Projeto spider
Tool Suite for Quality

Fundamentos de Engenharia de Software

Prof. Dr. Sandro Bezerra

srbo@ufpa.br

2024

Introdução

- Software
- the programs that run on a computer and perform certain functions (Merriam-Webster)
- Obs: a palavra ”softwares” não existe

Introdução

- Software
- Tecnologia mais importante no cenário mundial.
- Hoje, tem um duplo papel: é um produto e, ao mesmo tempo, o veículo para distribuir um produto.
- Distribui o produto mais importante de nossa era – a informação
- A enorme indústria de software tornou-se fator dominante nas economias do mundo industrializado.
- Equipes de especialistas em software, cada qual concentrando-se numa parte da tecnologia necessária para distribuir uma aplicação complexa, substituíram o programador solitário de antigamente.

Introdução

- Software não se desgasta…
- ... Mas se deteriora!
- Quando um componente de hardware se desgasta, ele é substituído por uma peça de reposição.
- Não existem peças de reposição de software! Cada defeito de software indica um erro no projeto ou no processo pelo qual o projeto foi traduzido em código de máquina executável.
- Portanto, as tarefas de manutenção de software, que envolvem solicitações de mudanças, implicam complexidade consideravelmente maior do que a de manutenção de hardware.

Introdução

- Campos de aplicação de software na atualidade
- Software de sistema
- Software de aplicação
- Software de engenharia/científico
- Software embarcado
- Software para linha de produtos
- Aplicações Web e Aplicativos Móveis
- Software de inteligência artificial

Introdução

- Software Legado
- *Sistemas de software legado... foram desenvolvidos décadas atrás e têm sido continuamente modificados para se adequar às mudanças dos requisitos de negócio e a plataformas computacionais. A proliferação de tais sistemas está causando dores de cabeça para grandes organizações que os consideram dispendiosos de manter e arriscados de evoluir*

Dayani-Fard et al

A evolução do software

- WebApps
- Aplicativos Móveis
- Computação em Nuvem
- Linhas de produto de software

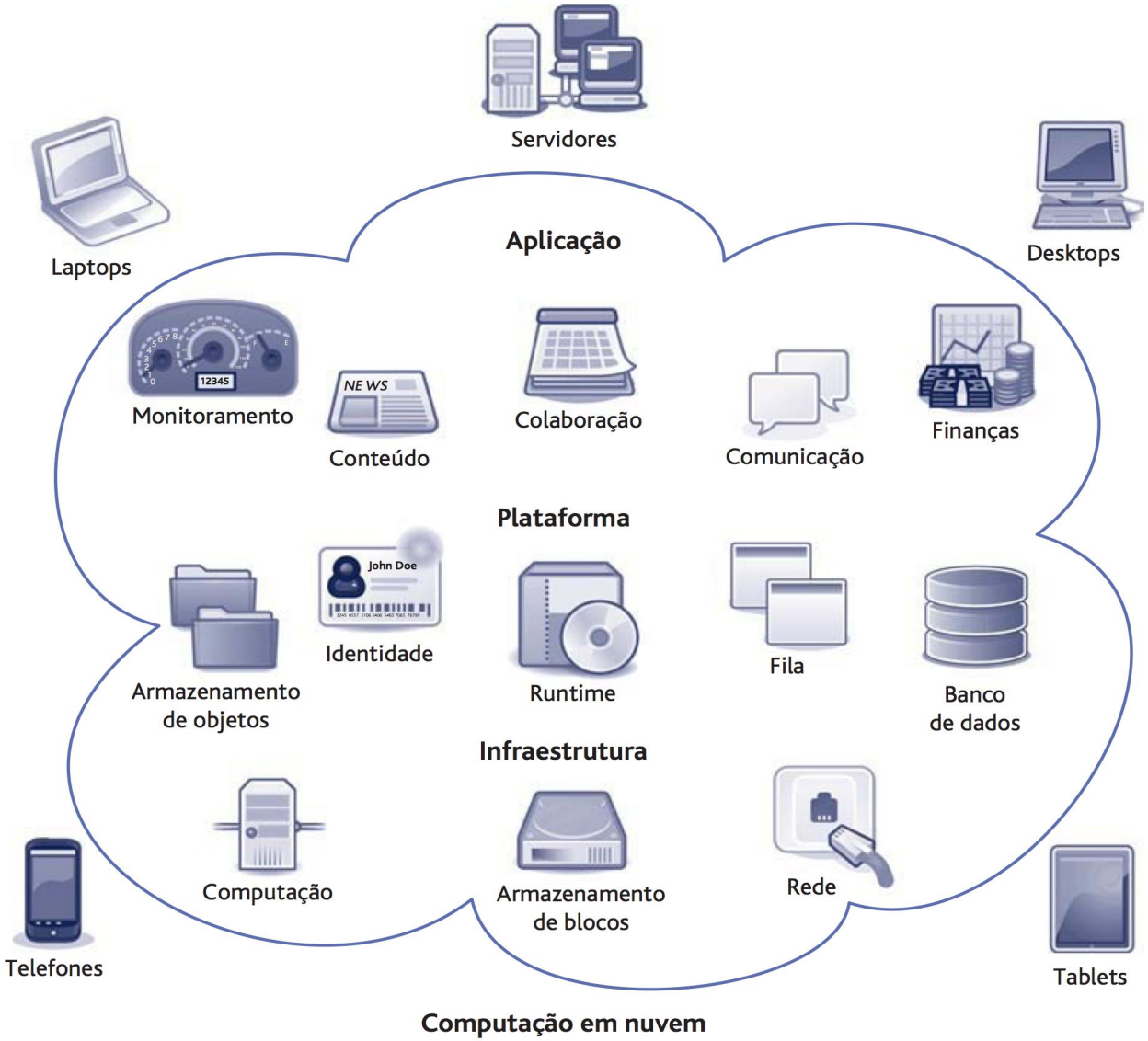


FIGURA 1.3 Arquitetura lógica da computação em nuvem [Wik13].

Introdução à Engenharia de Software

Para construir uma casa de cachorro…

- Uma pessoa consegue construir
- Precisa de pouco projeto
- O processo é simples
- As ferramentas são simples e fáceis de encontrar

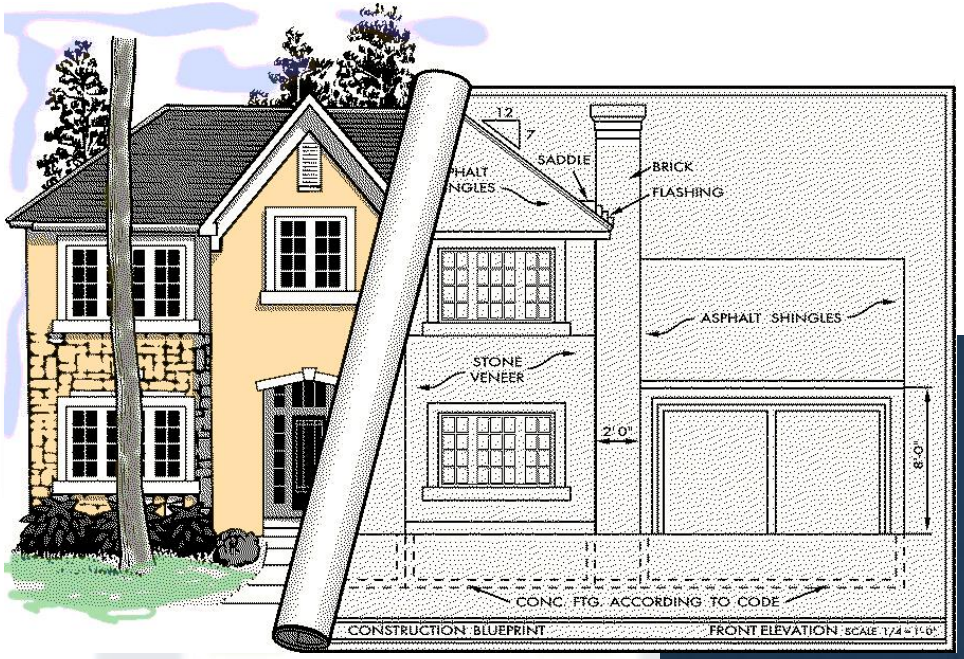
Fonte: Grady Booch. Software Architecture and the UML.

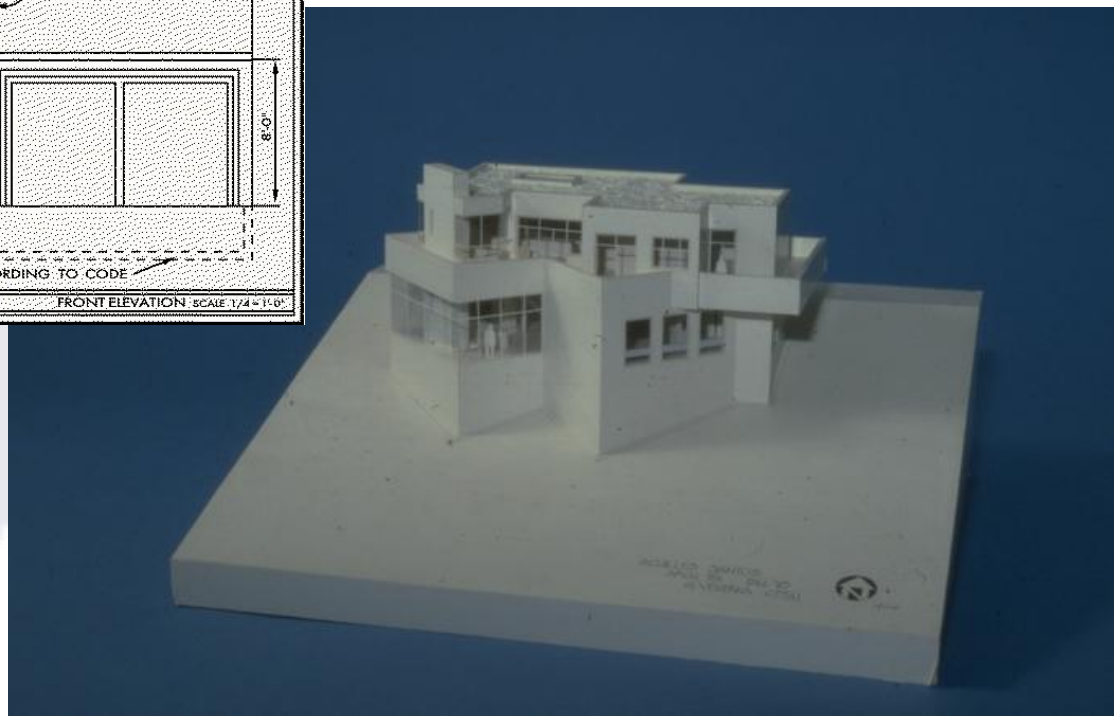
Para construir uma casa…

- É necessária uma equipe
- Requer projeto
- Requer um processo bem definido
- Requer ferramentas especializadas

Fonte: Grady Booch. Software Architecture and the UML.

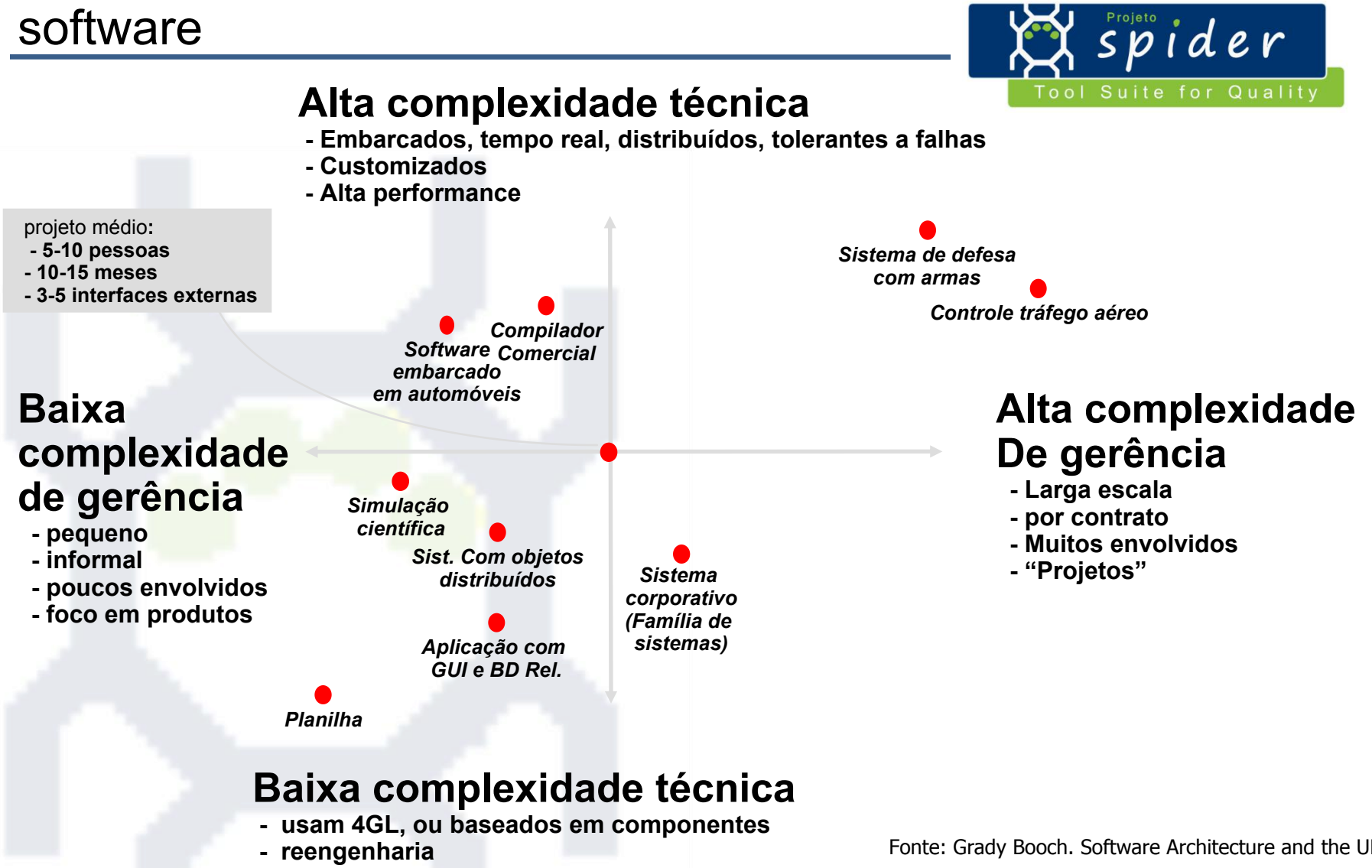
Projetando uma casa





Fonte: Grady Booch. Software Architecture and the UML.

Dimensões da complexidade de software



Fonte: Grady Booch. Software Architecture and the UML

Questões introdutórias

- Por que é fácil construir pequenos programas e há grande dificuldade quando é exigido algo mais complexo?
- Quanto custa corrigir problemas em software?

Custo de Correção

Custos gerados por problemas em requisitos

- Segundo Boehm e Papaccio (apud Pfleeger, 2004), o custo relativo para o conserto de um problema de requisitos em cada fase de desenvolvimento de sistema é:
- $1 na fase de análise de requisitos
- $5 na fase de projeto do sistema
- $10 na fase de codificação
- $20 na fase de teste de unidade
- $200 após a entrega do sistema

[Conte 2007]

Erros em Requisitos

- Quanto mais tarde um erro é detectado, maior o custo de corrigí-lo.
- Muitos erros são realizados durante a elicitação e definição de requisitos
- Muitos erros nos requisitos podem ser detectados cedo no ciclo de desenvolvimento.

- Erros típicos incluem fatos incorretos, omissões, inconsistências e ambiguidade.
- Erros nos requisitos constituem uma das maiores preocupações da indústria de software.

[Conte 2007]

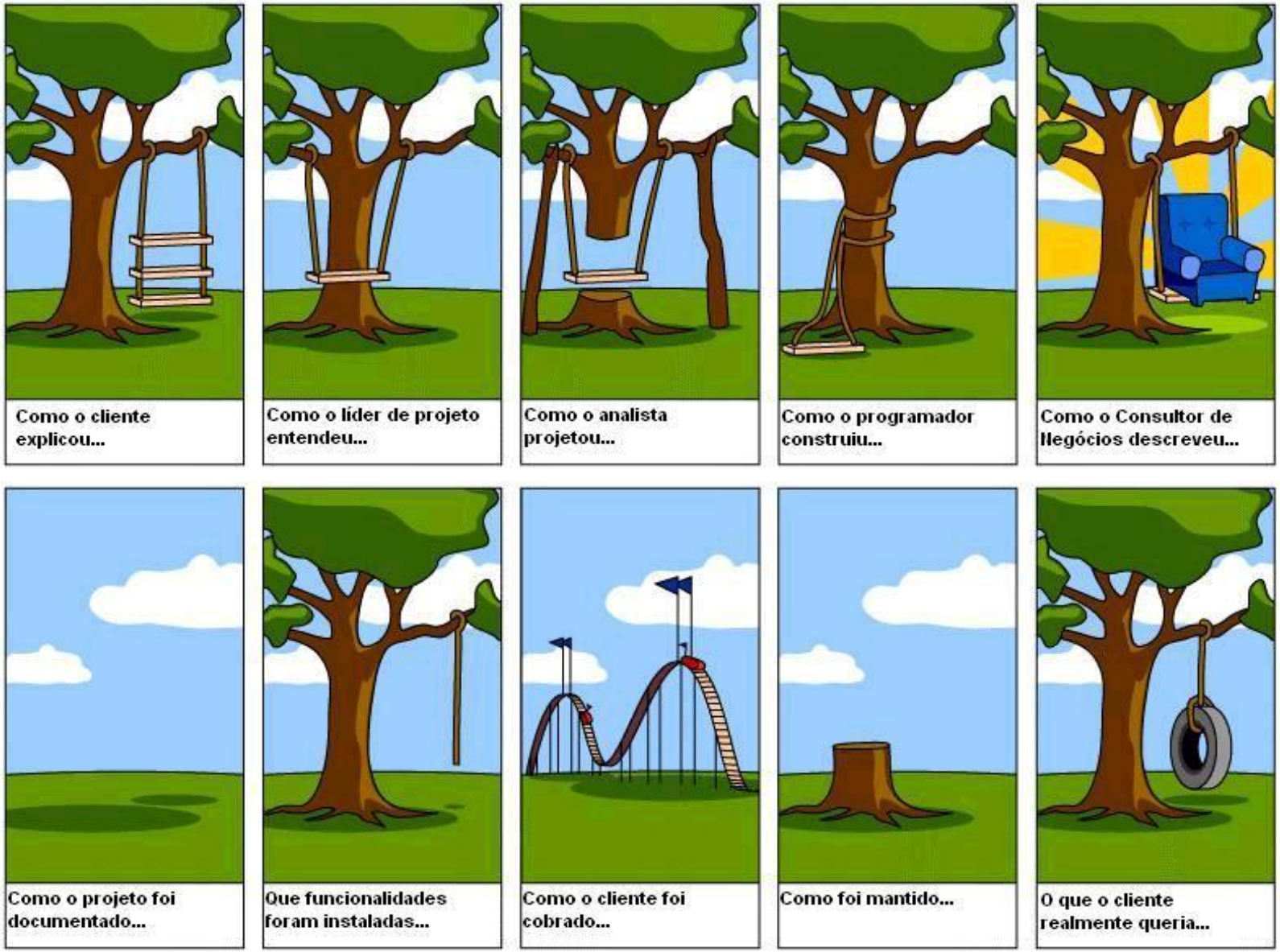
Erros em Requisitos

- Fred Brook's adiciona:

"a parte mais difícil da construção de um sistema de software é decidir, precisamente, o que deve ser construído... Nenhuma outra parte do trabalho aleja mais o sistema resultante se feita errada. Nenhuma outra parte é mais difícil de corrigir depois."

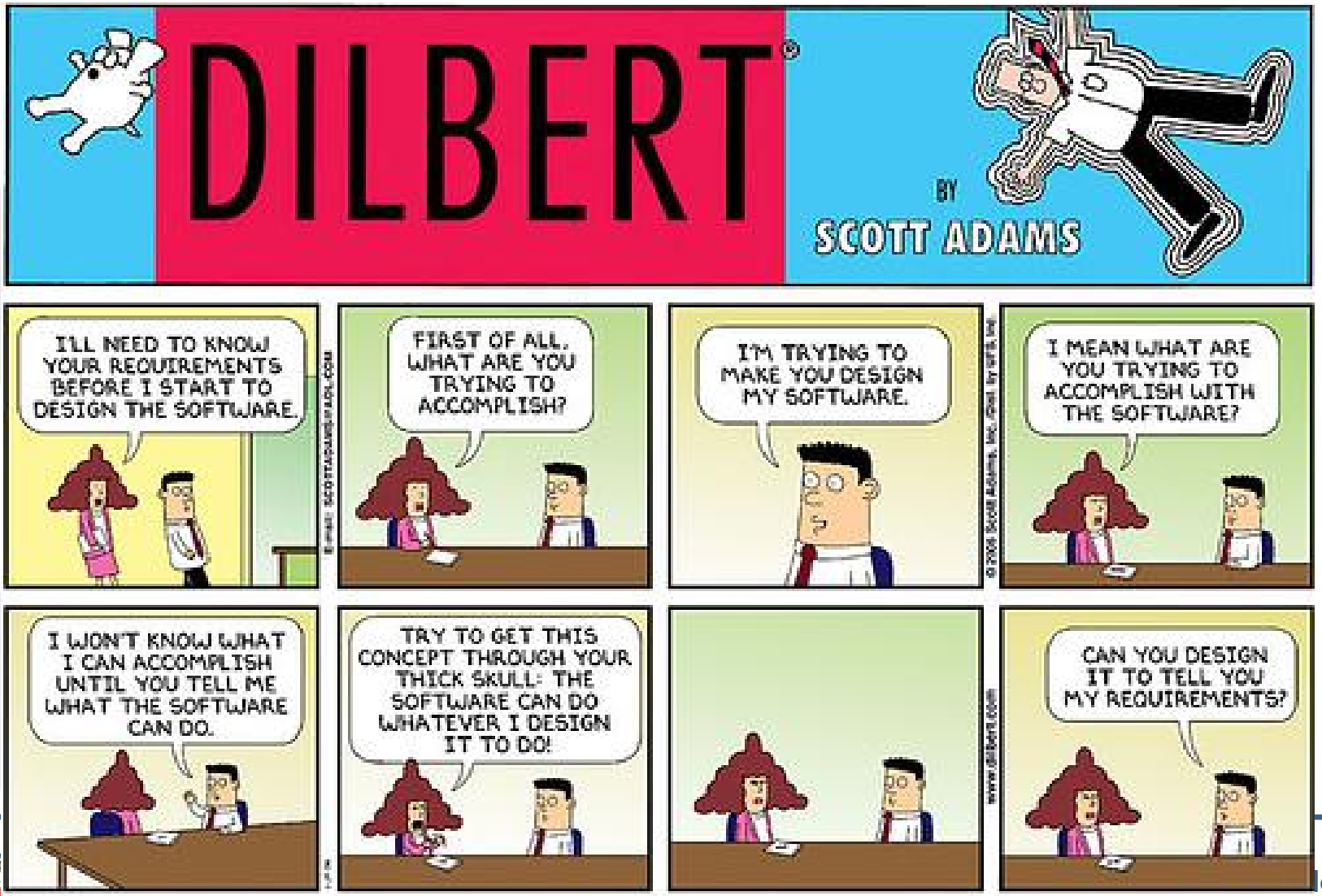
[Conte 2007]

Situação típica



Situação típica

Projeto spider
Tool Suite for Quality



Objetivos da Engenharia de Software

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

Vasconcelos, A. “Visão geral sobre a engenharia de software. UFPE.

O que é um software de qualidade?

- O software que satisfaz os requisitos solicitados pelo usuário. Deve ser fácil de manter, ter boa performance, ser confiável e fácil de usar
- Alguns atributos de qualidade
- Manutenibilidade
- O software deve evoluir para atender os requisitos que mudam
- Eficiência
- O software não deve desperdiçar os recursos do sistema
- Usabilidade
- O software deve ser fácil de usar pelos usuários para os quais ele foi projetado

Vasconcelos, A. "Visão geral sobre a engenharia de software. UFPE.

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

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

Vasconcelos, A. “Visão geral sobre a engenharia de software. UFPE.

Causas da Crise de Software

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

Vasconcelos, A. “Visão geral sobre a engenharia de software. UFPE.

Programação x Engenharia

- Programação
- Projeto pequeno
- Você
- Construir o que você mesmo quer
- Um produto
- Poucas modificações, feitas em seqüência
- Tempo de vida curto
- Barato
- Pouco impacto

- Engenharia
- Projeto Grande
- Times
- Construir o que os clientes querem
- Família de produtos
- Várias modificações, feitas em paralelo
- Tempo de vida longo
- Caro
- Grande Impacto

Software Atual

- Windows 10: aprox 80 milhões LOC
- Fonte: https://www.quora.com/How-many-lines-of-code-does-Windows-10-contain
- Linux 3.10 kernel: 19,5 milhões de LOC
- Fonte: Wikipedia
- Twitter: 3.000 imagens por segundo
- Fonte: http://highscalability.com/blog/category/example
- Netflix: cada filme é gravado em mais de 120 versões com diferentes resoluções
- Fonte: http://highscalability.com/blog/2015/11/9/a-360-degree-view-of-the-entire-netflix-stack.html
- WhatsApp: 700 milhões usuários, 30 bi de msgs/dia
- Fonte: https://m.facebook.com/story.php?story_fbid=10152994719980011&id=500035010

Software Atual

- Como organizar as atividades tal que 5000 pessoas possam trabalhar juntas ao mesmo tempo?
- Processo de software;
- Ferramentas de Gerência de Configuração;
- Como testar tanto código? Para tantas plataformas diferentes?
- Como projetar um sistema com 80 milhões de linhas de código? Como garantir a integridade deste projeto?

Engenharia de Software (ES) - Definições

- Uma disciplina cujo objetivo é a produção de software sem falhas, entregue dentro de prazo e orçamento e satisfazendo as necessidades dos usuários. Além disso, o software precisa ser facilmente modificável quando as necessidades do usuário mudam.[Schach]

- **O estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente software que seja confiável e que funcione eficientemente em máquinas reais. [Fritz Bauer-1969]**

Introdução

- Por que estudar ES? (1)
- Importância do software
- Utilização de computadores nos mais diversos ramos do conhecimento humano
- Da educação elementar à Engenharia Genética
- Indústria recente
- Evolução do hardware:
- 1960-1980:
- Alto custo limitava a evolução do software
- 1980-...
- Alto poder de processamento aliado a grandes quantidades de memória
- Redes de computadores e processamento paralelo
- Computadores com formatos não-convencionais
- Baixo custo

Introdução

- Por que estudar ES? (2)
- Indústria recente (continuação)
- Evolução no uso do software:
- 1950-1965
- Processamento *batch*, software para uso próprio, ...
- 1960-1975
- Multiusuário, processamento em tempo-real, produtos de software, banco de dados, programação estruturada;
- 1972-1988
- Sistemas distribuídos, impacto no consumidor, microprocessadores e PCs.
- 1985-...
- Sistemas “inteligentes”, arquiteturas paralelas, Internet, interfaces gráficas, software embutido, app mobile, ...