Cruzada C - Inspeção

...

Verificação e Validação de Software

Grupo 1

Equipe

Gabriel Araújo

Jonathan Moraes

Jonathan Rufino

Laércio

Luis Filipe

Pedro Sales

Phelipe Wener

Rafael Rabetti

Tâmara Barbosa

Qualidade de *Software*

“Inspeções representam um tipo de revisões formais por pares, ou *peer reviews*, as quais são técnicas de análise para avaliação de forma, estrutura e conteúdo de um documento, código fonte ou outro produto de trabalho.”

(Wieger, 2002)

Objetivo

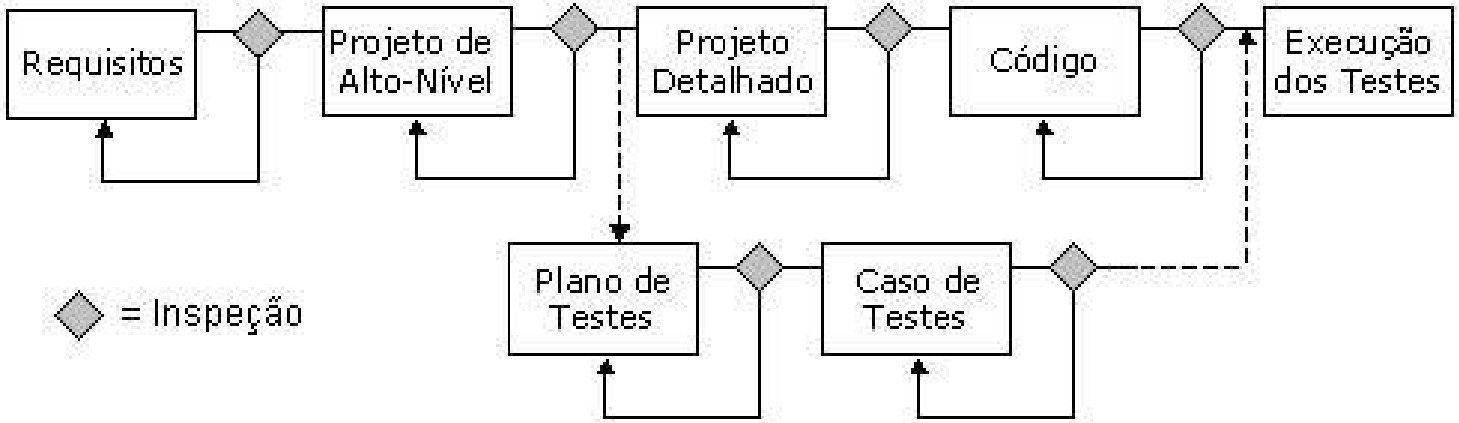
- Identificação e Remoção de Defeitos

Alvo

- Artefatos de *Software*, Código Fonte, Projeto

Saída

- Lista de Defeitos com Categorização Padronizada



Inspeções de Software nos Diferentes Artefatos.

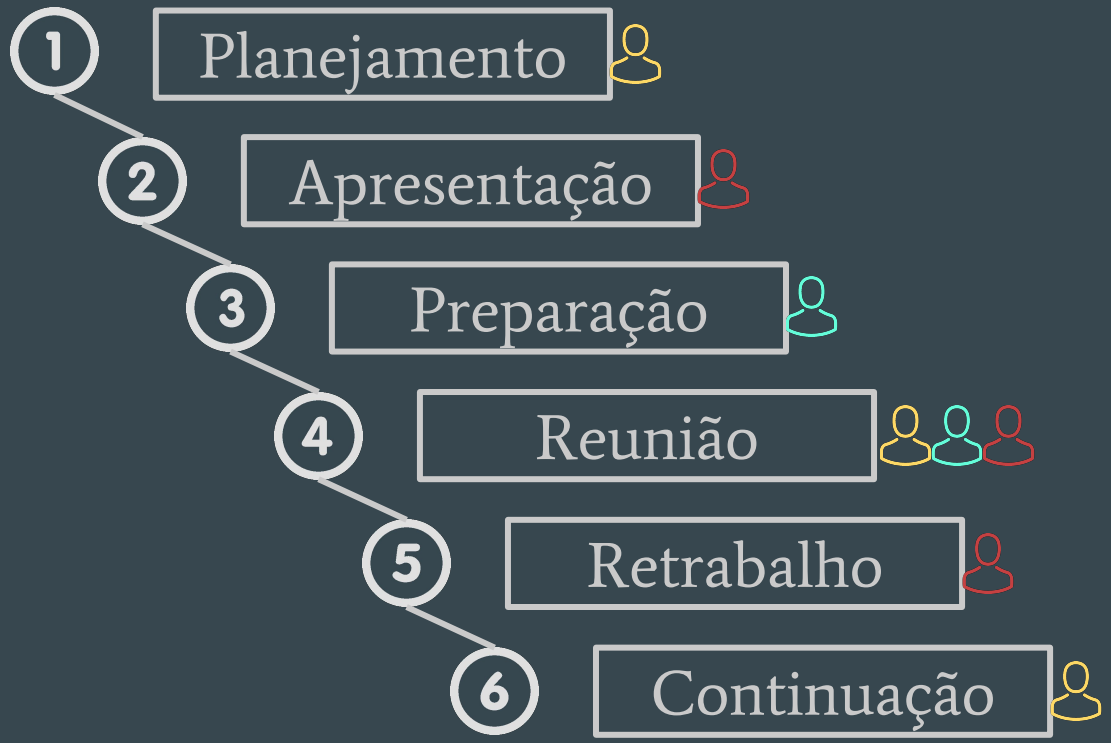
Adaptado de (ACKERMAN et al., 1989)

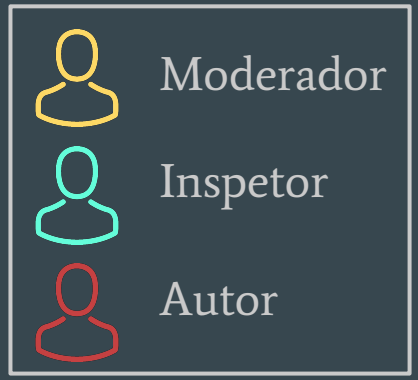
Classes de Defeitos

(IEEE 830, 1998)

- Omissão
- Ambiguidade
- Inconsistência
- Fato Incorreto
- Informação Estranha
- Outros

Processo de Inspeção (Fagan, 1976)



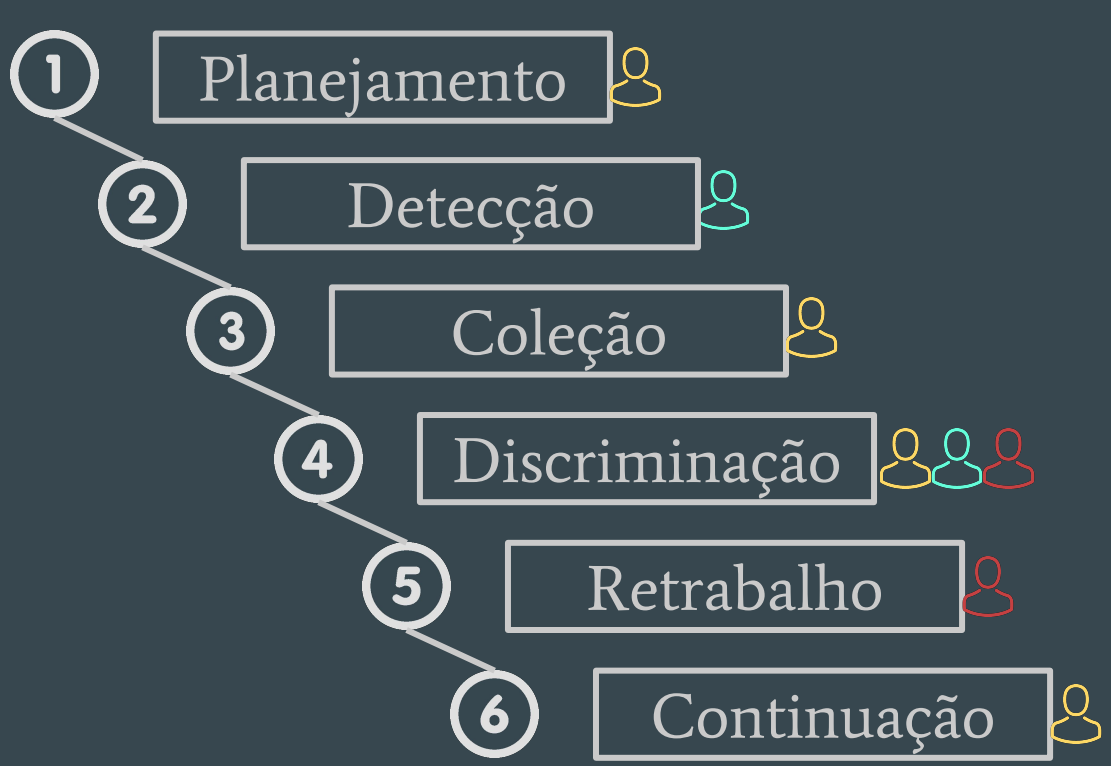


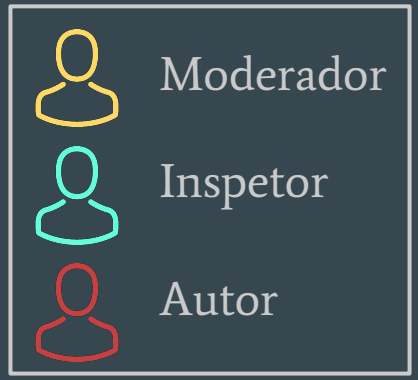
(Adaptado de FAGAN, 1976)

Reorganização do Processo

Baseados em diversos estudos experimentais sobre inspeções de software SAUER et al., (2000) propuseram uma reorganização do processo tradicional de inspeção. Essa reorganização visa a adequação do processo tradicional à inspeções com reuniões assíncronas e equipes geograficamente distribuídas. Assim, mudanças para reduzir o custo e o tempo total para a realização deste tipo de inspeção foram introduzidas.

Reorganização do Processo de Inspeção (SAUER et. al)





(Adaptado de SAUER et. al 2000)

Atores

- **Autor** é o próprio desenvolvedor do artefato que será inspecionado;
- **Moderador** é quem lidera a inspeção e as reuniões;
- **Redator** é quem relata os defeitos encontrados e as soluções sugeridas durante a inspeção;
- **Inspetor** é o membro da equipe que tenta encontrar erros no produto.

Aspectos Abordados

- Inspeção de documentos de requisitos
- Inspeção de código-fonte
- Inspeção de documentos de processos

Tipos de defeitos encontrados em cada aspecto

Documentos de requisitos e/ou processos

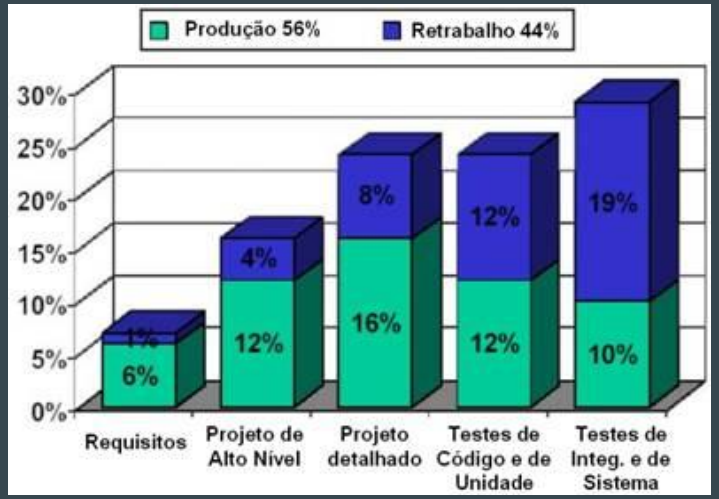
- Defeito de omissão
- Defeito de fato incorreto
- Defeito de inconsistência
- Defeito de ambigüidade
- Defeito de informação estranha

Código Fonte

- Defeitos de Omissão
- Defeitos de Comissão
- Defeito de inicialização
- Defeitos de computação
- Defeito de controle
- Defeito de interface
- Defeitos de dados
- Defeitos cosmético

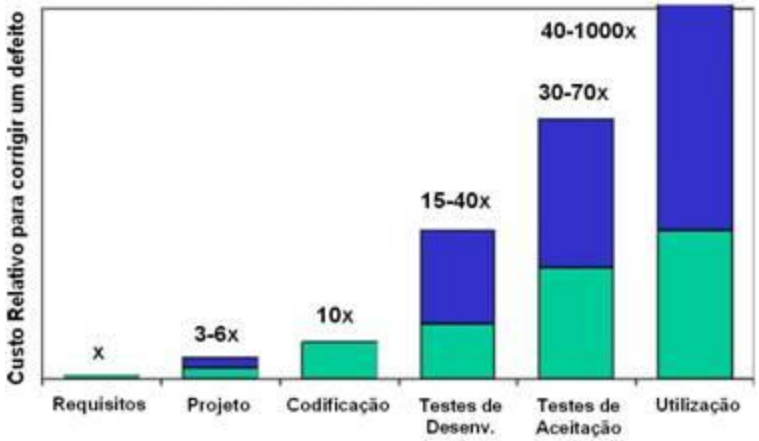
Benefícios da Aplicação de Inspeções de Software

O esforço gasto por organizações de software com retrabalho pode variar em média entre 40% e 50% do esforço total do desenvolvimento de um projeto. Uma estimativa da distribuição do retrabalho pelas atividades de desenvolvimento de software está ilustrada na figura abaixo:



Distribuição do retrabalho pelas atividades de desenvolvimento de software. Adaptado de (WHEELER et al., 1996)

Através da análise de 63 projetos, BOEHM (1981) apresenta o custo relativo da correção de defeitos encontrados em cada uma das atividades de desenvolvimento:



Custo relativo para corrigir um defeito.
Adaptado de (BOEHM, 1981).

Outros resultados significativos

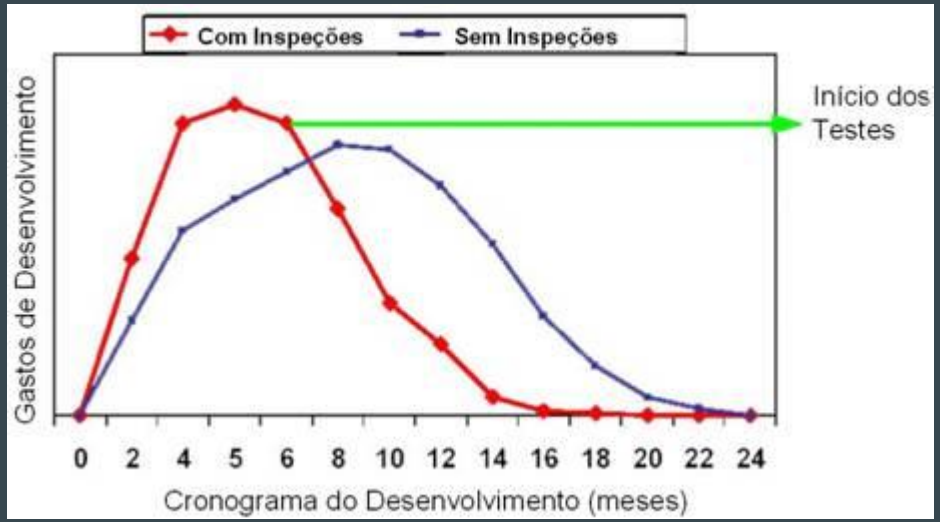
Esforço: o departamento de desenvolvimento da Ericsson em Oslo, Noruega, calculou uma redução bruta do esforço total de desenvolvimento em 20% aplicando inspeções. Além disso, resultados de estudos mostram que a introdução de inspeções de projeto pode reduzir o esforço com retrabalho em 44%.

Produtividade: de acordo com (GILB e GRAHAM, 1993), inspeções aumentam a produtividade de 30% a 50%;

Tempo: de acordo com (GILB e GRAHAM, 1993), inspeções reduzem o tempo de desenvolvimento de 10% a 30%;

Custo: resultados de estudos mostram que a introdução de inspeções de código pode reduzir os custos de implementação de projetos em 39%

Estimativa dos gastos de desenvolvimento utilizando e não utilizando inspeções



Outros Benefícios

- Aprendizado
- Integração entre processos de detecção e de prevenção de defeitos
- Produtos mais inteligíveis
- Dados defeituosos ajudam a melhorar o processo de software do projeto

Ferramentas

1. GRIP (GRoupware supported Inspection Process)
2. IBIS (Internet Based Inspection System)
3. ISPIS (Infra-estrutura de Suporte ao Processo de Inspeção de Software)

GRIP (GRoupware supported Inspection Process)

GRIP (Grünbacher et al., 2003). Nasceu da iniciativa de se adaptar um sistema COTS (Commercial Off-The-Shelf) de apoio a trabalho colaborativo (GSS) (Groupware Support System) para realizar inspeções em requisitos de software.

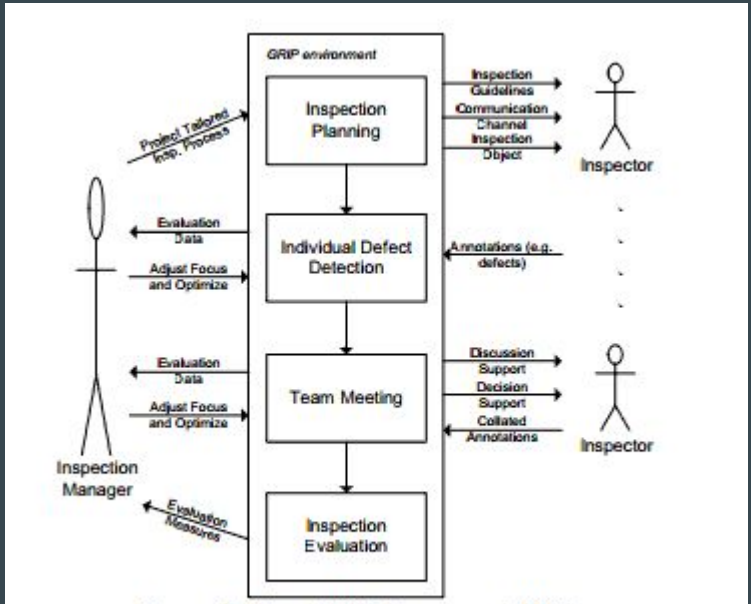


Figure 1: The GRIP framework [15].

Referências

- Kalinowski, Marcos. Introdução à Inspeção de Software. (2010) Engenharia de Software Magazine. 1ª ed.
- Grunbacher, et al. An Empirical Study on Groupware Support for Software Inspection Meetings. (2003)
- Melo, Silvana M. Inspeção de Software. (2010) Universidade de São Paulo (USP)
- Sommerville. Engenharia de Software. (2007) Addison Wesley, 8ª ed.
- Ferreira, Bruno. Uma Técnica para Validação de Processos de Desenvolvimento de Software. (2008)
- Kalinowski, Marcos. ISPIS: A Framework Supporting Software Inspection Proccesses. (2004)
- Vasconcelos, et al. Introdução à Engenharia de Software e à Qualidade de Software. (2006)
- Kalinowski, Marcos. Inspeções de Requisitos de Software em Desenvolvimento Incremental: Uma Experiência Prática. (2007) VI Simpósio Brasileiro de Qualidade de Software

Obrigado!