Processos de Software

Metodologias e Desenvolvimento de Software

Pedro Salgueiro
pds@di.uevora.pt
CLV-256

Processos de software

- Conjunto estruturado de actividades usado para desenvolver software

- Grande variedade de processos de software, mas todos partilham:
  - Especificação – define-se o que o sistema deve fazer;
  - Desenho e implementação – definir a organização do sistema e implementar o sistema;
  - Validação – verificar se o sistema faz o que deve
  - Evolução – alterar o sistema de acordo com novos requisitos

- Modelo de processo de software
  - Representação abstracta de um processo
  - Descrição de um processo visto uma perspectiva especifica

Processos de software

Descrição dos processos

- Conjunto de actividades:
  - Especificar o modelo de dados;
  - Desenhar o interface de utilizador;
  - …
- **Ordem entre estas actividades**
- Podem incluir:
  - Produtos resultantes de uma actividade;
  - “Papéis” que reflectem as várias responsabilidades das pessoas envolvidas no processo;
  - Pré e pós-condições que devem ser verificadas antes e depois do processo ou produto

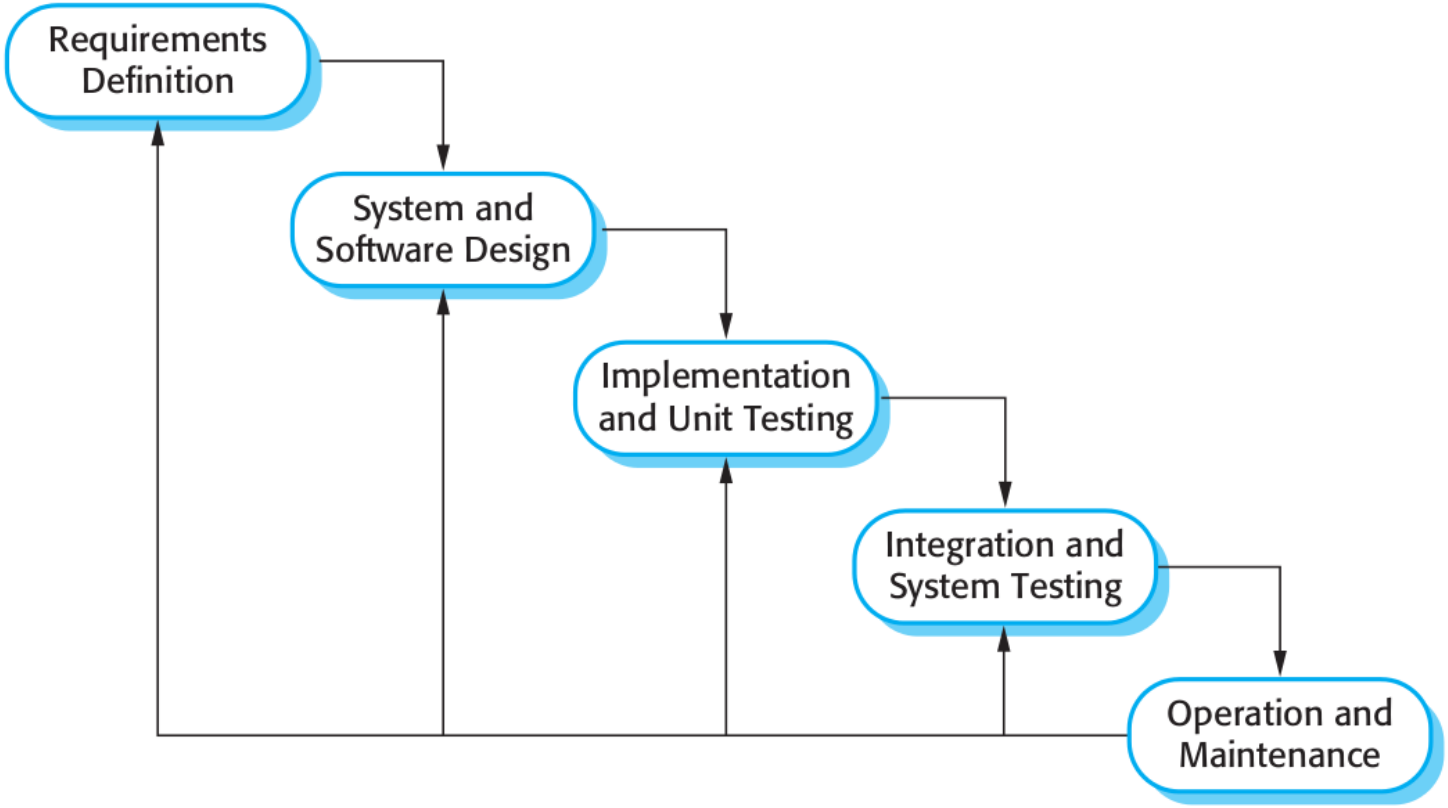
Tipos de processos de software

- Baseados em planos
  - Todas actividades do são planeadas com antecedência
  - Progresso é verificado de acordo com o o plano
- Processos ágeis
  - Planeamento incremental
  - Mais fácil alterar o processo
    - Reflectir as alterações dos requisitos
- Na prática
  - Elementos dois tipos
- Não existe um processo correcto ou errado

Modelos de processos de software

- Waterfall (cascata)
  - Baseado em planos
  - Especificação e desenvolvimento
    - Etapas separadas e distintas
- Desenvolvimento incremental
  - Especificação, desenvolvimento e validação
    - Entrelaçadas
  - Podem ser baseados em planos ou processos ágeis
- Reutilização de software
  - Software construído a partir de componentes existentes
  - Podem ser baseados em planos ou processos ágeis

Modelo Watterfall - overview



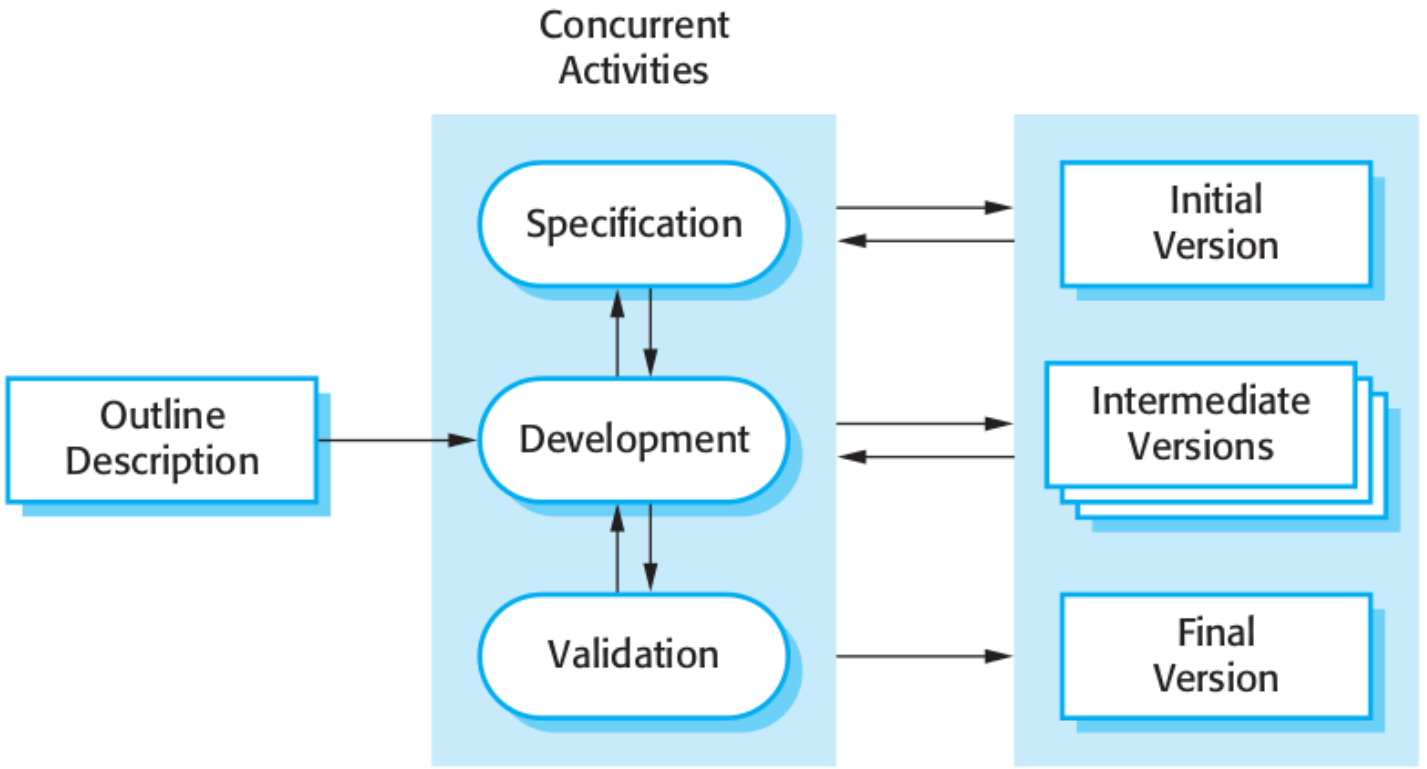
Modelo Waterfall

- Etapas
  - Análise e especificação dos requisitos
  - Desenho do software
  - Implementação e testes unitários
  - Integração e testes do sistema
  - Operação e manutenção
- Desvantagens
  - Dificuldade em incluir alterações depois de dar início ao processo
  - Cada etapa apenas começa depois da ultima terminar
    - Normalmente!

Modelo Waterfall

- Problemas
  - Divisão inflexível do projecto em diferentes etapas
    - Dificulta a resposta a alterações de requisitos
      - Apenas apropriado quando os requisitos são bem conhecidos desde o inicio e as alterações serão limitadas durante todo o processo
      - Poucos sistemas têm requisitos estáveis
  - Utilização (normalmente):
    - para sistemas grandes,
    - desenvolvidos por equipas grandes
    - Em diferentes locais (geográficos) diferentes
      - Processos baseados em planos ajudam a coordenar o trabalho

Modelo incremental - overview



Modelo incremental

- Vantagens
  - Custos para incluir alterações de requisitos é reduzido
    - Menos análise e documentação
  - Mais fácil obter *feedback* do cliente relativo ao desenvolvimento que está a ser feito
    - Clientes podem analisar demonstrações do software e perceber o que já está implementado
  - Entrega e *deployment (instalação, colocação em funcionamento)* mais rápido de software utilizável
    - Clientes podem usar o software mais cedo

Modelo incremental

- Problemas
  - O processo não é “visível”
    - Difícil de medir o progresso
      - Software desenvolvido de forma rápida
      - Pouca documentação
  - Estrutura do software degrada-se a cada incremento
    - Incorporar novas funcionalidades torna-se cada vez mais difícil
    - “Solução”
      - *Refactoring*

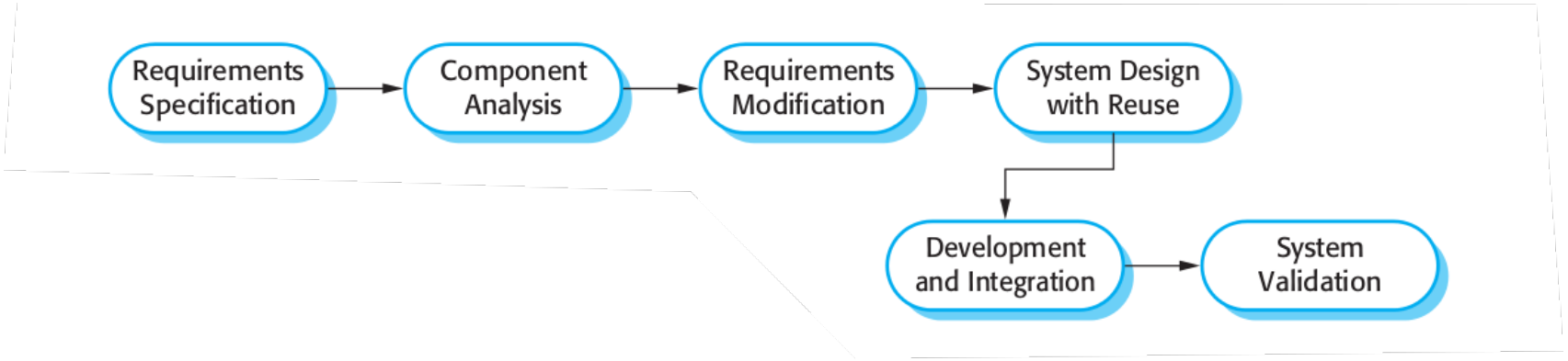
Modelos baseados em reutilização

- Integração de software/sistemas existentes

- Etapas
  - Especificação dos requisitos
  - Análise de componentes
  - Alteração de requisitos
  - Desenho do sistema
    - recorrendo à reutilização de software
  - Desenvolvimento e integração
  - Validação

- Standard em muitos sistemas

Modelos baseados em reutilização - overview



Modelos baseados em reutilização

- Tipos de componentes disponíveis
  - *Web services*
    - disponíveis para serem usados remotamente (ou localmente)
  - Colecções de objectos
    - funcionalidades especificas
    - Exemplos: bibliotecas diversas
  - Sistemas standalone
    - Configuráveis para diferentes ambientes

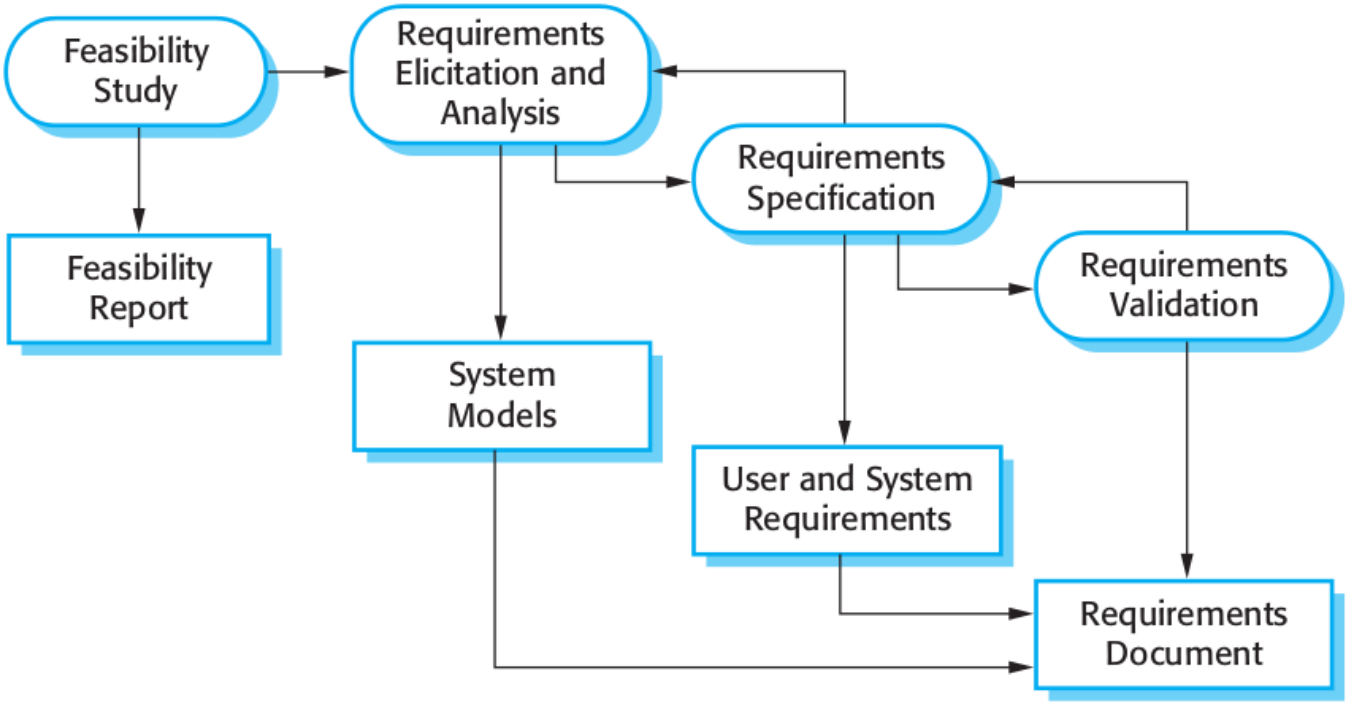
Actividades

- Processos de software reais
  - Sequências entrelaçadas/*imbricadas*
    - Actividades técnicas, colaborativas e de gestão
    - Objectivo
      - Especificar, desenhar, implementar e testar software
- Actividades básicas
  - Especificação, desenvolvimento, validação e evolução
  - Organizadas de forma diferente em diferentes processos
    - Waterfall: sequência
    - Incremental: entrelaçadas

Especificação de software

- Processo para especificar
  - Funcionalidades do sistema
  - Restrições
    - sistema e desenvolvimento
- Processo de engenharia de requisitos
  - Estudo de viabilidade
    - Técnico e financeiro
  - Análise dos requisitos
    - O que cada interessado necessita ou espera do sistema
  - Especificação dos requisitos
    - Definir os requisitos em detalhe
  - Validação dos requisitos
    - Verificar a validade dos requisitos

Processo de engenharia de requisitos



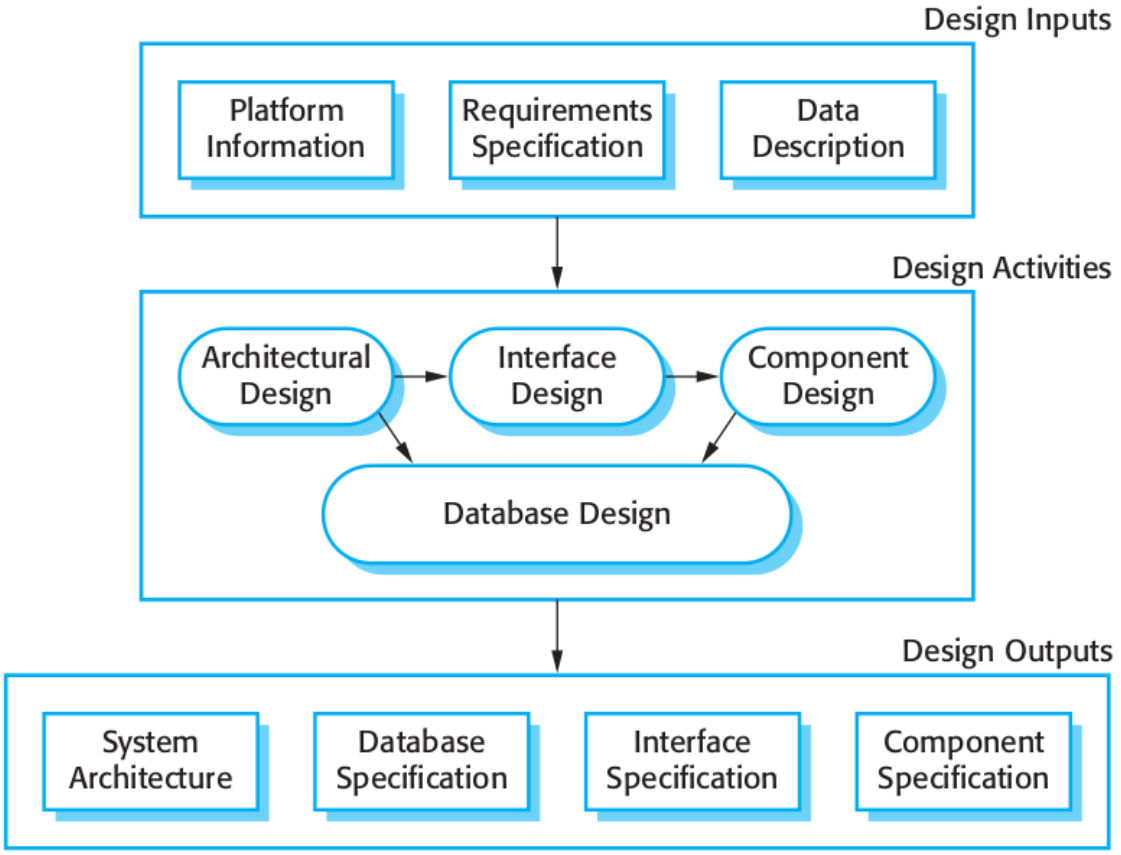
Desenho e implementação

- Converter a especificação num sistema executável/usável

- Tarefas
  - Desenho do software
    - Desenhar a estrutura que satisfaz a especificação
  - Implementação
    - “Traduzir” a estrutura num programa executável

- Desenho e implementação
  - Intrinsecamente relacionadas
  - Intercaladas

Processo de desenho - overview



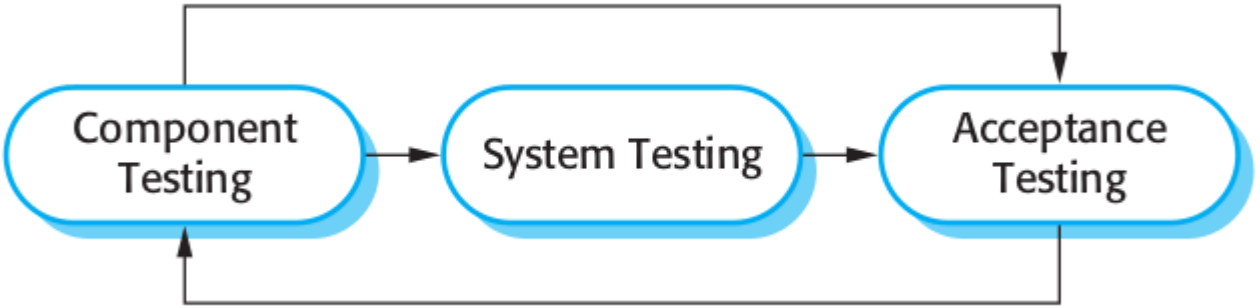
Processo de desenho

- Actividades
  - Desenho da arquitectura
    - Arquitectura global do sistema
    - Componentes principais (componentes ou módulos)
      - Relações entre si
  - Desenho do interface
    - entre os diversos componentes
  - Desenho dos componentes
    - de cada componente
  - Desenho da base de dados
    - Estrutura dos dados
    - Representação dos dados na base de dados

Validação de software

- Objectivo
  - Validar o software
- Métodos
  - Verificação e validação
  - Testes
- Verificação e validação
  - Demonstrar que o software está de acordo com os requisitos
- Testes
  - Executar o sistema
  - Casos derivados das especificações e dos dados reais
  - Encontrar erros
- Actividade de verificação e validação é normalmente juntamente com os testes

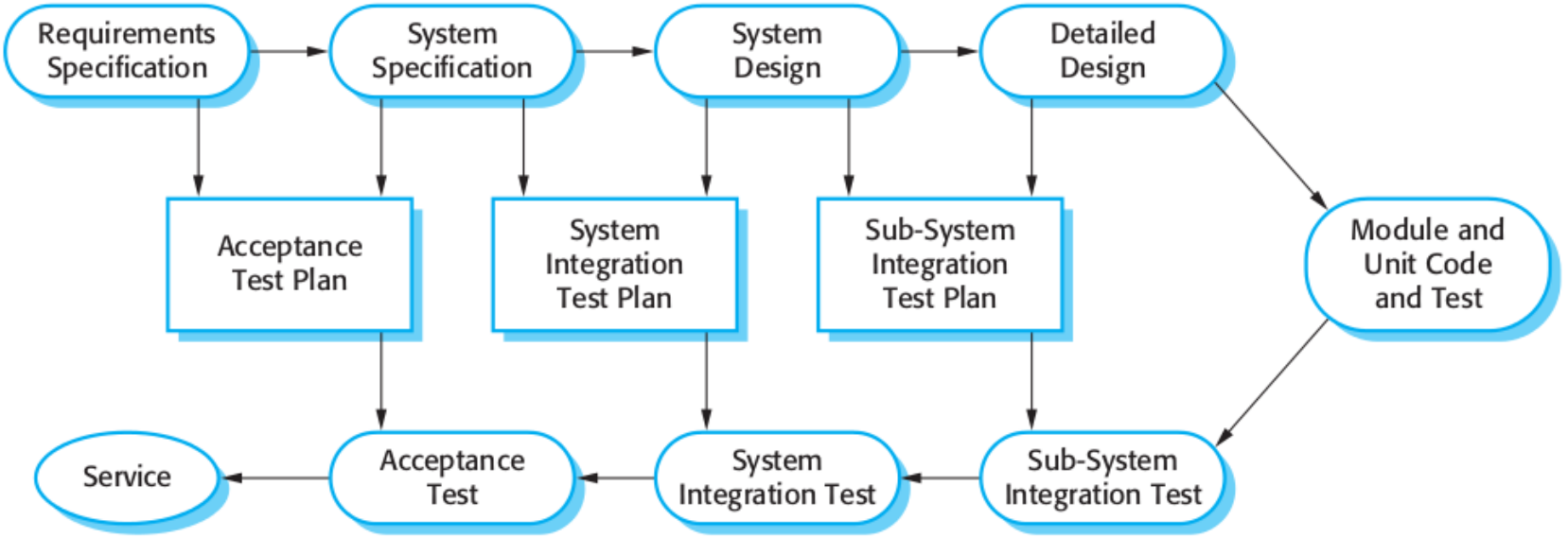
Etapas dos testes



Etapas dos testes

- Testes de desenvolvimento ou de componentes
  - Testes aos componentes individuais
  - Componentes: funções, objectos ou grupos (de funções e objectos)
- Testes de sistema
  - Testar o sistema como um só
- Testes de aceitação
  - Testes ao sistema usando dados do cliente
  - Verificar se está de acordo com as necessidades do cliente

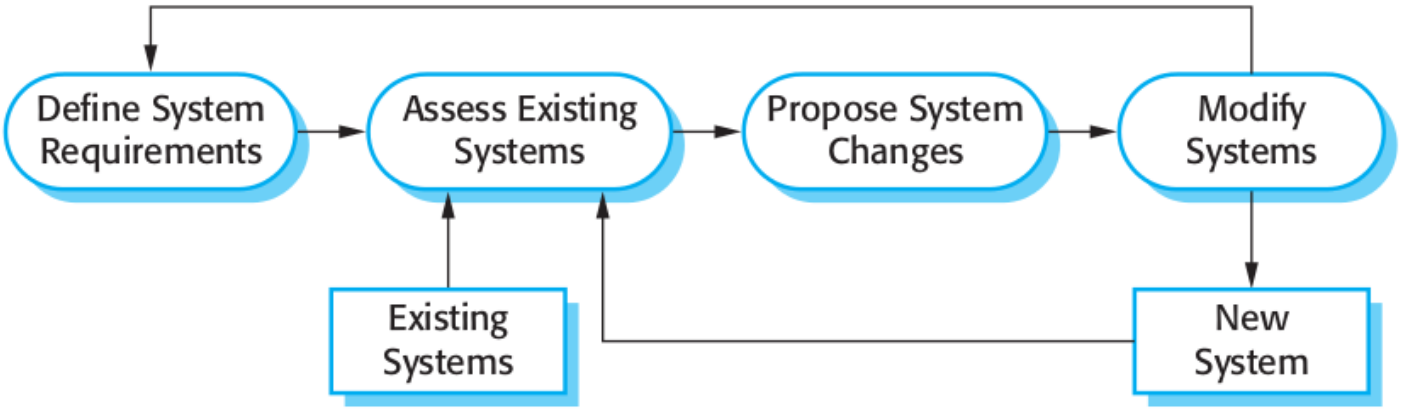
Etapas de testes - processo baseado em planos



Evolução de software

- Software é flexível e altera-se (evolui)
  - Regras do negócio mudam
  - Requisitos mudam
  - Software tem de adaptar-se
- Desenvolvimento e evolução (manutenção)
  - Actividades consideradas distintas
  - Evolução
    - Continuação do desenvolvimento
  - Grande parte dos sistemas não são feitos completamente de raiz
    - Evolução tem um papel importante

Overview


Define System Requirements
Assess Existing Systems
Propose System Changes
Modify Systems
Existing Systems
New System

Entrega incremental

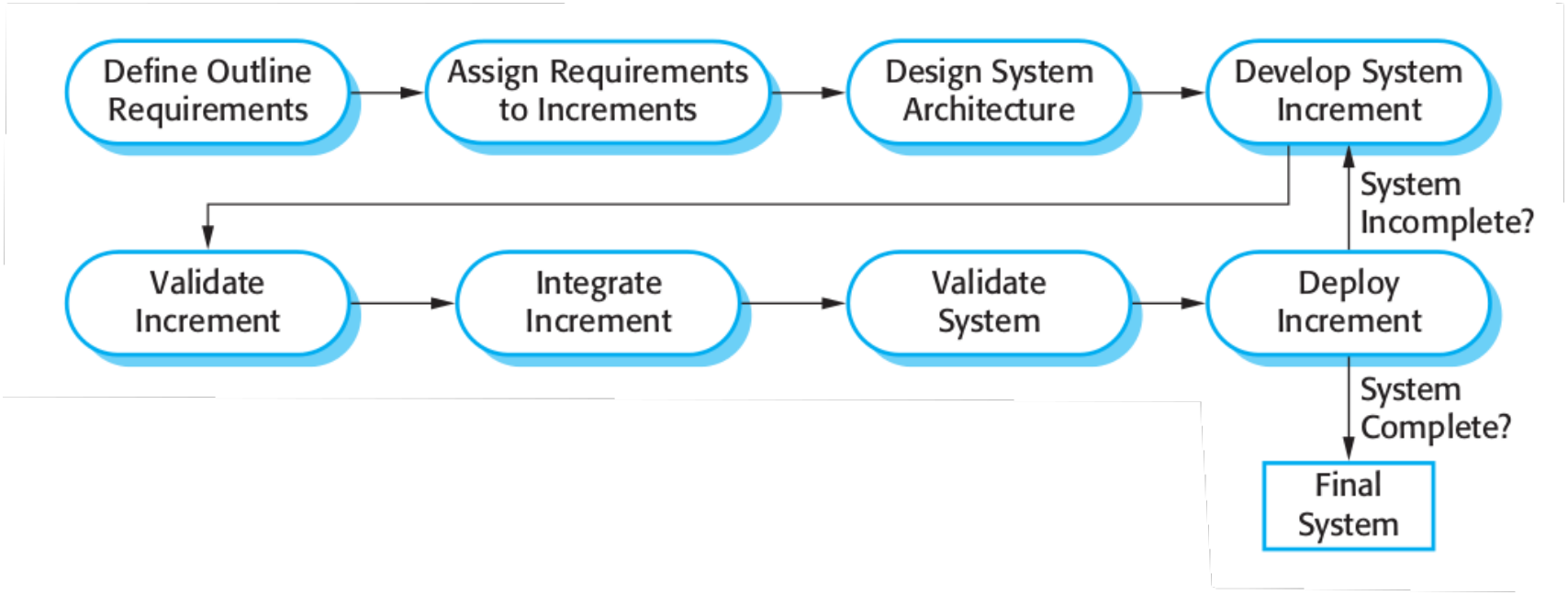
- Sistema entregue por fases
  - Desenvolvimento e entrega são divididos em “incrementos”
  - Cada incremento
    - Introduz uma funcionalidade (ou parte)
- Prioridades
  - Requisitos do utilizador têm maior prioridade.
    - Incluídos nos primeiros incrementos
- Em cada incremento
  - Requisitos são desse incremento são “congelados”
    - Requisitos para os incrementos seguintes continuam a evoluir

Desenvolvimento e entrega incremental

- Desenvolvimento incremental
  - Desenvolver o sistema por partes/incrementos
  - Avaliar cada incremento antes de passar para o próximo
  - Prática comum em processos ágeis
  - Avaliação pode ser feita pelo cliente
- Entrega incremental
  - Permite fazer o *deploy* de incrementos
    - Usáveis pelo utilizador final
    - Avaliação mais real sobre a utilização prática do sistema

Entrega incremental

Overview



Entrega incremental

Vantagens

- A cada incremento podem ser entregues novas funcionalidades ao cliente
  - Sistema está disponível mais cedo
- Incrementos iniciais podem servir de protótipos
  - Podem ajudar a identificar requisitos para os próximos incrementos
- Menor risco do projecto falhar
- Os serviços/funcionalidades mais importantes tendem a ser mais testados
  - São implementados nas fases iniciais

Desvantagens/problemas

- Muitos sistemas necessitam de um conjunto base de funcionalidades que é comum a todo o sistema
  - Como os requisitos apenas são especificados em detalhe quando o incremento vai ser implementado, pode tornar-se difícil identificar as funcionalidades comuns
- Especificação desenvolvida juntamente com o software
  - Pode ser um conflito para algumas empresas
    - Onde a especificação completa do sistema pode fazer parte do “contrato” para desenvolver o sistema

Bibliografia

- **Software Engineering, Ian Sommerville, 9th Edition, Addison-Wesley, 2010. Capítulo 2.**