Uma Introdução à Engenharia de Software

Profs. Márcio Lopes Cornélio
Alexandre Vasconcelos

Slides originais elaborados por Ian Sommerville
O autor permite o uso e a modificação dos *slides* para fins didáticos

Apresentação da Disciplina

Aulas de Monitoria e Laboratório

- Exercício de conteúdos vistos em sala
  - Os monitores ajudarão/ministrarão as aulas práticas
- Ajuda no uso de ferramentas
- Acompanhamento das equipes

Avaliação da Disciplina

- Nota 1 (N1) = "Prova 1"*0,3
- Nota 2 (N2) = "Prova 2"*0,3
- Nota 3 (N3) = Projeto*0,4
- Nota Geral = N1+N2+N3

- Obs: Equipes com no máximo 5 alunos

Publicação dos projetos

- A entrega dos artefatos deve ser publicada na ferramenta Redmine e será monitorada pelo monitor de sua equipe
- Deve conter:
  - Equipe (nome, login)
  - Documentos desenvolvidos nos projeto
  - Cronograma atualizado
  - Atas de reuniões da equipe e com o monitor responsável pela equipe
  - Informações úteis a usuários finais (opcional)

Referências

- Básica
  - Ian Sommerville. Engenharia de Software - 9a edição - Versão em Português editada pela Pearson/Addison-Wesley, 2012.
- Extras
  - Carlo Ghezzi, Mehdi Jazayeri e Dino Mandrioli. Fundamentals of Software Engineering. Prentice-Hall. 1991.
  - Alain Abran, James W. Moore, Pierre Bourque e Robert Dupuis (eds.). Guide to the Software Engineering Book of Knowledge. IEEE Computer Society Press, 2004. Disponível online em: http://www.computer.org/portal/web/swebok/htmlformat
  - Pressman R. Engenharia de Software - 6a edição - McGraw-Hill Interamericana do Brasil. Material suplementar em inglês
  - PÁDUA, W. Engenharia de Software – Fundamentos, Métodos e Padrões. 3. Ed. Rio de Janeiro: LTC, 2009.
  - Pfleeger, S. Engenharia de Software – Teoria e Prática 4a edição – Pearson/Prentice-Hall, 2009.

Introdução

Motivação

- As economias de TODAS as nações desenvolvidas são dependentes de software.
- Cada vez mais sistemas são controlados por software.
- A engenharia de software se dedica às teorias, métodos e ferramentas para desenvolvimento de software profissional
  - Sistemas não-triviais
  - Com base em um conjunto de requisitos

Custos de software

- Os custos de software geralmente dominam os custos de sistemas computacionais.
  - Em geral, software custa mais que hardware
- Manter um software geralmente custa mais que desenvolvê-lo
- A engenharia de software dedica-se ao desenvolvimento de software com custos adequados
  - Respeitando o cronograma acordado
  - Satisfazendo as necessidades dos clientes
  - Minimizando o custo de manutenção

Definições

Deve-se entender por engenharia a ciência relacionada com o uso prático de conhecimentos científicos

Engenharia de Software - Disciplina gerencial e tecnológica que lida com a produção e manutenção sistemática de produtos de software desenvolvidos dentro de estimativas de custo e tempo

Objetivos da Engenharia de *Software*

- Obter software de qualidade
- Com produtividade no seu desenvolvimento, operação e manutenção
- Empregando profissionais que desenvolvam o software dentro de custos, prazos e níveis de qualidade controlados
- E, além disso, que obtenham o melhor custo-benefício possível entre Qualidade × Produtividade

Características da Engenharia de Software

- A Engenharia de Software se refere a software (sistemas) desenvolvidos por grupos ao invés de indivíduos
- usa princípios de engenharia ao invés de arte, e
- inclui tanto aspectos técnicos quanto não técnicos

O que é software?

- Programas de computador e artefatos associados
- Produtos de software podem ser:
  - **Genéricos** – desenvolvidos para serem vendidos para uma grande variedade de clientes (e.g., Excel e Word)
  - **Personalizados** – desenvolvidos para um único cliente de acordo com as suas especificações.
- Um software novo pode ser criado através de
  - desenvolvimento de novos programas;
  - configuração de sistemas de software genéricos; ou
  - reutilização de um software existente.

O que é engenharia de software?

- É uma disciplina relacionada com todos os aspectos da produção de software que propõe ferramentas, técnicas e processos para:
  - Entender com precisão qual é o problema (as necessidades associadas ao sistema que deve ser construído/modificado)
  - Produzir uma solução adequada para esse problema (um sistema pronto para usar, levando-se em consideração as necessidades das partes interessadas)
  - Levando-se em conta restrições de desenvolvimento e recursos disponíveis

Qual é a diferença entre engenharia de software e ciência da computação?

- A ciência da computação dedica-se à teoria e aos fundamentos;
  - Engenharia de software dedica-se aos aspectos práticos de desenvolvimento e de entrega de software

Qual é a diferença entre engenharia de software e engenharia de sistemas?

- Engenharia de sistemas:
  - Mais ampla
  - Muita ênfase em aspectos de hardware e infra-estrutura
    - Abstração do hardware
    - Organização física das partes do sistema
    - Aspectos de comunicação
  - Engloba a engenharia de software
- Os engenheiros de sistema estão envolvidos em diversas atividades da engenharia de software
  - Projeto da arquitetura
  - Elicitação e especificação de requisitos

O que é processo de software?

- Um conjunto estruturado de atividades, práticas, artefatos e ferramentas necessários para o desenvolvimento de um sistema de software
  - Especificação;
  - Desenvolvimento;
  - Validação;
  - Evolução.
- Exemplos: Processo Unificado (RUP), OpenUP, Programação Extrema (XP)

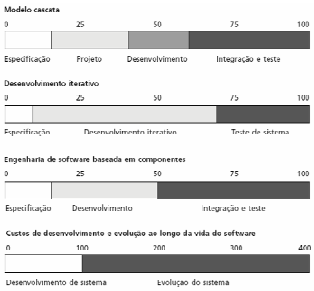
O que é processo de software? (continuação)

- Alguns elementos de um processo:
  - Modelos de sistema:
    - Modelos gráficos que podem/devem ser produzidos e as notações que devem ser empregadas;
    - Restrições aplicadas aos modelos de sistema;
  - Recomendações de boas práticas de projeto;
  - Atividades que devem ser seguidas em determinada ordem
  - Às vezes também prescrevem ferramentas
- Um processo adere a um ou mais modelos de processo

Quais são os custos da engenharia de software?

- 60% dos custos são custos de desenvolvimento
- 40% são custos de testes
- Para software sob encomenda, os custos de evolução normalmente excedem os de desenvolvimento.
- Os custos variam dependendo do tipo de sistema que está sendo desenvolvido e dos requisitos do sistema, tais como desempenho e confiabilidade
- A distribuição de custos depende do modelo de desenvolvimento que é usado.

Distribuição de custos nas atividades



O que é CASE (*Computer-Aided Software Engineering*)?

- Sistemas de software que se destinam a fornecer apoio automatizado para as atividades de desenvolvimento de software.
- Sistemas CASE são usados freqüentemente para apoiar um método específico
- *Upper-CASE*
  - Ferramentas para apoiar as atividades iniciais de processo de requisitos e de projeto;
- *Lower-CASE*
  - Ferramentas para apoiar as atividades finais tais como programação, *debugging* e teste.

Quais são alguns atributos de um bom software?

- O software deve fornecer a funcionalidade e o desempenho requeridos para o usuário e deve apresentar certas características
- Facilidade de manutenção
  - Deve ser fácil e barato fazer com que o sistema, depois de implantado, evolua para atender às necessidades dos clientes
- Confiabilidade
  - O software deve funcionar de maneira que o usuário possa depender dele. No caso de falha, não traz danos físicos ou econômicos
- Eficiência
  - O software deve usar o mínimo de recursos e prover máxima funcionalidade
- Usabilidade
  - O software deve ser compreensível e fácil de usar

Quais são os desafios-chave enfrentados pela engenharia de software?

- Heterogeneidade
  - Sistemas de software devem ser capazes de lidar com diferentes plataformas de hardware e ambientes de execução;
- Entrega
  - O sistema deve ser entregue ao cliente no menor tempo possível, com o menor custo possível;
- Confiança
  - O usuário deve poder justificadamente depositar sua confiança no sistema
- Escalabilidade
  - O sistema deve funcionar adequadamente mesmo quando um grande número de usuários está usando-o

Responsabilidade profissional e ética

- A engenharia de software envolve responsabilidades mais amplas do que simplesmente a aplicação de habilidades técnicas.
- Os engenheiros de software devem se comportar de modo honesto e eticamente responsável para serem respeitados como profissionais.
- O comportamento ético é mais do que simplesmente a sustentação de leis.

Questões de responsabilidade profissional

- Confidencialidade
  - Os engenheiros de software devem respeitar a confidencialidade de seus funcionários ou clientes, independentemente de ter ou não assinado um acordo formal.
    - Caso não aceitem essas condições, devem deixar isso expiícito para seus contratantes
- Competência
  - Os engenheiros não devem conscientemente aceitar um trabalho que esteja fora de sua competência.

Questões de responsabilidade profissional

- Direitos sobre propriedade intelectual
  - Desenvolvedores devem estar cientes das leis locais que regem o uso de propriedade intelectual, tais como patentes, direitos autorais, etc.
  - Eles devem tomar cuidado para assegurar que a propriedade intelectual dos funcionários e clientes seja protegida.

Exemplos de dilemas éticos

- Discordância, em princípio, das políticas da gerência sênior.
- Liberação de um sistema de segurança crítico sem finalizar o teste do sistema.
- Participação no desenvolvimento de sistemas de armamentos militares ou de sistemas nucleares.

Exemplo de um Processo Simplificado de Desenvolvimento

O Início de Tudo...

"A intenção do cliente é..."

O Mais Importante Aqui é...

A Idéia
é
Viável??

O Que Devo Fazer Exatamente?

Ou, em outras palavras, quais são os **requisitos** da aplicação?

Requisitos

- O Que devo fazer?
  - Funcionalidades
- Há restrições sobre as funcionalidades?
  - Limites de tempo, memória, etc.?
- Há restrições mais amplas?
  - Empresa, Governo, etc.?

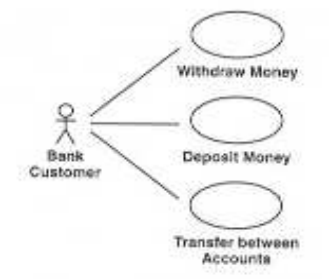
O que faço então?

"O documento de requisitos..."

Como apresentar ao Cliente?

"O cliente não vai ler 500 páginas de requisitos!!!"

Uma Figura Vale Mais Que ...



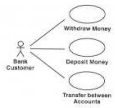
Mas Paralelamente ...

- "Precisamos saber quanto tempo levaremos para fazer nosso trabalho, quanto isso custará e o que pode nos atrapalhar... Precisamos Planejar!!!"
- Também precisamos controlar acessos, versões e mudanças (Gerência de configuração)

Estimando Esforço

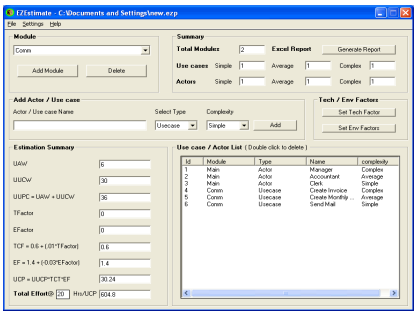
- Modelo de casos de uso pode ser usado para calcular estimativa
- Baseia-se em uma série de fatores que determinam a complexidade da aplicação
- Há ferramentas para realizar o cálculo

Estimando Esforço



X horas

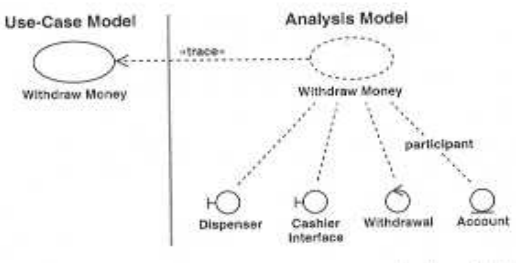
Estimando Esforço



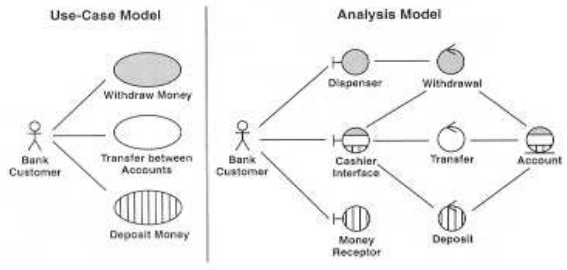
Iniciando a Solução...

"Temos que identificar em nossos requisitos, quais são os elementos essenciais para satisfazê-los..."

Iniciando a Solução...



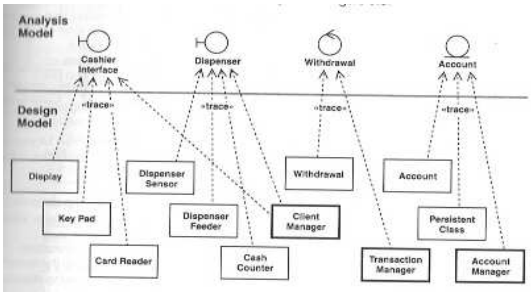
Iniciando a Solução...



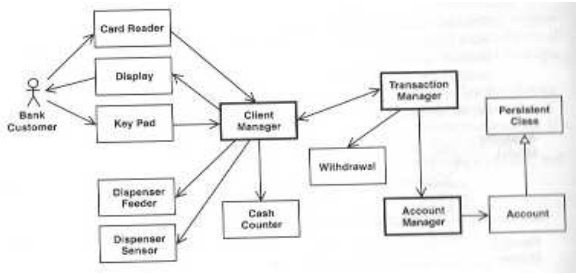
Sedimentando a Solução...

"A partir dos elementos essenciais, precisamos definir estratégias para satisfazê-los incluindo suas restrições..."

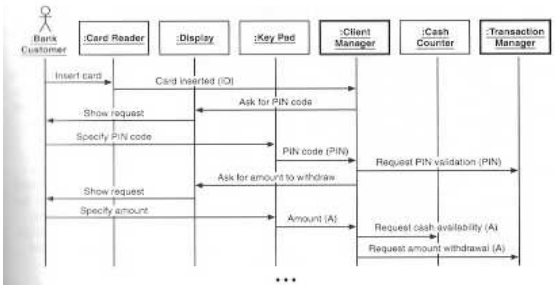
Sedimentando a Solução...



Sedimentando a Solução...



Sedimentando a Solução...



Operacionalizando a Solução...

"Com a solução definida, o passo final é operacionalizá-la. Isto é, codificá-la."

Classe Account...

```
public class Account {
  private int balance;
  ...
  void debit(int amount) {
    if(amount<=balance)
       balance = balance – amount;
    else throw new AccountException("...");
  }
...
}
```

Funciona???

"Com a implementação feita, podemos então executar os testes!!!"

Avaliando a qualidade

```
public class AccountTest extends TestCase {

  void testDebit() {
    Account acc = new Account(10);
    acc.debit(10);
    assertEquals(0, acc.getBalance());
  }
}
```