

Processos de Software

Gidevaldo Novais
(gidevaldo.vic@ftc.br)

Riscos

- Muitos problemas no desenvolvimento de software provêm de riscos
- Seriam problemas potenciais que poderão ocorrer em um futuro próximo
- Outras engenharias usam métodos de previsão de riscos

Fontes de Riscos

- Ferramentas inadequadas
- Melhores profissionais deixam projeto
- Requisitos errados
- Projeto errado
- Equipamento esperado não chega
- Documentos ambíguos
- Cronograma estimado é inatingível
- Custo estimado está incorreto

Principais Áreas de Riscos

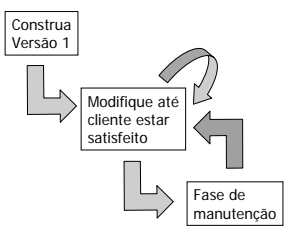
- Pessoal
- Cronograma e Custo
- Funcionalidade do Sistema
 - Falta de entendimento da aplicação
 - Análise de requisitos inadequada
- Estabilidade dos Requisitos
 - Cliente tenta alterar requisitos o tempo todo
- Qualidade de Componentes Externos
- **DIFICULDADE EM ANTECIPAR TUDO!!!**

Modelos de ciclo de vida

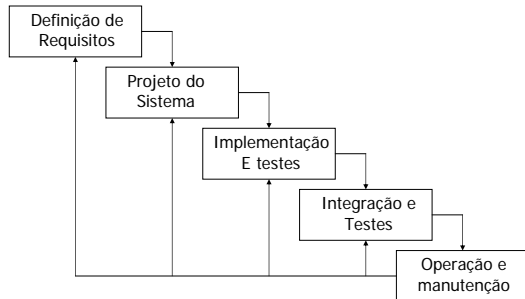
- Construa e Conserte (Code-and-Fix, Build-and-Fix)
- Cascata ("Waterfall")
 - Cascata Modificado
- Prototipação
- Espiral
- Formal
- Baseado em reuso

Construa e Conserte

- Desvantagens
 - Sem especificação
 - Sem projeto
 - Não gerencia riscos
- Vantagens
 - Baixa sobrecarga de desenvolvimento
 - Não precisa de especialização
- Para sistemas muito pequenos



Modelo Cascata



Características do Modelo Cascata

■ Desvantagens

- Partição inflexível do projeto em estágios distintos
- Dificuldade para lidar com as mudanças nos requisitos do sistema
- Não gerencia riscos

■ Vantagens

- Orientado a documentação
- Manutenção é mais simples

Características do Modelo Cascata*

■ Desvantagens

- Milestones podem tornar-se ambíguos
- Atividades em paralelo podem levar a problema de comunicação, suposições errôneas e ineficiência

■ Vantagens

- Atividades podem ser sobrepostas
- Mais fácil de lidar com mudanças nos requisitos
- Capacidade para gerenciar riscos

Processo Evolucionário

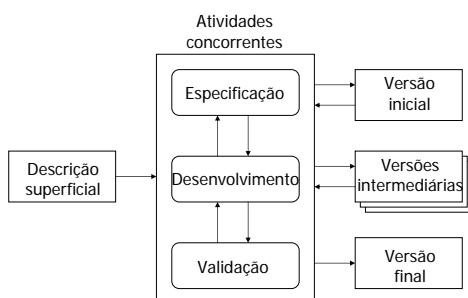
■ Desenvolvimento exploratório (Protótipo evolucionário)

- Construa, avalie e evolua para produto
 - Trabalhar com os clientes até se obter o produto final a partir de uma especificação bem-definida e completa do sistema.

■ Protótipo descartável

- Construa, use e descarte
 - Obter requisitos do cliente. Inicia-se com uma especificação incompleta e simples do sistema

Processo Evolucionário



Perspectivas

■ Desvantagens

- Falta de visibilidade do processo
- Sistemas são geralmente mal estruturados
- Conhecimentos específicos (ling. de prot. rápida) podem ser necessários

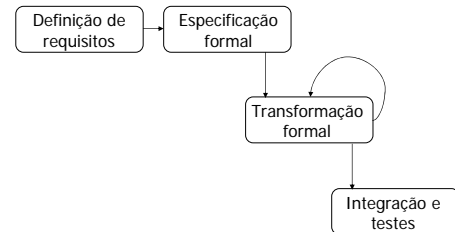
■ Aplicações

- Sistemas interativos de pequeno e médio porte
- Partes de sistemas grandes (interface com usuário)
- Sistemas com vida útil curta

Processo Formal

- Transformação de uma especificação formal em um programa executável
- Pode ser empregado de duas formas:
 - Através de um cálculo de refinamentos
 - Implementação é derivada por construção, garantindo correteza
 - Através de passos de refinamento
 - Versão mais concreta do sistema é proposta e depois deve-se verificá-la em relação a anterior

Processo Formal



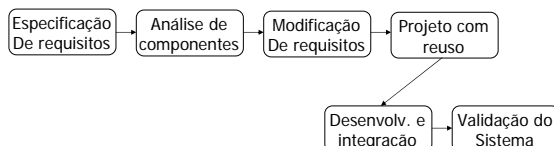
Perpectivas

- Desvantagens
 - Necessita de profissionais altamente qualificados para aplicar a técnica
 - Alguns aspectos ainda são difíceis de especificar (GUI)
- Vantagens
 - Garantia de segurança e confiabilidade
- Aplicações
 - Sistemas críticos e complexos

Processo baseado em Reuso

- Baseado no reuso sistemático de componentes existentes na própria empresa ou no mercado
- Estágios do processo
 - Especificação de requisitos
 - Análise dos componentes
 - Modificação dos requisitos
 - Projeto do sistema com reuso
 - Desenvolvimento e integração
 - Validação
- Essa abordagem está se tornando cada vez mais importante, mas ainda faltam casos experimentais bem-sucedidos

Processo baseado em Reuso



Modelos Evolutivos

- São iterativos
- Desenvolvendo novas versões ...
- Modelo Espiral
 - acopla a natureza iterativa do modelo de prototipação com os aspectos controlados e sistemáticos do modelo sequencial
- Modelo Incremental
 - combina elementos do modelo sequencial com a filosofia iterativa do modelo de prototipação, liberação por incrementos

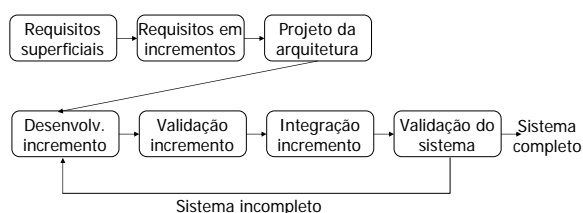
Processo Iterativo

- Os requisitos do sistema SEMPRE mudam durante o desenvolvimento
- Iteração pode ser aplicado a qualquer um dos processos de desenvolvimento
- Duas abordagens são destacadas:
 - Desenvolvimento incremental
 - Desenvolvimento em espiral

Desenvolvimento Incremental

- Ao invés de entregar o sistema completo, divide-se o sistema em partes de tal forma a cada entrega corresponder a alguma funcionalidade do sistema
- Requisitos do usuário são priorizados e os quanto maior a prioridade mais cedo tal funcionalidade deve ser entregue
- Uma vez que o desenvolvimento de um incremento seja iniciado, esses requisitos são fixados enquanto que os posteriores são deixados serem modificados

Desenvolvimento Incremental



Modelo Incremental

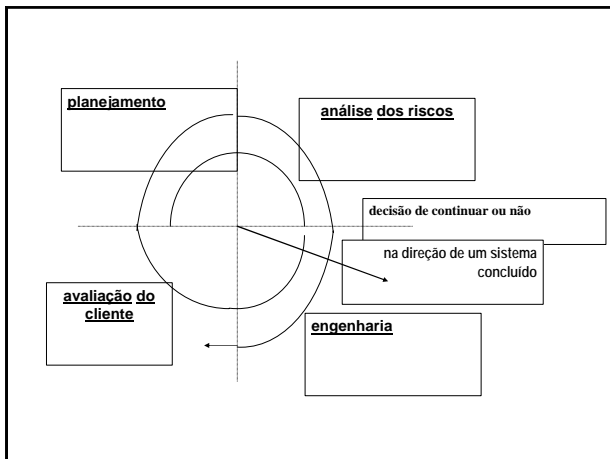
- Cada sequência linear produz uma liberação por incremento do software
- Exemplo: Processador de Texto
- Primeiro incremento: núcleo do produto
- Em cada incremento: entrega de um produto operacional

Vantagens

- Solicitações dos clientes são entregues a cada incremento. Assim, as funcionalidades são entregues o mais cedo possível
- Incrementos iniciais servem de protótipo para obter requisitos para incrementos posteriores
- Diminuição do risco de falha de todo o projeto
- Serviços de maior prioridade tendem a receber maior ênfase em testes

Ciclo de Vida em Espiral

- Engloba as melhores características do ciclo de vida Clássico como o da Prototipação, adicionando um novo elemento: a **ANÁLISE DOS RISCOS**
- Segue a abordagem de passos sistemáticos do *Ciclo de Vida Clássico* incorporando-os numa estrutura iterativa que reflète mais realisticamente o mundo real
- Usa a *Prototipação*, em qualquer etapa da evolução do produto, como mecanismo de redução de riscos



Atividades do Ciclo de Vida em Espiral

- 1- PLANEJAMENTO: determinação dos objetivos, alternativas e restrições
- 2- ANÁLISE DE RISCO: análise das alternativas e identificação / resolução dos riscos
- 3- CONSTRUÇÃO: desenvolvimento do produto no nível seguinte
- 4- AValiação DO CLIENTE: avaliação do produto e planejamento das novas fases

Comentários sobre o Ciclo de Vida em Espiral

- ✎ É uma abordagem realística para o desenvolvimento de software em grande escala.
- ✎ Usa uma abordagem que capacita o desenvolvedor e o cliente a entender e reagir aos riscos em cada etapa evolutiva.
- ✎ Pode ser difícil convencer os clientes que uma abordagem "evolutiva" é controlável.
- ✎ Exige considerável experiência na determinação de riscos e depende dessa experiência para ter sucesso.

Comentários sobre o Ciclo de Vida em Espiral

- ☐ A cada iteração ao redor da espiral, versões progressivamente mais completas do software são construídas
- ☐ o modelo é relativamente novo e não tem sido amplamente usado

Características do Modelo Espiral

- Desvantagens
 - ☐ Bem aplicado somente a sistemas de larga escala
 - ☐ Sistemas devem ser produtos internos da empresa
- Vantagens
 - ☐ Fácil de decidir o quanto testar
 - ☐ Não faz distinção entre desenvolvimento e manutenção

Extreme Programming (XP)

- Abordagem meio controversa
- Baseado em histórias (características que clientes desejam)
- Estima duração e custo de cada história
- Seleciona histórias para próxima fase
- Cada fase é dividida em tarefas
- Casos de testes são escritos para tarefas primeiramente
- Programação em pares
- Integração contínua das tarefas

Engenharia de Software *uma visão genérica*

O processo de desenvolvimento de software contém 3 fases genéricas, independentes do modelo de engenharia de software escolhido:

- ☆ DEFINIÇÃO
- ☆ DESENVOLVIMENTO
- ☆ MANUTENÇÃO

Engenharia de Software *uma visão genérica*

FASE DE DEFINIÇÃO: “o *quê*” será desenvolvido

- **Análise do Sistema:** define o papel de cada elemento num sistema baseado em computador, atribuindo em última análise, o papel que o software desempenhará.
- **Planejamento do Projeto de Software:** assim que o escopo do software é estabelecido, os riscos são analisados, os recursos são alocados, os custos são estimados, e tarefas e programação de trabalho são definidas.
- **Análise de Requisitos:** o escopo definido para o software proporciona uma direção, mas uma definição detalhada do domínio da informação e da função do software é necessária antes que o trabalho inicie.

Engenharia de Software *uma visão genérica*

FASE DE DESENVOLVIMENTO: “como” o software vai ser desenvolvido

- **Projeto de Software:** traduz os requisitos do software num conjunto de representações (algumas gráficas, outras tabulares ou baseadas em linguagem) que descrevem a estrutura de dados, a arquitetura do software, os procedimentos algorítmicos e as características de interface
- **Codificação:** as representações do projeto devem ser convertidas numa linguagem artificial (a linguagem pode ser uma linguagem de programação convencional ou uma linguagem não procedimental) que resulte em instruções que possam ser executadas pelo computador
- **Realização de Testes do Software:** logo que o software é implementado numa forma executável por máquina, ele deve ser testado para que se possa descobrir defeitos de função, lógica e implementação

Engenharia de Software *uma visão genérica*

FASE DE MANUTENÇÃO: concentra-se nas “mudanças” que ocorrerão depois que o software for liberado para uso operacional

- ⇒ Correção
- ⇒ Adaptação
- ⇒ Melhoramento Funcional

Engenharia de Software *uma visão genérica*

Correção: mesmo com as melhores atividades de garantia de qualidade de software, é provável que o cliente descubra defeitos no software. A manutenção *corretiva* muda o software para corrigir defeitos.

Adaptação: com o passar do tempo, o ambiente original (por exemplo a CPU, o sistema operacional e periféricos) para o qual o software foi desenvolvido provavelmente mudará. A manutenção *adaptativa* muda o software para acomodar mudanças em seu ambiente.

Engenharia de Software *uma visão genérica*

Melhoramento Funcional: a medida que o software é usado, o cliente/usuário reconhecerá funções adicionais que oferecerão benefícios.

A *manutenção perfectiva* estende o software para além de suas exigências funcionais originais.

Engenharia de Software

uma visão genérica

ATIVIDADES DE PROTEÇÃO as fases e etapas correlatas descritas são complementadas por uma série de atividades de proteção.

Revisões: efetuadas para garantir que a qualidade seja mantida à medida que cada etapa é concluída.

Documentação: é desenvolvida e controlada para garantir que informações completas sobre o software estejam disponíveis para uso posterior.

Controle das Mudanças: é instituído de forma que as mudanças possam ser aprovadas e acompanhadas.

Conclusão

ENGENHARIA DE SOFTWARE

pode ser vista como uma abordagem de desenvolvimento de software elaborada com disciplina e métodos bem definidos.

.....“a construção por múltiplas pessoas de um software de múltiplas versões” (Parnas 1987)

Bibliografia

- Capítulo 3 - Engenharia de Software - I.Sommerville