Engenharia de Sistemas de Informação I

Conceitos básicos

Marcos L. Chaim
Escola de Artes, Ciências e Humanidades – EACH
Universidade de São Paulo – USP

06-08-2007

Problema

- Software está em todo lugar:
  - no sistema de energia elétrica que você utiliza;
  - no telefone celular que tocou para acordá-lo de manhã;
  - no metrô, ônibus e talvez... no trem que você utilizou para chegar a EACH.
- Desafio: cite em alguma atividade econômica que não é apoiada por um sistema computacional.
- O que o futuro nos reserva: roupas com software? remédios inteligentes? Nanossistemas?

Problema

Algumas dúvidas:

- estamos sendo bem sucedidos nos softwares que construímos?
- quão difícil é construir software?
- como são desenvolvidos os softwares?

Problema

Jim Neighboors, pesquisador e consultor, SBES,Ouro Preto, 1991:

> *Obrigado por desenvolver software. Há maneiras mais fáceis de ganhar dinheiro.*

Afirmação ainda válida hoje...

Engenharia de Software

- Em 1968 a OTAN organizou a primeira conferência sobre Engenharia de Software em Garmish, Alemanha Ocidental.
- O termo, *Engenharia de Software*, é atribuído a Fritz Bauer.
- Mas o que é essa tal de *Engenharia de Software*?

  > É uma abordagem sistemática – baseada em processos, procedimentos, ferramentas e técnicas – para produzir sistemas computacionais que possuem requisitos de *qualidade* observáveis.

- EP não é software!

Engenharia de Software

Análise do problema × síntese da solução. Ver Figura 1

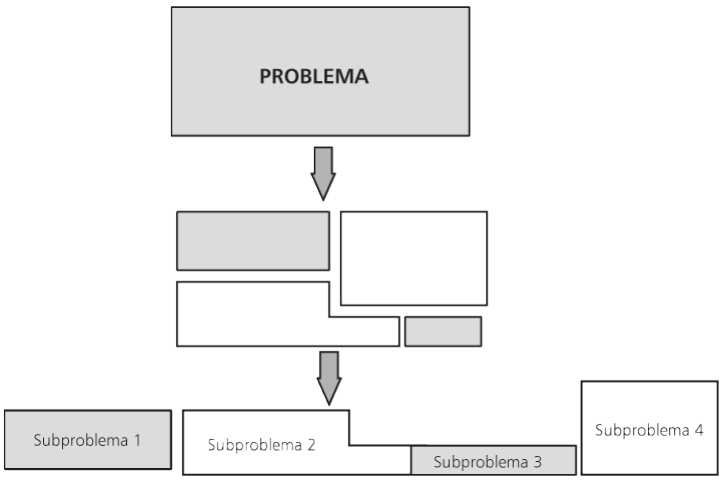

Figura 1: O processo de análise.

Engenharia de Software

Análise do problema × síntese da solução. Ver Figura 2

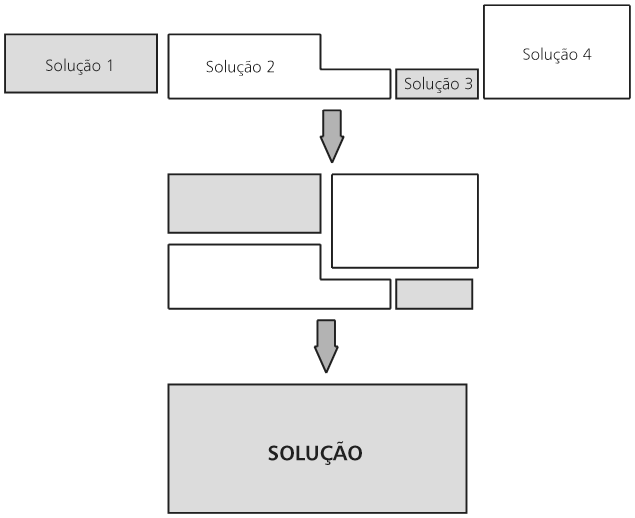

Figura 2: O processo de síntese.

Engenharia de Software

- *Processo*: seqüência de atividades a serem realizadas para construir um sistema de software.
- *Método ou técnica*: procedimento formal para a produção de um resultado.
- *Ferramenta*: instrumento ou sistema automatizado para realizar alguma coisa.
- *Procedimento*: receita de combinação de ferramentas e técnicas.
- *Paradigma*: estilo de fazer algo.

Engenharia de Software

- Ciência da Computação $\times$ Engenharia de Software.
- Paralelo semelhante: Física $\times$ Engenharia Civil.
- Cientista da computação: soluciona problemas genéricos. Exemplo: qual o melhor algoritmo? Como melhorá-lo? Ou desenvolve um algoritmo para um problema genérico ainda não solucionado.
- Engenheiro de Software: escolhe o melhor algoritmo, dentre os existentes, para resolver o problema do cliente; utiliza técnicas para reduzir o problema do cliente em subproblemas menores e mais gerenciáveis; utiliza técnicas para garantir a qualidade do produto.

Engenharia de Software

Ciência da Computação × Engenharia de Software. Ver Figura 3

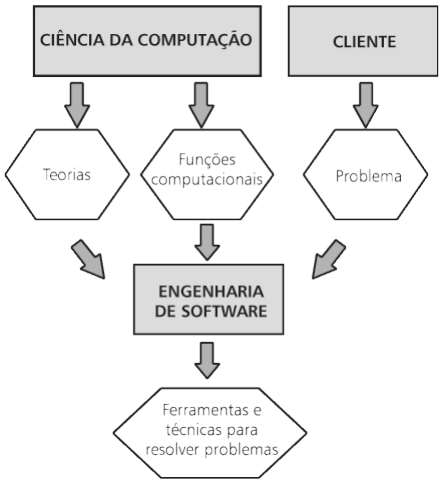

Figura 3: Relação Ciência da Computação e Engenharia de Software.

Engenharia de Software

Pergunta: Já temos uma Engenharia para produzir software?

- David Parnas[1]: engenharia de software é uma forma de engenharia. [2].
- Steve McConnell[2]: que a Engenharia de Software não é uma engenharia, mas deveria a ser. Por que não é? [2].
- Donald Knuth[3]: programação é uma arte. [2].
- Opinião do professor: a Engenharia de Software ainda é muito jovem (apenas 40 anos) e, apesar dos inúmeros progressos, os processos, procedimentos, técnicas e ferramentas ainda não estão maduros para garantir a qualidade e estimar prazos e custos com precisão, principalmente para sistemas novos.

[1] Criador do conceito de abstração de partes do software em módulos com interface e saída definida.
[2] Autor de livros sobre desenvolvimento de software.
[3] Grande teórico da computação que desenvolveu os conceitos para a análise de algoritmos e criador dos sistemas TeX e Metafont. Autor da série clássica de livros "A Arte da Programação de Computadores". Recebeu o Prêmio Turing em 1974.

O que é software

- Programa executável?
- Código fonte?
- Manual?
- Documentos: documento de requisitos, casos de uso, modelo conceitual, modelo de classes etc ?

Todos estes artefatos fazem parte do software. Dependendo da fase de desenvolvimento, o software é composto apenas por alguns artefatos. Por exemplo, durante o levatamento de requisitos o software é apenas o artefato *documento de requisitos*. À medida que ocorre o desenvolvimento mais artefatos são adicionados.

O que é um bom software

- Produto com *qualidade* e *utilidade* aceitáveis.
- Estes conceitos variam da aplicação: missão crítica $\times$ sistema de informação típico.
- Cada vez mais os sistemas estão se tornando de missão crítica, pois os negócios dependem deles. Exemplo: bolsa eletrônica de valores; comércio eletrônico.

Perspectivas de qualidade

- Visão transcendental: algo que podemos reconhecer, mas não definir. Exemplo: interface amigável.
- Visão do usuário: adequação propósito pretendido. Visão externa.
- Visão do fabricante: conformidade com especificação.
- Visão do produto: relação com as características inerentes ao produto (organização, eficiência etc.). Visão interna.
- Visão do mercado: dependência de quanto os consumidores estão dispostos a pagar.

Qualidade de produto

- Visão externa, do usuário: correção dos resultados; facilidade de uso; segurança; quantidade e tipos de falhas.
- Visão interna: facilidade de manutenção (é fácil fazer um programador escrever um programa que só ele entenda; difícil é fazer programas que um bom número de programadores entenda); facilidade para testar.
- Necessidade de medir a qualidade do produto.

Qualidade de processo

- Muitas atividades afetam a qualidade final do produto; se não forem bem realizadas, a qualidade será prejudicada.
- Muitos engenheiros de software consideram qualidade do processo tão importante quanto a qualidade do produto.
- Se há um processo de desenvolvimento então pode-se perguntar:
  - Onde e quando é mais provável encontrarmos um tipo particular de defeito?
  - Como encontrar defeitos mais cedo?
  - Existem atividades alternativas para tornar o processo mais efetivo ou eficiente em relação à qualidade?

  Nos anos 90 surgiram os modelos de maturidade de processo de software: CMM (*Capability Maturity Model*), ISO 9000 e SPICE (*Software Process Improvement and Capability dEtermination*).

Quem faz Engenharia de Software

Ver Figura 4

Quem faz Engenharia de Software

Componente chave do desenvolvimento de software é a comunicação entre clientes, usuários e desenvolvedores; se ela falhar o sistema irá fracassar. Mas qual é o papel de cada um?

- Cliente: quem paga o sistema. Exemplo: o *Chief Information Officer*, também conhecido como o diretor de TI, da *Transportes Rápidão*.
- Usuário: quem vai usar o sistema. Exemplo: gerente e funcionários do departamento de logística.
- Desenvolvedor: quem faz o software. Exemplo: desenvolvedores da empresa *SoftLeste*.

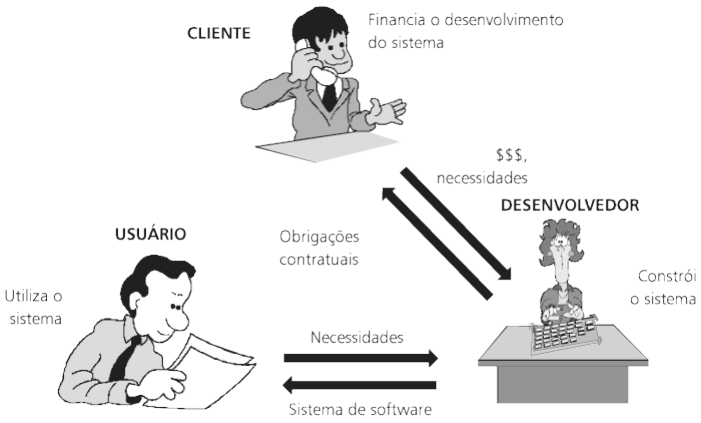

Figura 4: Participantes no desenvolvimento de software.

Quem faz Engenharia de Software

Dependendo do projeto, a distinção de papéis acima pode variar.

Muitas vezes o cliente e o desenvolvedor fazem parte de uma mesma organização. Por exemplo, o departamento de informática pode desenvolver o sistema para a empresa.

Uma outra situação é quando o cliente, o usuário e o desenvolvedor são um grupo de pessoas. Por exemplo, isto é o que ocorre em um projeto de software livre.

Uma abordagem de sistemas

Um sistema é uma união de:

- hardware;
- software;
- usuário;
- conexões com outros hardwares;
- conexões com outros softwares.

Importante: estabelecer as fronteiras do sistema.

Uma abordagem de sistemas

Exemplo:

> Suponha que seu superivsor peça que você escreva um programa para imprimir os contracheques dos funcionários do seu escritório.

Você deve saber se o programa simplesmente lê a quantidade de horas trabalhadas de um outro sistema e imprime o resultado, ou se ele deve calcular as informações do pagamento.

De modo semelhante, saber se o programa deve fazer cálculos relativos a impostos, previdência e benefícios, ou se um relatório desses intes será fornecido com cada contracheque.

O que está sendo perguntado é: onde começa e onde termina o sistema?

Uma abordagem de sistemas

Ver Figura 5

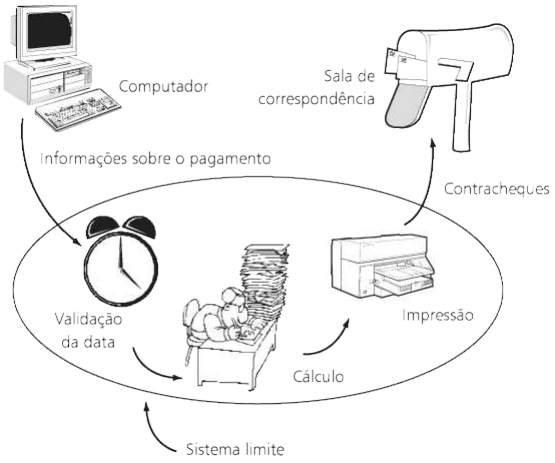

Figura 5: Sistema de produção de contracheques.

Uma abordagem de engenharia

Construindo uma casa × construindo um software

Casa	Software
Identificar e analisar os requisitos	Análise e definição dos requisitos
Produzir e documentar todo o projeto	Projeto do sistema
Detalhar as especificações	Projeto do programa
Identificar e projetar os componentes	Escrever os programas
Construir cada componente	Testes das unidades
Testar cada componente	Teste de integração
Integrar os componentes	Teste do sistema
Fazer as modificações finais	Entrega do sistema
Manutenção contínua	Manutenção

Equipe de desenvolvimento

Ver Figura 5

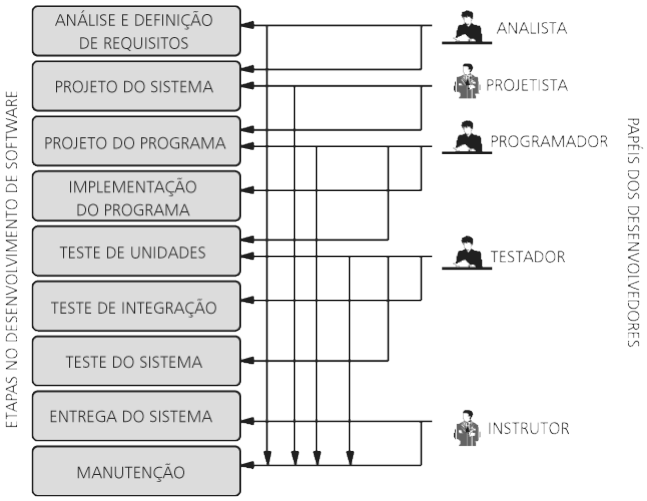

Figura 6: Papéis da equipe de deenvolvimento.

Membros de atividades de apoio: bibliotecários; equipe de gerência de configuração.

Resumo

- Importância do software.
- Engenharia de Software.
- Software: o que é e quando ele é bom.
- Qualidade de produto.
- Qualidade de processo.
- Quem faz engenharia de software.
- Abordagem de sistemas.
- Abordagem de engenharia.
- Equipe de desenvolvimento.

Exercícios

1. Em um diagrama de casos de uso o que define a fronteira ou escopo do sistema?
2. Por que a contagem de defeitos pode ser uma medida equivocada da qualidade de produto ?
3. Muitas organizações compram softwares comerciais pensando que será mais barato do que desenvolver e manter o software internamente. Descreva os prós e contras de se utilizar um software do tipo COTS. Por exemplo, o que acontece quando os fabricantes não dão mais suporte técnico a um produto? O que os clientes, usuários e desenvolvedores devem prever durante o projeto de um produto que utiliza software COTS em um sistema maior?

Exercícios retirados de [1].

*

Referências

[1] Shari Lawrence Pfleeger. *Engenharia de Software: Teoria e prática*. Prentice-Hall, São Paulo, 2a. edition, 2004.

[2] Wikipedia. Verbete – Software Engineering. `http:///en.wikipedia.org`. Visitado em 10 de agosto, 2007.