

# Introdução à Engenharia de Software

Gidevaldo Novais  
(gidevaldo.vic@ftc.br)

## Introdução à Engenharia de Software

- o Objetivo
  - Depois desta aula você terá uma noção geral do que é a engenharia de software e dos seus objetivos e conceitos básicos relacionados.

## Introdução à Engenharia de Software

- o Leitura recomendada
  - Capítulo 1 e 2 - livro Engenharia de Software (Ian Sommerville)

## O que é a Engenharia de Software?

*Estudo ou aplicação de abordagens sistemáticas, econômicas e quantificáveis para o desenvolvimento, operação e manutenção de software de qualidade.*

## Objetivos da Engenharia de Software

- o Qualidade de software
- o Produtividade no desenvolvimento, operação e manutenção de software
- o Qualidade versus Produtividade
- o Permitir que profissionais tenham controle sobre o desenvolvimento de software dentro de custos, prazos e níveis de qualidade desejados

## Qualidade de Software (um exemplo para o Varejo)

- o Correto
  - A loja não pode deixar de cobrar por produtos comprados pelo consumidor
- o Robusto e altamente disponível
  - A loja não pode parar de vender
- o Eficiente
  - O consumidor não pode esperar
  - A empresa quer investir pouco em recursos computacionais (CPU, memória, rede)

## Qualidade de Software (um exemplo para o Varejo)

- o Amigável e fácil de usar
  - A empresa quer investir pouco em treinamento
- o Altamente extensível e adaptável
  - A empresa tem sempre novos requisitos (para ontem!)
  - A empresa quer o software customizado do seu jeito (interface, teclado, idioma, moeda, etc.)
- o Reusável
  - Várias empresas precisam usar partes de um mesmo sistema

## Qualidade de Software (um exemplo para o Varejo)

- o Aberto, compatível, de fácil integração com outros sistemas
  - A empresa já tem controle de estoque, fidelização, etc.
- o Portável e independente de plataforma (hw e sw)
  - A empresa opta por uma determinada plataforma
- o Baixo custo de instalação e atualização
  - A empresa tem um grande número de PDVs

## Produtividade

- o Custo de desenvolvimento reduzido
  - A empresa consumidora quer investir pouco em software
  - A empresa produtora tem que oferecer “software barato”
- o Tempo de desenvolvimento reduzido
  - Suporte rápido às necessidades do mercado

## “Software Barato”

*Nem tanto resultado de baixos custos de desenvolvimento, mas principalmente da distribuição dos custos entre vários clientes.*

*Reuso, extensibilidade e adaptabilidade são essenciais para viabilizar tal distribuição.*

## Relevância da Engenharia de Software

- o Qualidade de software e produtividade garantem:
  - Disponibilidade de serviços essenciais
  - Segurança de pessoas
  - Competitividade das empresas
    - Produtores
    - Consumidores

## Mas, na realidade, temos a Crise de Software...

- o 25% dos projetos são cancelados
- o o tempo de desenvolvimento é bem maior do que o estimado
- o 75% dos sistemas não funcionam como planejado
- o a manutenção e reutilização são difíceis e custosas
- o os problemas são proporcionais a complexidade dos sistemas

## Causas da Crise de Software

- o Complexidade dos sistemas
- o Dificuldade de formalização
- o Má qualidade dos métodos, linguagens, ferramentas, processos, e modelos de ciclo de vida
- o Falta de qualificação técnica

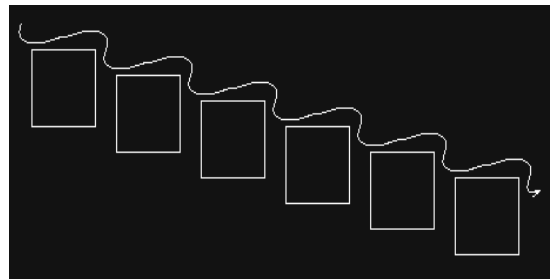
## Elementos e Atividades da Engenharia de Software

- o Elementos
  - Modelos do ciclo de vida do software
  - Linguagens
  - Métodos
  - Ferramentas
  - Processos
- o Atividades
  - Modelagem do negócio
  - Elicitação de requisitos
  - Análise e Projeto
  - Implementação
  - Testes
  - Distribuição
  - Planejamento
  - Gerenciamento
  - Gerência de Configuração e Mudanças
  - Manutenção

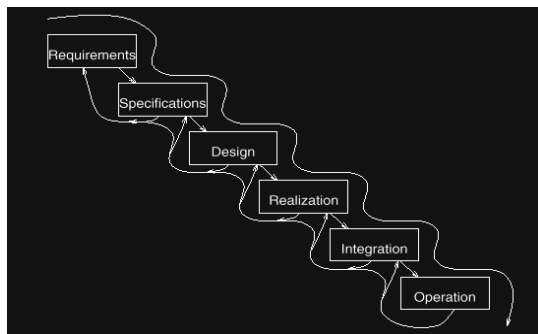
## Modelos do Ciclo de Vida de Software

- o Cascata
- o Espiral
- o Iterativo (do RUP)
- o Prototipagem evolucionária
- o Força bruta, gambiarra, ...

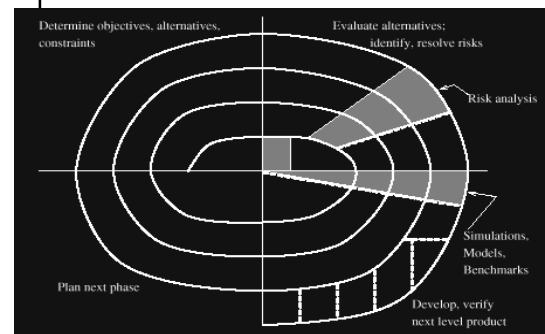
## Modelo Cascata



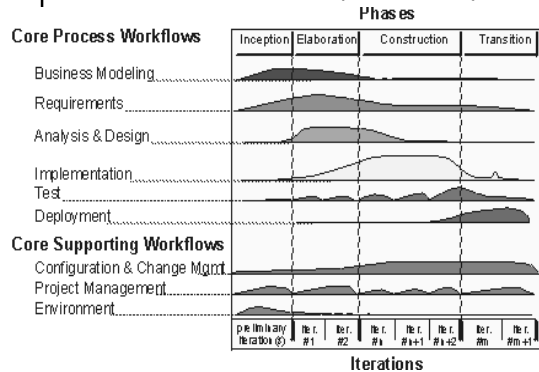
## Modelo Cascata na Prática



## Modelo Espiral



## Modelo Iterativo (do RUP)



## Linguagem

- o Notação com sintaxe e semântica bem definidas
  - com representação gráfica ou textual
- o Usada para descrever os artefatos gerados durante o desenvolvimento de software
- o Exemplos: UML, Java

## Método

- o Descrição sistemática de como deve-se realizar uma determinada atividade ou tarefa
- o A descrição é normalmente feita através de padrões e guias
- o Exemplos: Booch, OMT, ...

## Ferramenta

- o Provê suporte computacional a um determinado método ou linguagem
- o Ambiente de desenvolvimento: conjunto de ferramentas integradas (CASE)
- o Exemplos: Rational Rose, Inprise JBuilder

## Processo

- o Conjunto de atividades
  - bem definidas
  - com responsáveis
  - com artefatos de entrada e saída
  - com dependências entre as mesmas e ordem de execução
  - com modelo de ciclo de vida

## Metodologia

- o Conjunto de métodos + processo

## ● ● ● | Tópicos a serem abordados...

- O presente curso abordará os seguintes tópicos sob a perspectiva em cascata:
  - Planejamento e Gerenciamento de Software
  - Requisitos de Software
  - Análise e Projeto de Software
  - Codificação de Software
  - Depuração e Testes
  - Qualidade e Inspeção

## ● ● ● | Planejamento e Gerenciamento

- O que e como faremos?
- Quem fará o quê?
- Quanto tempo levaremos?
- O que poderá dá errado (riscos)?
- O que usaremos?
- Quanto custará?
- Como estamos indo?
- Estamos documentando (tempo, etc.)?

## ● ● ● | Requisitos

- Como elicitá-los?
  - O que faremos “exatamente”?
- Documentar é preciso ...
- Como apresentar?
  - Texto + UML
- Usar o documento também ...
  - Como fonte para Análise
  - Como contrato perante o cliente

## ● ● ● | Análise

- Identificar elementos constituintes do sistema (domínio do problema)
  - Classes, associações, etc.
- Documentar através de modelo abstrato
- Descrever exatamente O QUE faremos
- Investigar propriedades do sistema
- Comparar com documento de requisitos e validar sistema

## ● ● ● | Projeto

- Identificar elementos constituintes do sistema (domínio da solução)
  - Módulos, classes, associações, etc.
- Documentar através de modelo concreto
  - Algoritmos e tecnologias existentes, etc.
- Descrever exatamente COMO faremos
  - UML + Comentários
- Comparar com documento de análise
  - Verificar sistema

## ● ● ● | Codificação

- Que linguagens usaremos?
- Que ferramentas usaremos?
- Como faremos a codificação?
- Codificar através do modelo de projeto
- Garantir pré e pós-condições
- Integração ...

## ● ● ● | Depuração e Testes

- Avaliando artefatos de forma incremental
  - Métodos, classes, módulos, requisitos não-funcionais, etc.
- Avaliando pela especificação (teste da caixa preta) ou pelo código (teste da caixa branca)?
- Comparar com documentos anteriores
- Documentando erros e calculando produtividade e qualidade

## ● ● ● | Qualidade e Inspeção

- Quanto bom foi o desenvolvimento?
- Qual a quantidade de erros encontrados?
- Qual a produtividade da equipe?
- Esquecemos de verificar algum aspecto do sistema?
- Os documentos estão atualizados?

## ● ● ● | Exercício

- Faça uma **pesquisa na Internet sobre Engenharia de Software** e organize uma home page ou apresentação sobre o assunto.
- Na próxima aula você apresentará o resultado da sua pesquisa em 05 minutos.
- Estruture a home page da seguinte forma:
  1. Páginas sobre o assunto;
  2. Grupos de pesquisa;
  3. Cursos (inclusive cursos *online*);
  4. Síntese da pesquisa em texto