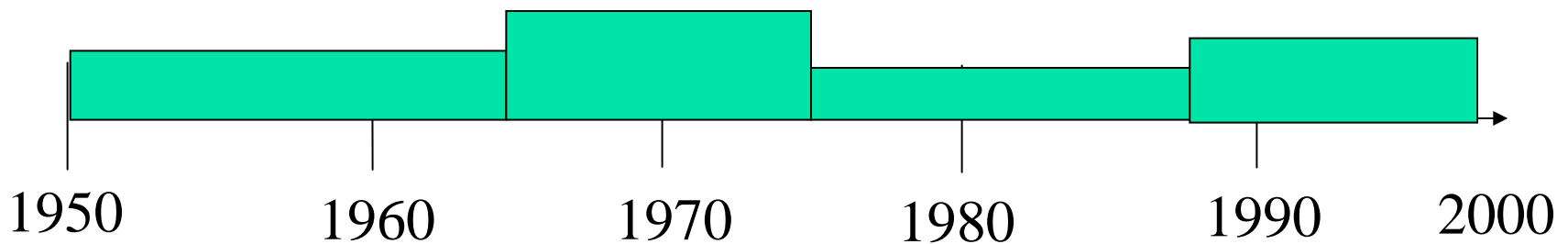
- sistemas multiusuários
- sistemas em tempo real
- banco de dados
- software produto

A terceira era

- sistemas distribuídos
- incorporação de inteligência
- hardware de baixo custo
- impacto do consumidor

A quarta era

- sistemas desktop poderosos
- tecnologia de orientação a objetos
- sistemas especialistas
- redes neurais
- computação paralela
- comunicação intergaláctica





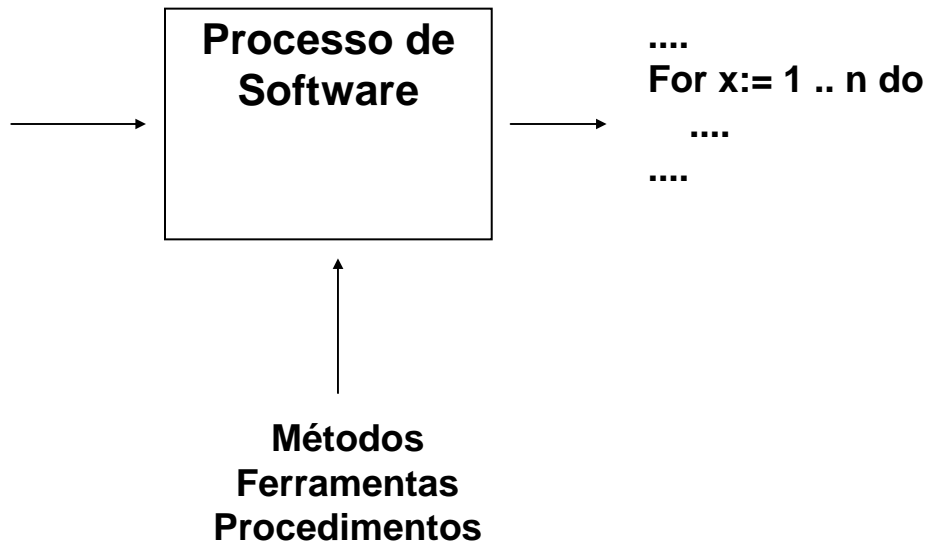
Modelos de Processo de Desenvolvimento de Software

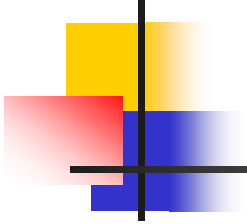
- Modelo de ciclo de vida
 - descrições abstratas do processo de desenvolvimento e modificação, tipicamente, mostrando os principais estágios de desenvolvimento e manutenção de um software executável.
- Processo de Software
 - Desenvolvimento
 - Manutenção
 - Uso



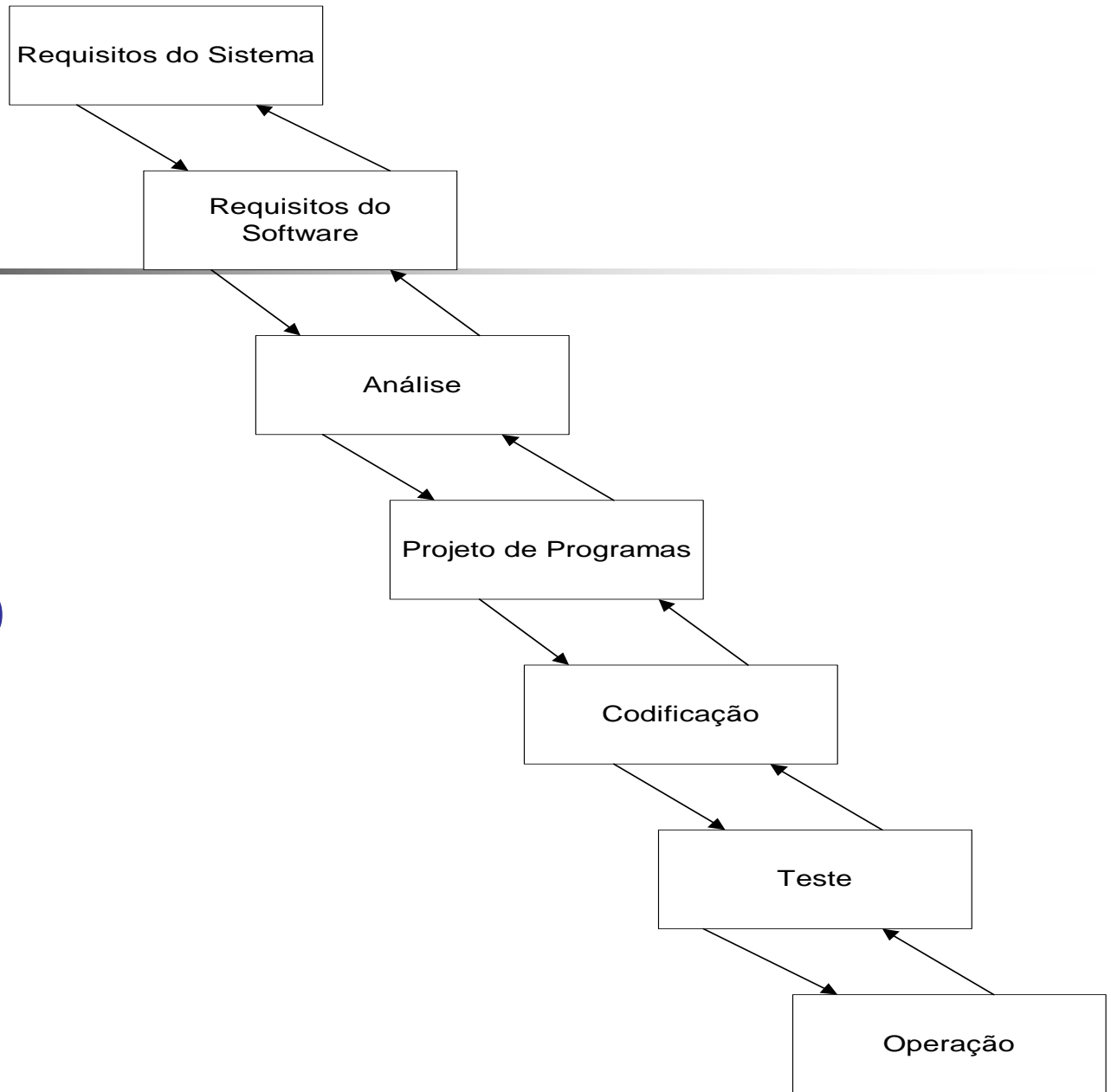
A idéia do processo de software

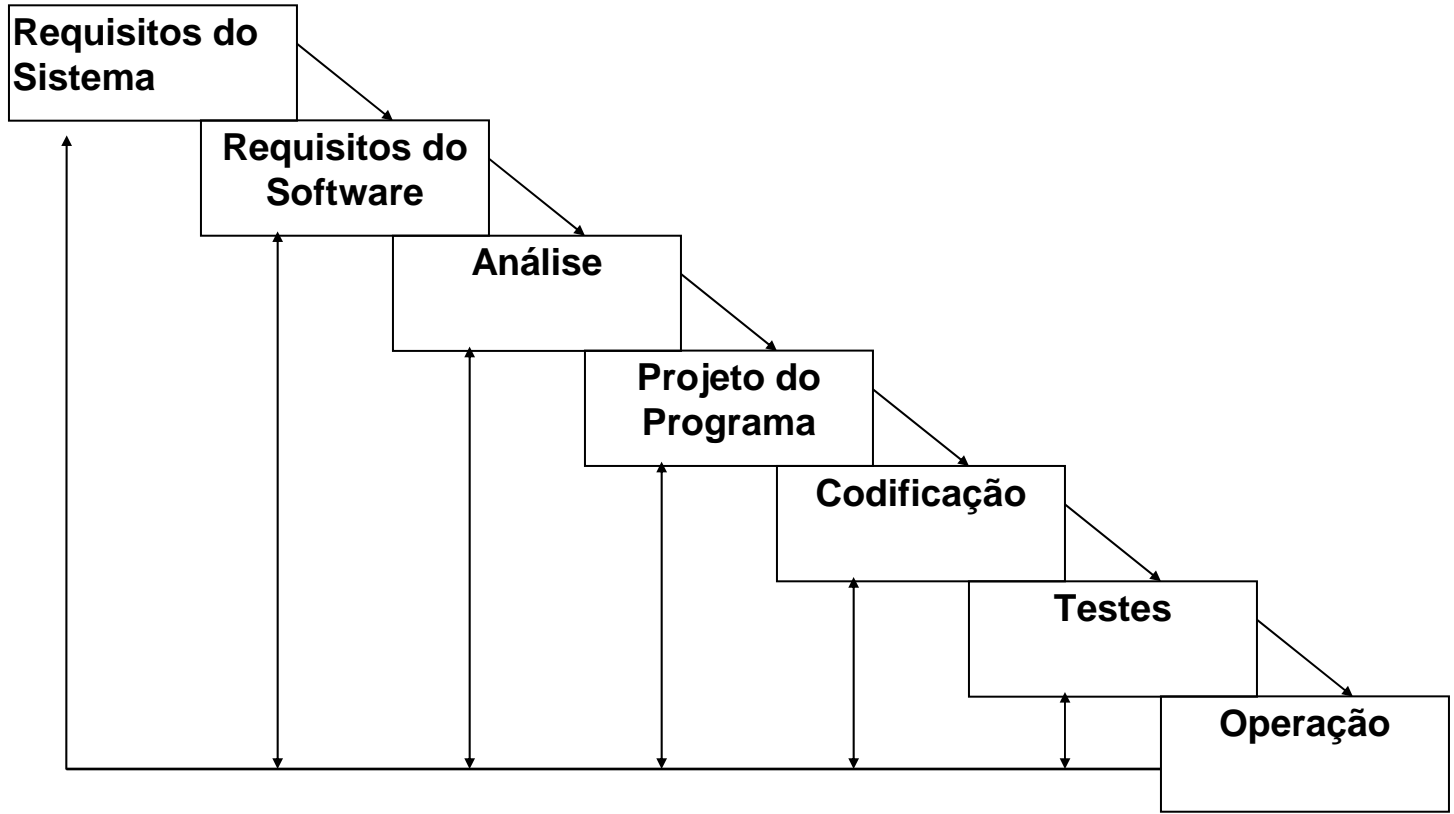
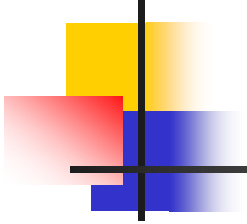
*Tenho uma lista de nomes
pessoas ...
Essas pessoas tem códigos
de identificação ...
Quero uma lista em ordem
alfabética ...
mas seria bom também ter
um relatório por
departamento*



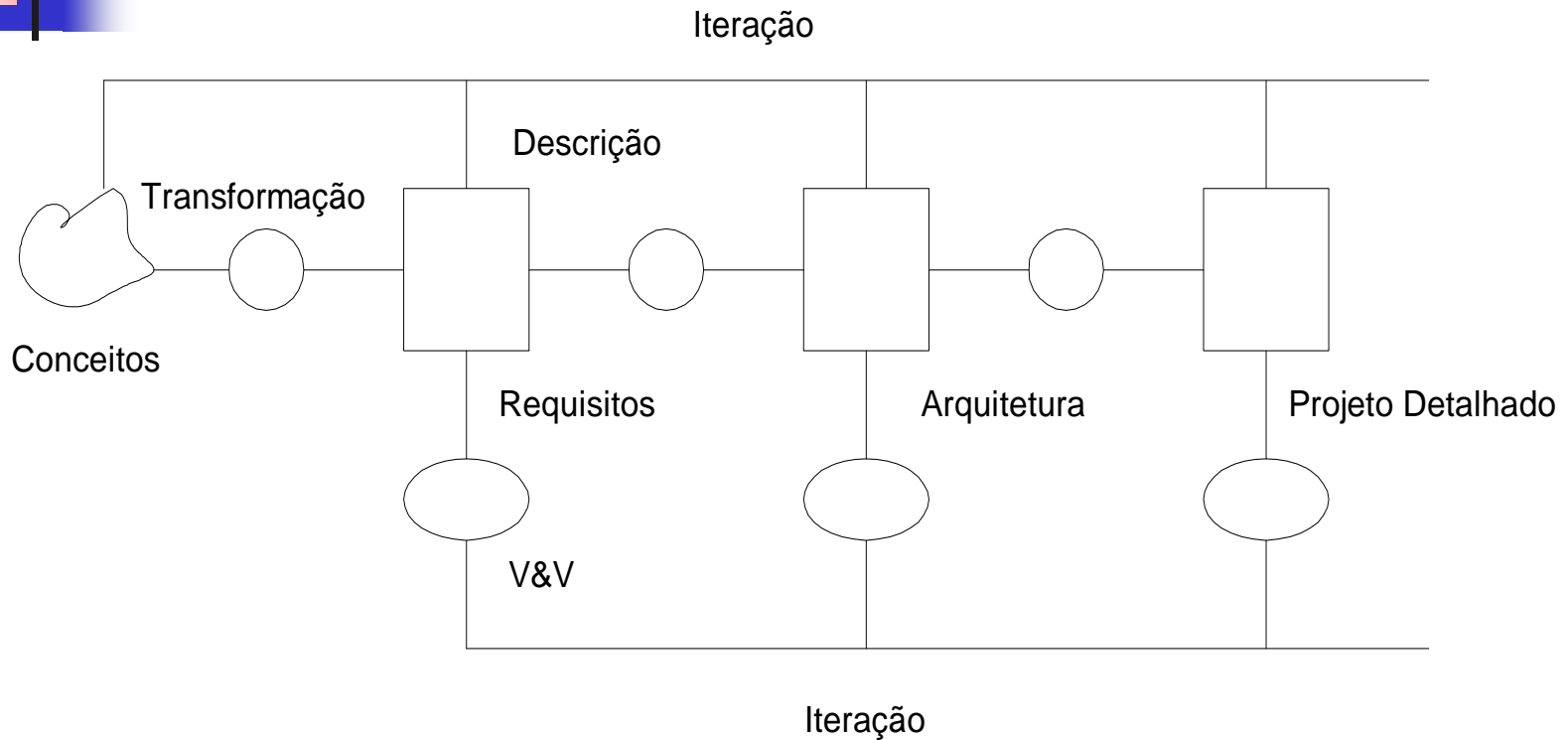


O Modelo Cascata





O Modelo Transformacional





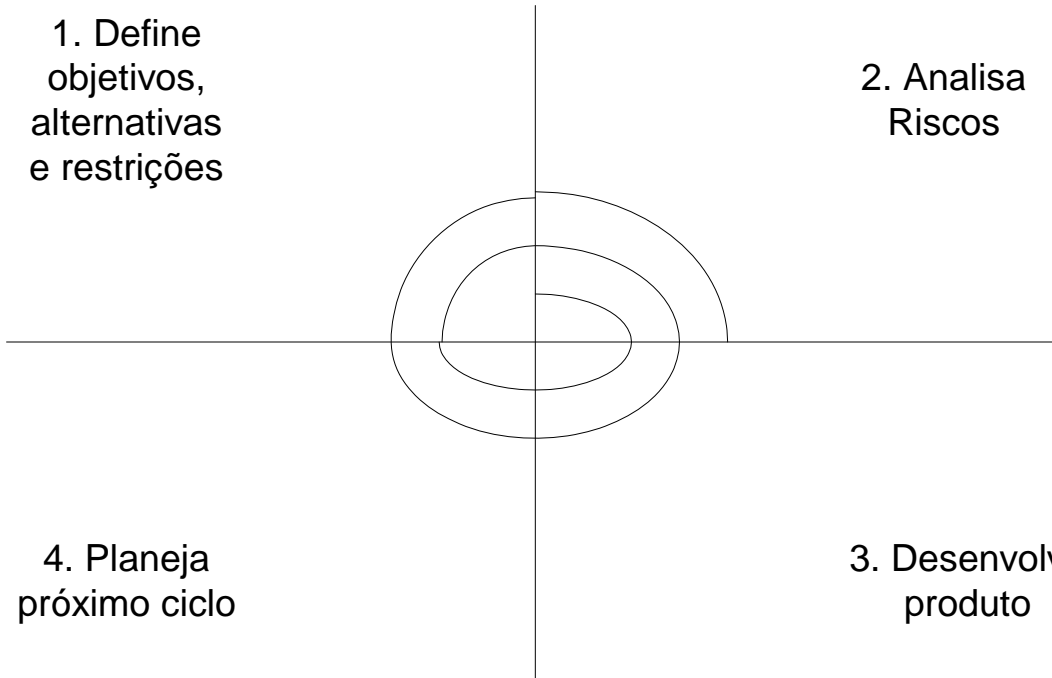
O Modelo Espiral

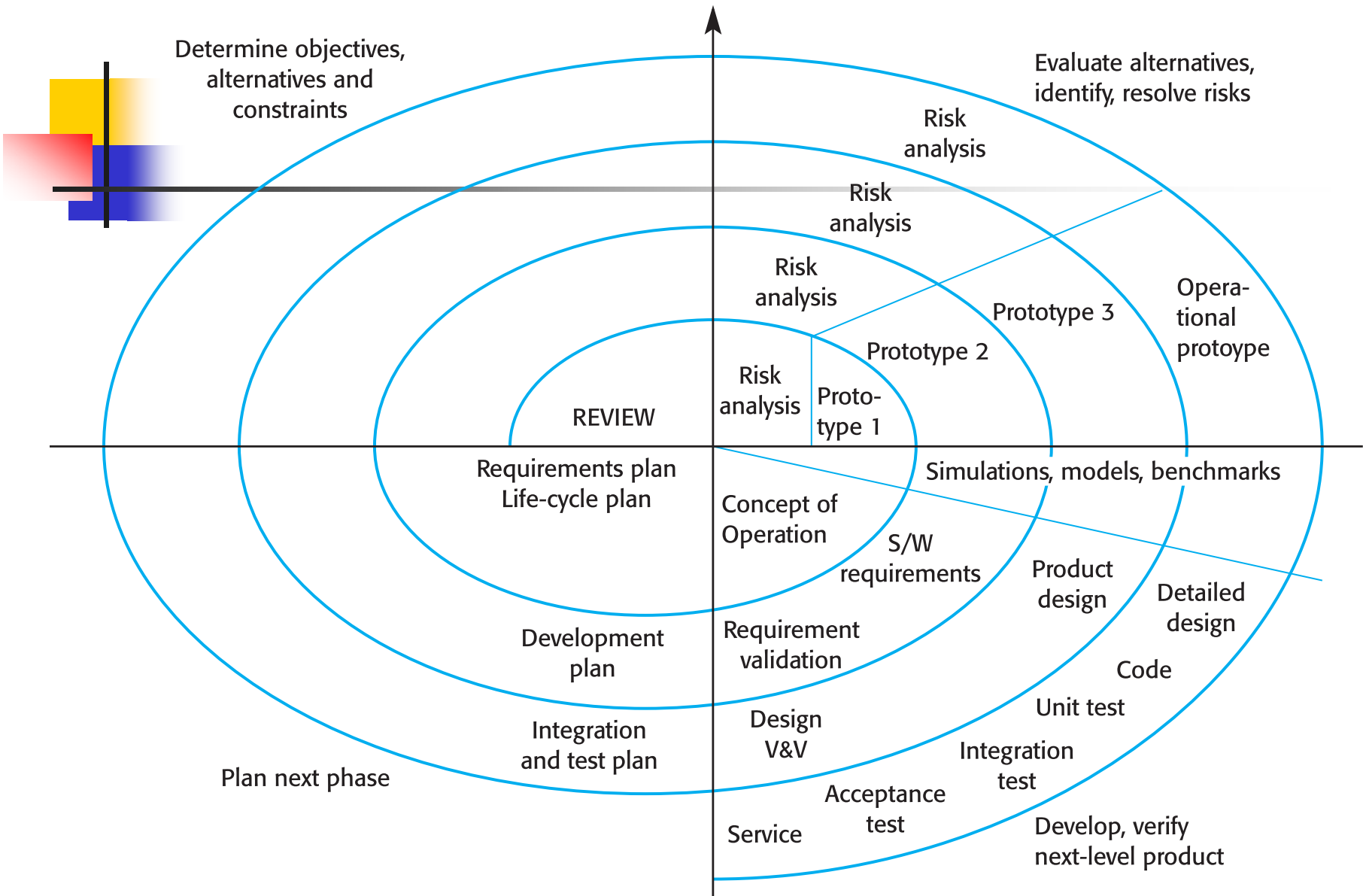
1. Define
objetivos,
alternativas
e restrições

2. Analisa
Riscos

3. Desenvolve
produto

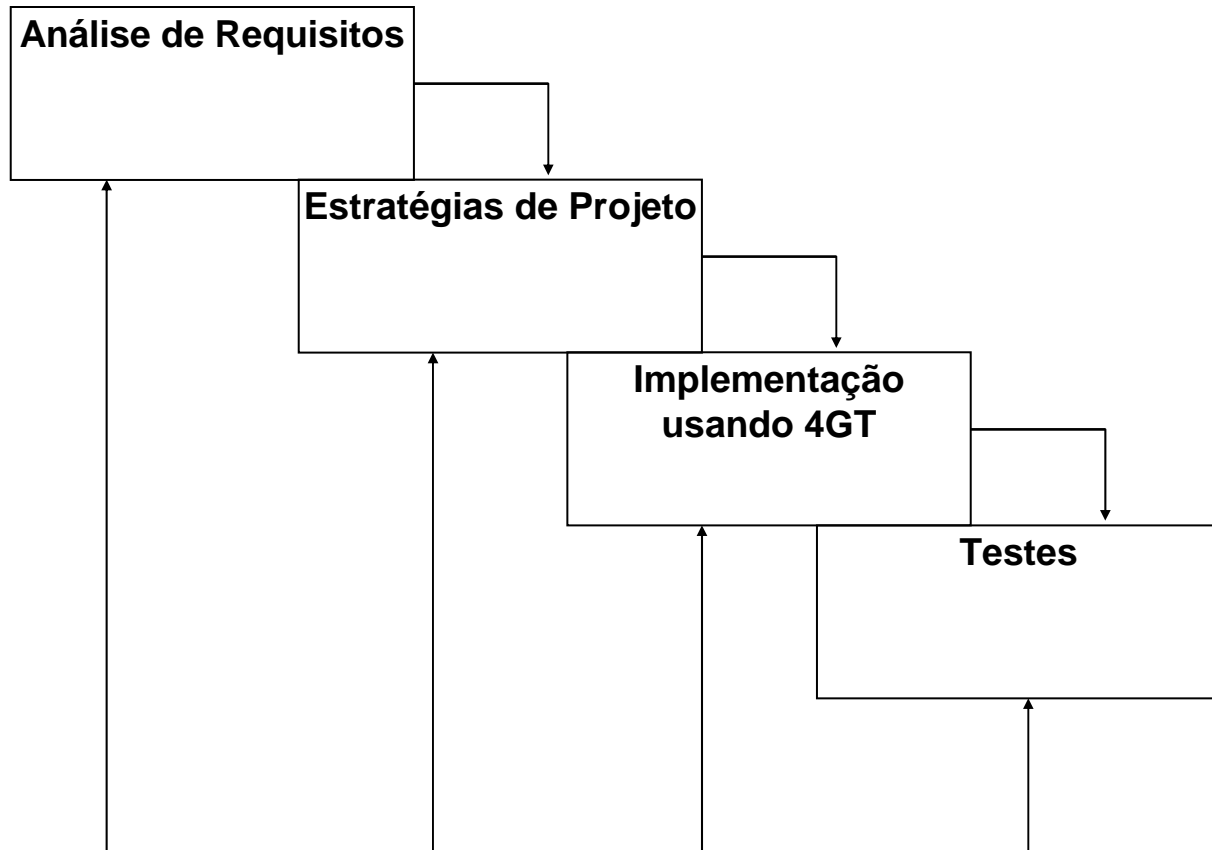
4. Planeja
próximo ciclo







Modelo 4GT





Abordagem Prototipação

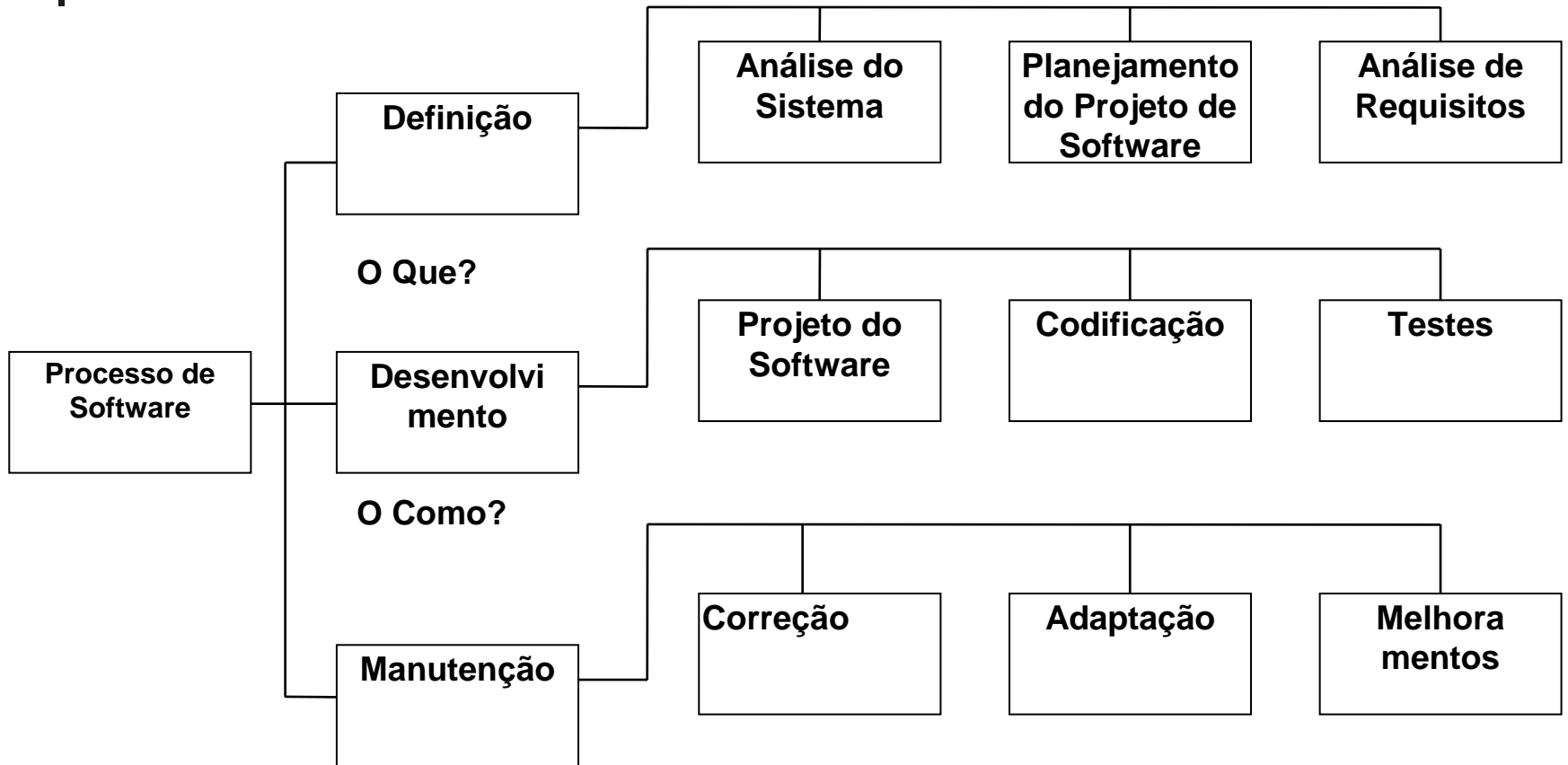
- Validar a precisão dos requisitos ou aceitabilidade das decisões.
- Validar a viabilidade de uma estratégia proposta.
- Observações:
 - protótipos só são válidos se construídos rapidamente
 - protótipos devem ser desprezados.



Abordagem Incremental

- Definir e desenvolver uma pequena parte do sistema de cada vez.
- Desenvolver um núcleo do sistema inicialmente e depois adicionar funcionalidades em subprojetos.
- Grande atração é que gera sistemas parciais executáveis e utilizáveis de onde se pode obter *feedbacks* e ganhar credibilidade do usuário.
- Na prática é difícil de desenvolver uma abordagem incremental sem ter um entendimento completo dos requisitos.

Generalização



A Obrigação ...



Seleção de Estágios de Desenvolvimento

- Análise de requisitos
- Especificação do software
- Projeto da arquitetura
- Projeto detalhado
- Implementação
- (Manutenção e evolução)

What?

How?



Análise de requisitos (Sistema e Software)

- Software é sempre parte de um sistema maior que envolve hardware, pessoas, etc.-
Modelo de negócios
- Coleta de requisitos do software especificamente.
- Técnicas de comunicação



Especificação do sistema

- Expressar os requisitos de maneira formal através de diagramas bem definidos ou especificações matemáticas.



Projeto da arquitetura

- Determinar a estrutura do software com seus componentes e conectores.



Projeto Detalhado

- Projetar a concretização da especificação do software, definindo base de dados, representações de interfaces, algoritmos, etc.



Implementação

- Escrever o projeto do sistema em uma linguagem de programação.



Manutenção e Evolução

- Corrigir eventuais erros no software e efetivar atualizações.



O Ciclo de Vida Canônico

- Estudo de Viabilidade
- Iniciação do projeto
- Especificação de requisitos
- Projeto da arquitetura
- Projeto detalhado
- Codificação
- Teste de unidade
- Teste de aceitação
- Teste operacional
- Encerramento do projeto
- Operação
- Desativação do produto



Métodos de Construção de Software

- Abordagem sistemática para construir software
 - Procedimentos
 - Notação
 - Ferramenta
 - Exemplo: Análise estruturada, OMT, Catalysis, Processo Unificado
- Paradigma de desenvolvimento
 - Estruturado, orientado a objetos
- A escolha deve considerar:
 - características técnicas da aplicação
 - características técnicas do método
 - cobertura do ciclo de vida (Pode-se usar mais de um método)



Observações sobre o processo de desenvolvimento

- Sempre deve existir um processo de software definido - padrões de qualidade.
- O modelo canônico deve ser tratado como uma referência que deve ser adaptada para cada situação.
- Criar um processo baseado em fases específico para cada projeto.
- O profissional deve estar apto a avaliar a aplicação a ser desenvolvida e a situação do ambiente de desenvolvimento para decidir qual o melhor processo de software a ser definido.