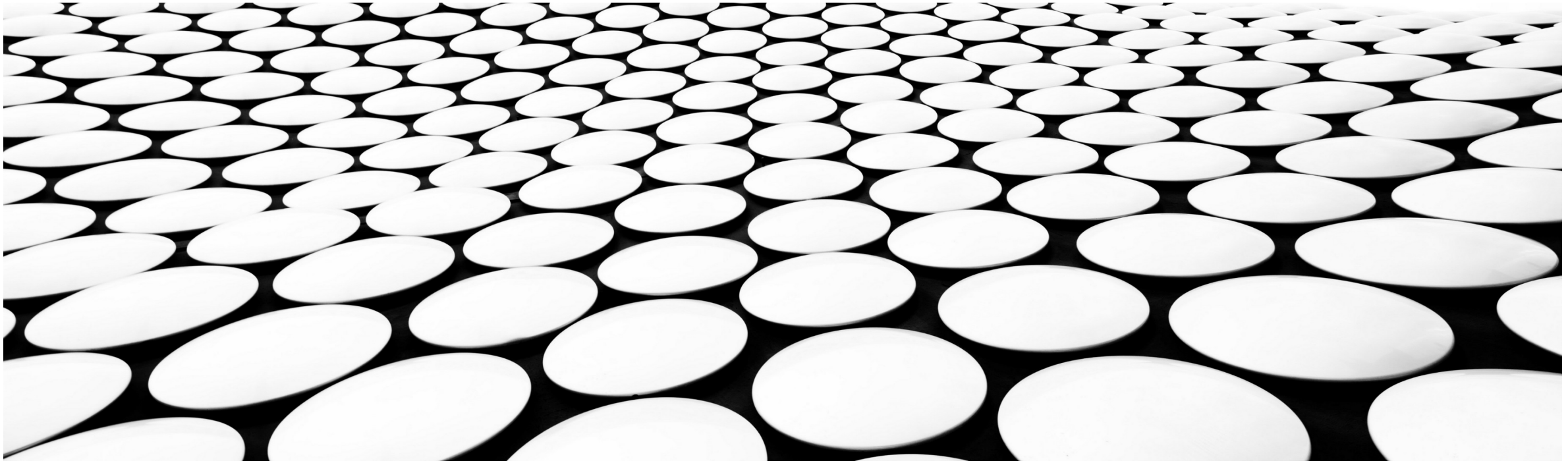
# QUALIDADE DE SOFTWARE

CAROLINA MARIA FRANCISCO COTA



# REFERÊNCIAS


- PRESSMAN, Roger S., MAXIM, Bruce R.. Engenharia de Software: uma abordagem profissional. 9ed. Porto Alegre: AMGH 2021.

# QUALIDADE DE SOFTWARE

- Em 2005, ComputerWorld [Hil05] lamentou que:
  ”O software ruim prejudica quase todas as organizações que usam computadores, causando perda de horas de trabalho durante o tempo de inatividade do computador, dados perdidos ou corrompidos, oportunidades de vendas perdidas, altos custos de suporte e manutenção de TI e baixa satisfação do cliente.”
- Um ano depois, InfoWorld [Fos06] escreveu sobre o
  ”O lamentável estado da qualidade do software” relatando que o problema de qualidade não havia melhorado.

# QUALIDADE DE SOFTWARE

- Hoje (2021), a qualidade do software continua sendo um problema, mas de quem é a culpa?
  - Os clientes culpam os desenvolvedores, argumentando que práticas descuidadas levam a software de baixa qualidade.
  - Os desenvolvedores culpam os clientes (e outros envolvidos), argumentando que datas de entrega absurdas e um fluxo contínuo de mudanças os obrigam a entregar o software antes de ele estar completamente validado.
- Quem está com a razão?

# O QUE É QUALIDADE? (VISÃO FILOSÓFICA)

Qualidade... sabemos o que ela é, embora não saibamos o que ela é. Mas essa afirmação é contraditória. Mas algumas coisas são melhores do que outras; ou seja, elas têm mais qualidade. Mas quando tentamos dizer o que é qualidade, separando-a das coisas que a têm, tudo desaparece como num passe de mágica! Não há nada a dizer a respeito. **Obviamente, algumas coisas são melhores do que outras... mas o que quer dizer "melhor"?**... E por aí vai (andando em círculos), girando rodas mentais e em nenhum lugar encontrando um ponto de tração. Mas o que é mesmo qualidade? O que é?

# O QUE É QUALIDADE? (VISÃO PRAGMÁTICA)

- A **visão transcendental** sustenta que qualidade é algo que se reconhece imediatamente, mas não se consegue definir explicitamente.
- A **visão do usuário** enxerga a qualidade em termos das metas específicas de um usuário. Se um produto atende a essas metas, ele apresenta qualidade.
- A **visão do fabricante** define qualidade em termos da especificação original do produto. Se o produto atende às especificações, ele apresenta qualidade.
- A **visão do produto** sugere que a qualidade pode ser ligada às características inerentes (p. ex., funções e recursos) de um produto.
- A **visão baseada em valor** mede a qualidade tomando como base o quanto um cliente estaria disposto a pagar por um produto.

# QUALIDADE DE SOFTWARE

- **Qualidade de projeto**: engloba o grau de atendimento às funções e características especificadas no modelo de requisitos.
- **Qualidade de conformidade**: focaliza o grau em que a implementação segue o projeto e que o sistema resultante atende às suas necessidades e às metas de desempenho.

# QUALIDADE DE SOFTWARE

- **Satisfação do usuário** = produto compatível + boa qualidade + entrega dentro do orçamento e do prazo previsto

- Robert Glass [Gla98] argumenta que a qualidade é importante, mas se o usuário não estiver satisfeito, nada mais importa.
- DeMarco [DeM98] reforça esse ponto de vista ao afirmar: “A qualidade de um produto é função do quanto ele transforma o mundo para melhor”.

# QUALIDADE DE SOFTWARE [PRESSMAN, 2021]

- ***“Uma gestão de qualidade efetiva aplicada de modo a criar um produto útil que forneça valor mensurável para aqueles que o produzem e para aqueles que o utilizam.”***

# GESTÃO DE QUALIDADE EFETIVA

- Estabelece a infraestrutura que dá suporte a qualquer tentativa de construir um produto de software de alta qualidade.
- Os aspectos administrativos do processo criam mecanismos de controle e equilíbrio de poderes que ajudam a evitar o caos no projeto – um fator-chave para a má qualidade.
- As práticas de engenharia de software permitem ao desenvolvedor analisar o problema e elaborar uma solução consistente – aspectos críticos na construção de software de alta qualidade.
- As atividades de apoio, como o gerenciamento de mudanças e as revisões técnicas, têm muito a ver com a qualidade, assim como qualquer outra parte da prática de engenharia de software

# PRODUTO UTIL

- Um **produto útil** fornece o conteúdo, as funções e os recursos que o usuário deseja.
- E não menos importante, deve fornecer confiabilidade e isenção de erros.
- Um produto útil sempre satisfaz às exigências definidas explicitamente pelos envolvidos.
- Além disso, ele satisfaz a um conjunto de requisitos implícitos (p. ex., facilidade de uso) que se espera de todo software de alta qualidade.

# AGREGAR VALOR

- Ao agregar valor tanto para o fabricante quanto para o usuário de um produto de software, um software de alta qualidade gera benefícios para a empresa de software e para a comunidade de usuários.
- A empresa fabricante do software ganha valor agregado pelo fato de um software de alta qualidade exigir menos manutenção, menos correções de erros e menos suporte ao cliente.
- A comunidade de usuários ganha um valor agregado, pois a aplicação fornece a capacidade de agilizar algum processo de negócio.

# FATORES DE QUALIDADE [MCC77]

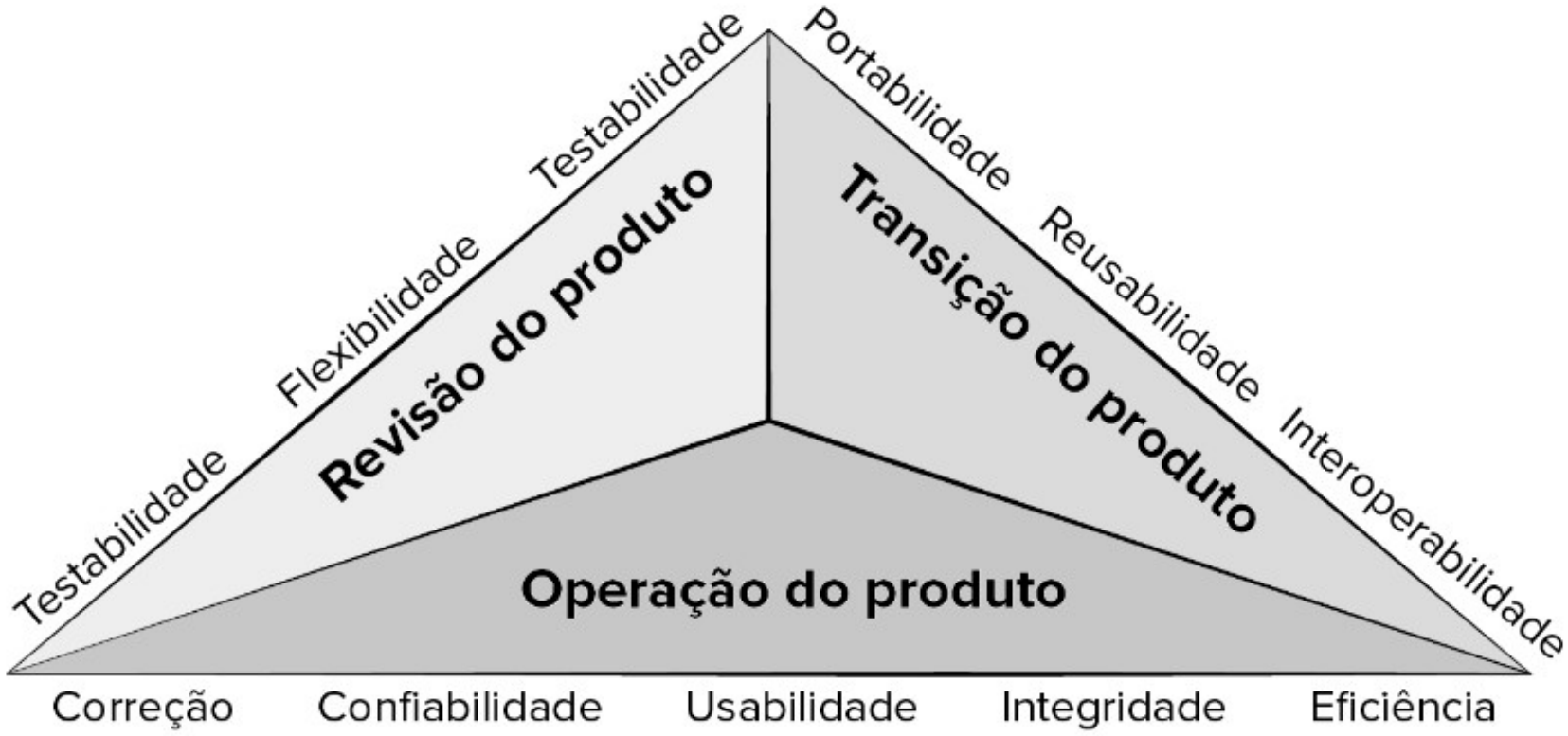



**Figura 15.1**
Fatores de qualidade de *software* de McCall.

# MODELO DE QUALIDADE ISO 25010 (2017)

- Modelo da **qualidade em uso**: características apropriadas quando consideramos utilizar o produto em um determinado contexto.
- Modelo da **qualidade de produto**: características que enfocam a natureza dinâmica e a estática dos sistemas de computador.

# MODELO DE QUALIDADE EM USO - ISO 25010

- **Eficácia**: Precisão e completude com as quais os usuários atingem suas metas.
- **Eficiência**: Recursos despendidos para atingir completamente os objetivos dos usuários com a precisão desejada.
- **Satisfação**: Utilidade, confiança, prazer, conforto.
- **Ausência de riscos**: Mitigação de riscos econômicos, de saúde, de segurança e ambientais.
- **Cobertura do contexto**: Completude, flexibilidade.

# MODELO DE QUALIDADE DE PRODUTO - ISO 25010

- **Adequação funcional**: Completo, correto, apropriado.
- **Eficiência de desempenho**: Tempestividade, utilização de recursos, capacidade.
- **Compatibilidade**: Coexistência, interoperabilidade.
- **Usabilidade**: Adequabilidade, facilidade de aprendizagem, operabilidade, proteção contra erros, estética, acessibilidade.
- **Confiabilidade**: Maturidade, disponibilidade, tolerância a falhas, facilidade de recuperação.
- **Segurança**: Confidencialidade, integridade, responsabilidade, autenticidade.
- **Facilidade de manutenção**: Modularidade, reusabilidade, modificabilidade, testabilidade.
- **Portabilidade**: Adaptabilidade, instalabilidade, facilidade de substituição.

# TESTES DE USABILIDADE

- Com que velocidade os usuários determinam se o produto de software pode ou não ser usado para ajudá-los a completar a sua tarefa? (adequabilidade)
- Quanto demora para os usuários aprenderem a usar as funções do sistema necessárias para completar a sua tarefa? (facilidade de aprendizagem)
- O usuário consegue lembrar como usar as funções do sistema em sessões de teste subsequentes sem precisar reaprendê-las? (facilidade de aprendizagem)
- Quanto demora para os usuários completarem suas tarefas usando o sistema? (operabilidade)

# TESTES DE USABILIDADE

- O sistema tenta impedir os usuários de cometer erros? (previsão de erros)
- O sistema permite que os usuários desfaçam operações que poderiam resultar em erros? (previsão de erros)
- As respostas oferecem reações favoráveis a perguntas sobre a aparência da interface do usuário? (estética)
- A interface do usuário se conforma aos itens da lista de verificação de acessibilidade exigidos para os usuários pretendidos? (acessibilidade)

# DIMENSÕES DE QUALIDADE
# DAVID GARVIN [GAR87] APUD [PRESMANN, 2016]

- **Qualidade de desempenho**: O software fornece todo o conteúdo, funções e recursos que são especificados como parte do modelo de requisitos de uma forma que agrega valor ao usuário final?
- **Qualidade do recurso**: o software oferece recursos que surpreendem e encantam os usuários finais iniciantes?
- **Confiabilidade**: o software oferece todos os recursos e capacidades sem falhas? Está disponível quando é necessário? Ele oferece funcionalidade livre de erros?

# DIMENSÕES DE QUALIDADE
# DAVID GARVIN [GAR87] APUD [PRESMANN, 2016]

- **Conformidade**: o software está em conformidade com os padrões de software locais e externos que são relevantes para o aplicativo? Está de acordo com as convenções de design e codificação de fato? Por exemplo, a interface do usuário está em conformidade com as regras de design aceitas para seleção de menu ou entrada de dados?
- **Durabilidade**: o software pode ser mantido (alterado) ou corrigido (depurado) sem a geração inadvertida de efeitos colaterais indesejados? As mudanças farão com que a taxa de erro ou a confiabilidade diminuam com o tempo?

# DIMENSÕES DE QUALIDADE
# DAVID GARVIN [GAR87] APUD [PRESMANN, 2016]

- **Facilidade de manutenção**: o software pode ser mantido (alterado) ou corrigido (depurado) em um período de tempo aceitavelmente curto. A equipe de suporte pode obter todas as informações de que precisam para fazer alterações ou corrigir defeitos?
- **Estética**: a maioria de nós concordaria que uma entidade estética tem uma certa elegância, um fluxo único e uma “presença” óbvia que são difíceis de quantificar, mas, mesmo assim, evidentes.
- **Percepção**: em algumas situações, você tem um conjunto de preconceitos que vão influenciar sua percepção de qualidade.

# O DILEMA DA QUALIDADE DO SOFTWARE

- Se produzimos um sistema de software de péssima qualidade, perdemos porque ninguém vai querer comprá-lo.
- Se, por outro lado, gastamos um tempo infinito, um esforço extremamente grande e grandes somas de dinheiro para construir um software absolutamente perfeito, então ele levará muito tempo para ser concluído, e o custo de produção será tão alto que iremos à falência.
- Construção de software “bom o suficiente”.

# CUSTO DA QUALIDADE

- Sabemos que a qualidade é importante, mas ela nos custa tempo e dinheiro.
- O custo da qualidade pode ser dividido em custos associados à:
  - Prevenção.
  - Avaliação.
  - Falhas

# CUSTOS DA PREVENÇÃO

- Custo de atividades de gerenciamento necessárias para planejar e coordenar todas as atividades de controle e garantia da qualidade.
- Custo de atividades técnicas adicionais para desenvolver modelos completos de requisitos e de projeto.
- Custos de planejamento de testes.
- Custo de todo o treinamento associado a essas atividades.

# CUSTOS DE AVALIAÇÃO

- Custos de avaliação incluem atividades para a compreensão aprofundada da condição do produto.
  - Custo da realização de revisões técnicas nos artefatos de engenharia de software.
  - Custo da coleta de dados e avaliação de métricas.
  - Custo dos testes e depuração.

# CUSTOS DE FALHA

- Custos de falhas internas ocorrem quando se detecta um erro em um produto antes de ele ser entregue.
  - Custo necessário para realizar reformulações (reparos) para corrigir um erro;
  - Custo que ocorre quando reformulações geram, inadvertidamente, efeitos colaterais que devem ser reduzidos.
  - Custos associados à reunião de métricas de qualidade que permitam a uma organização avaliar os modos de falha.

# CUSTOS DE FALHA

- Custos de falhas externas estão associados a defeitos encontrados após o produto ter sido entregue ao cliente.
  - Custo de solução de reclamações.
  - Custo de devolução e/ou substituição de produtos.
  - Custo com suporte por telefone/ e-mail.
  - Custos de mão de obra associados à garantia do produto.
  - Custos de má reputação e a consequente perda de negócios.

# CUSTOS DA CORREÇÃO DE ERROS E DEFEITOS

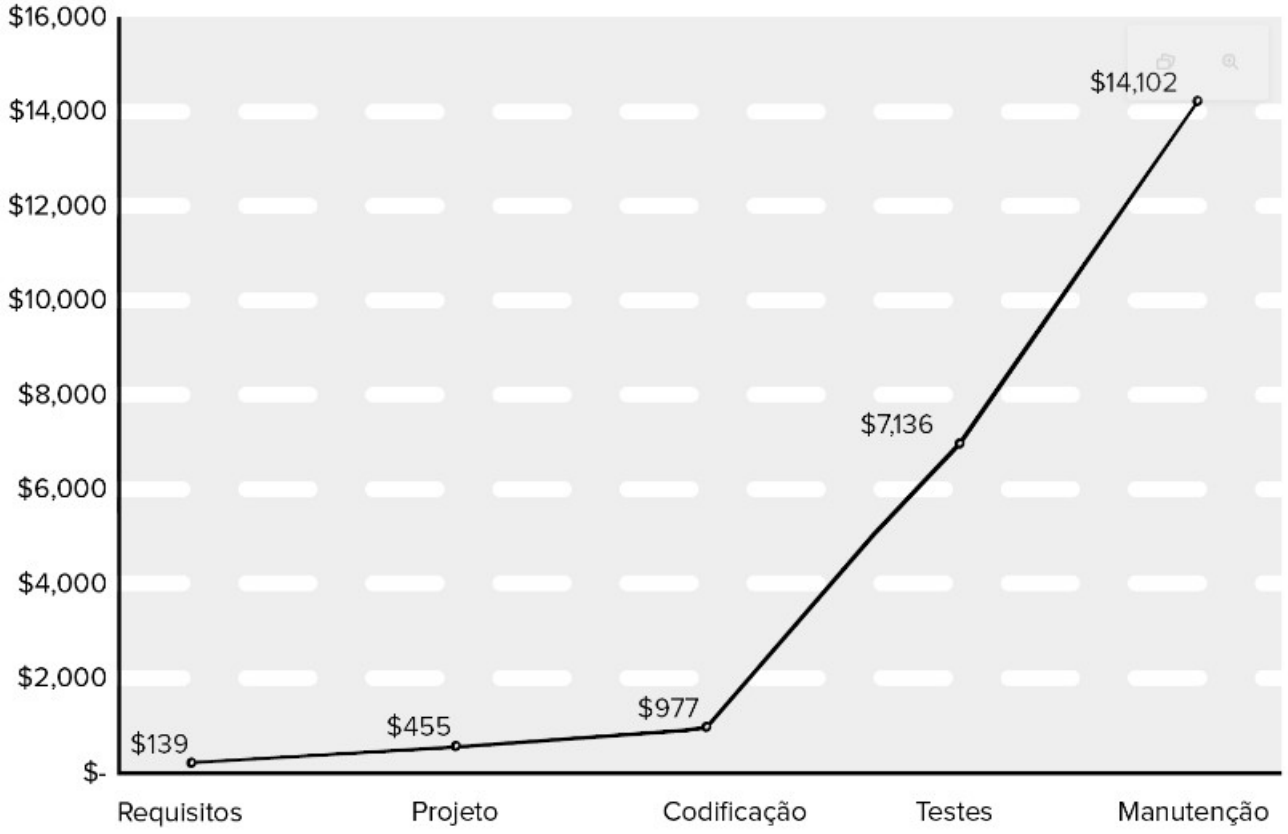



**Figura 15.2**
Custo relativo para correção de erros e defeitos.
*Fonte*: Boehm, Barry and Basili, Victor R., "Software Defect Reduction Top 10 List," IEEE Computer, vol. 34, no. 1, January 2001.

# QUALIDADE X RISCOS

- “As pessoas apostam seus empregos, seu conforto, sua segurança, seu entretenimento, suas decisões e suas próprias vidas em software. Tomara que estejam certas.”
- Exemplo:
  - Ao longo do mês de novembro de 2000, em um hospital no Panamá, 28 pacientes receberam doses maciças de raio gama durante tratamento para uma série de tipos de câncer. Nos meses seguintes, cinco desses pacientes morreram por contaminação radioativa, e outros 15 desenvolveram sérias complicações. O que provocou essa tragédia? Um pacote de software, desenvolvido por uma companhia norte-americana, foi modificado por técnicos do hospital para calcular doses alteradas de radiação para cada paciente.

# NEGLIGÊNCIA E RESPONSABILIDADE CIVIL

- A história é bastante comum. Um órgão do governo ou empresa contrata uma grande empresa de consultoria ou desenvolvedora de software para analisar os requisitos e então projetar e construir um “sistema” baseado em software para apoiar alguma atividade importante.
- O trabalho inicia com as melhores das intenções de ambas as partes, mas, na época em que o sistema é entregue, as coisas não andam bem.
- O sistema é lento, não fornece os recursos e funções desejados, é suscetível a erros e não tem a aprovação do usuário.
- Segue-se um litígio.

# SOFTWARE DE BAIXA QUALIDADE

- Software de baixa qualidade aumenta os riscos para desenvolvedores e usuários finais.
- Quando os sistemas são entregues com atraso, falham ao entregar a funcionalidade e não atendem às expectativas do cliente, o litígio segue...
- Software de baixa qualidade é mais fácil de hackear e pode aumentar os riscos de segurança para o aplicativo, uma vez implantado.

# SOFTWARE DE BAIXA QUALIDADE

- Um sistema seguro não pode ser construído sem focar na qualidade (segurança, confiabilidade) durante a fase de design.
- O software de baixa qualidade pode conter falhas arquitetônicas, bem como problemas de implementação (bugs).

# SOFTWARE DE QUALIDADE

- A qualidade do software é o resultado de uma boa gestão de projetos e práticas sólidas de engenharia.
- Para construir um software de alta qualidade, você deve entender o problema a ser resolvido e ser capaz de criar um projeto de qualidade que esteja em conformidade com os requisitos do problema.
- Eliminar falhas arquitetônicas durante o projeto pode melhorar a qualidade

# SOFTWARE DE QUALIDADE

- Gerenciamento do projeto - o plano do projeto inclui técnicas explícitas para a gestão da qualidade e da mudança.
- Controle de qualidade - série de inspeções, análises e testes usados para garantir a conformidade de um produto de trabalho com suas especificações.
- Garantia de qualidade - consiste nos procedimentos de auditoria e relatórios usados para fornecer à gestão os dados necessários para tomar decisões proativas.

# MODELO DE MATURIDADE
# CMMI (CAPABILITY MATURITY MODEL INTEGRATION)

- Desenvolvido pelo Software Engineering Institute (SEI) na Universidade Carnegie Mellon.
- Conjunto de melhores práticas que as organizações utilizam para melhorar seus processos e sistemas.
- Organizado em níveis de maturidade, cada um representando um nível diferente de maturidade e capacidade de processo.
- Organizações podem passar por avaliações para determinar seu nível de maturidade em diferentes áreas de processo.

# MODELO DE MATURIDADE
# CMMI (CAPABILITY MATURITY MODEL INTEGRATION)

- **Inicial (Nível 1)**: Os processos são imprevisíveis, mal controlados e reativos.
- **Gerenciado (Nível 2)**: São estabelecidos processos básicos de gerenciamento de projeto para rastrear custo, cronograma e funcionalidade.
- **Definido (Nível 3)**: Os processos são bem caracterizados e compreendidos e são adaptados a partir do conjunto de processos padrão da organização.
- **Gerenciado Quantitativamente (Nível 4)**: O desempenho do processo é medido e controlado quantitativamente.
- **Otimizado (Nível 5)**: A melhoria contínua do processo é possibilitada pelo feedback quantitativo do processo e pela implementação de ideias e tecnologias inovadoras.

# MODELO DE MATURIDADE MPS.BR (MELHORIA DE PROCESSO DO SOFTWARE BRASILEIRO)

- Iniciativa brasileira que visa promover a melhoria contínua dos processos de software nas organizações.
- Baseado em padrões internacionais e adaptado para a realidade brasileira, com o objetivo de elevar a qualidade e a maturidade dos processos de desenvolvimento de software no país.