SCE306 – Engenharia de Software I

Garantia de Qualidade: Inspeção em DR

Profa. Ellen Francine Barbosa
francine@icmc.usp.br

Instituto de Ciências Matemáticas e de Computação — ICMC/USP

Roteiro

Garantia de Qualidade (SQA)
Revisões de Software
Inspeção de Software
Inspeção em DR
Exercício de Fixação

- Garantia de Qualidade (SQA)
- Revisões de Software
- Inspeção de Software
- Inspeção em DR
- Exercício de Fixação

Garantia de Qualidade (SQA – Software Quality Assurance) I

- Conjunto de atividades técnicas aplicadas durante todo o processo de desenvolvimento.
- Dentre as atividades de SQA estão as atividades de **verificação** e **validação** de software.
  - O objetivo é minimizar a ocorrência de erros e riscos associados.
    - Detectar a presença de erros nos produtos de software.

Garantia de Qualidade (SQA – Software Quality Assurance) II

- Verificação
  - Assegurar consistência, completitude e corretitude do produto em cada fase e entre fases consecutivas do ciclo de vida.

Estamos construindo corretamente o produto?

  - Assegurar que o produto, ou uma determinada função do mesmo, esteja sendo implementado corretamente.
    - Verifica-se inclusive se os métodos e processos de desenvolvimento foram adequadamente aplicados.

Garantia de Qualidade (SQA – Software Quality Assurance) III

- Validação
  - Assegurar que o produto sendo desenvolvido corresponde ao produto correto, conforme os requisitos do usuário.

Estamos construindo o produto certo?

Garantia de Qualidade (SQA – Software Quality Assurance) IV

- V&V abrangem um amplo conjunto de atividades de SQA:
  - Revisões técnicas formais
  - Auditoria de qualidade e configuração
  - Monitoramento de desempenho
  - Simulação
  - Estudo de viabilidade
  - Revisão da documentação
  - Revisão da base de dados
  - Testes
- V&V envolvem atividades de análise estática e de análise dinâmica.

Análise Estática

- Não requerem a execução propriamente dita do produto.
- Podem ser aplicadas em qualquer produto intermediário do processo de desenvolvimento.
  - Documento de requisitos, diagramas de projeto, código-fonte, planos de teste, ...
- As revisões são o exemplo mais clássico de análise estática.
  - **Inspeção**
  - Walkthrough
  - Peer Review

Análise Dinâmica

- Requerem a execução do produto.
  - Código ou quaisquer outras representações executáveis do sistema.
- Exemplos de atividades que constituem uma análise dinâmica do produto:
  - Teste de Software
  - Simulação

Revisões de Software I

- Meio efetivo para melhorar a qualidade de software.
  - Filtro para o processo de Engenharia de Software.
- Podem ser aplicadas em vários pontos durante o desenvolvimento do software.
- Maneira de usar a diversidade de um grupo de pessoas para:
  - Apontar melhorias necessárias ao produto.
  - Confirmar as partes de um produto em que uma melhoria não é desejada ou não é necessária.
  - Realizar um trabalho técnico de qualidade mais uniforme de forma a torná-lo mais administrável.

Objetivo das Revisões de Software I

- Encontrar erros durante o processo de desenvolvimento, de forma que eles não se transformem em defeitos depois da entrega do software.
- Descoberta precoce dos erros.
  - Melhoria da qualidade já nas primeiras fases do processo de desenvolvimento.
  - Aumento da produtividade e diminuição dos custos.
    - Erros são detectados quando sua correção é mais barata.

Objetivo das Revisões de Software II

- É uma oportunidade de treinamento.
  - Aprender por experiência.
  - Participantes aprendem as razões e padrões em descobrir erros.
  - Participantes aprendem bons padrões de desenvolvimento de software.
- Com o decorrer do tempo....
  - A revisão auxilia os participantes a desenvolver produtos mais fáceis de entender e de manter.

Tipos de Revisões

- Discussão informal de um problema técnico.
- Apresentação do projeto de software para uma audiência de clientes, administradores e pessoal técnico.
- Revisões Técnicas Formais (RTF), as quais incluem avaliações técnicas do software realizadas em pequenos grupos.
  - Inspeção
  - Walkthrough
  - Peer-Review

Reunião de Revisão Técnica I

- Independentemente do formato da RTF, toda reunião de revisão deve seguir as seguintes recomendações:
  - Envolver de 3 a 5 pessoas.
  - Deve haver uma preparação para a reunião.
    - A preparação não deve exigir mais de 2 horas de trabalho de cada pessoa.
  - A reunião deve durar menos de 2 horas.
  - Deve-se focalizar uma parte específica do software.
    - Maior probabilidade de descobrir erros.

Reunião de Revisão Técnica II

SCE306 – Engenharia de Software I

Garantia de Qualidade (SQA)

Revisões de Software

Objetivo das Revisões

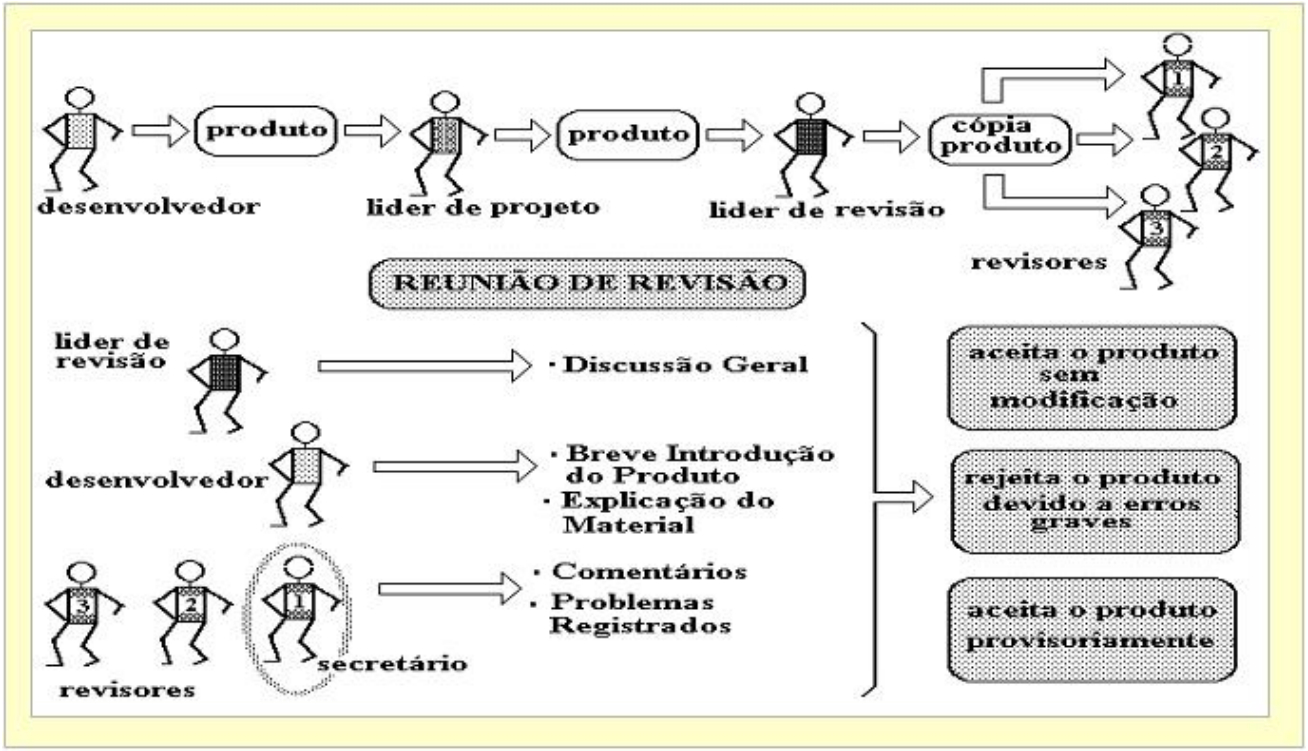
Tipos de Revisões

Reunião de Revisão Técnica

Inspeção de Software

Inspeção em DR

Exercício de Fixação



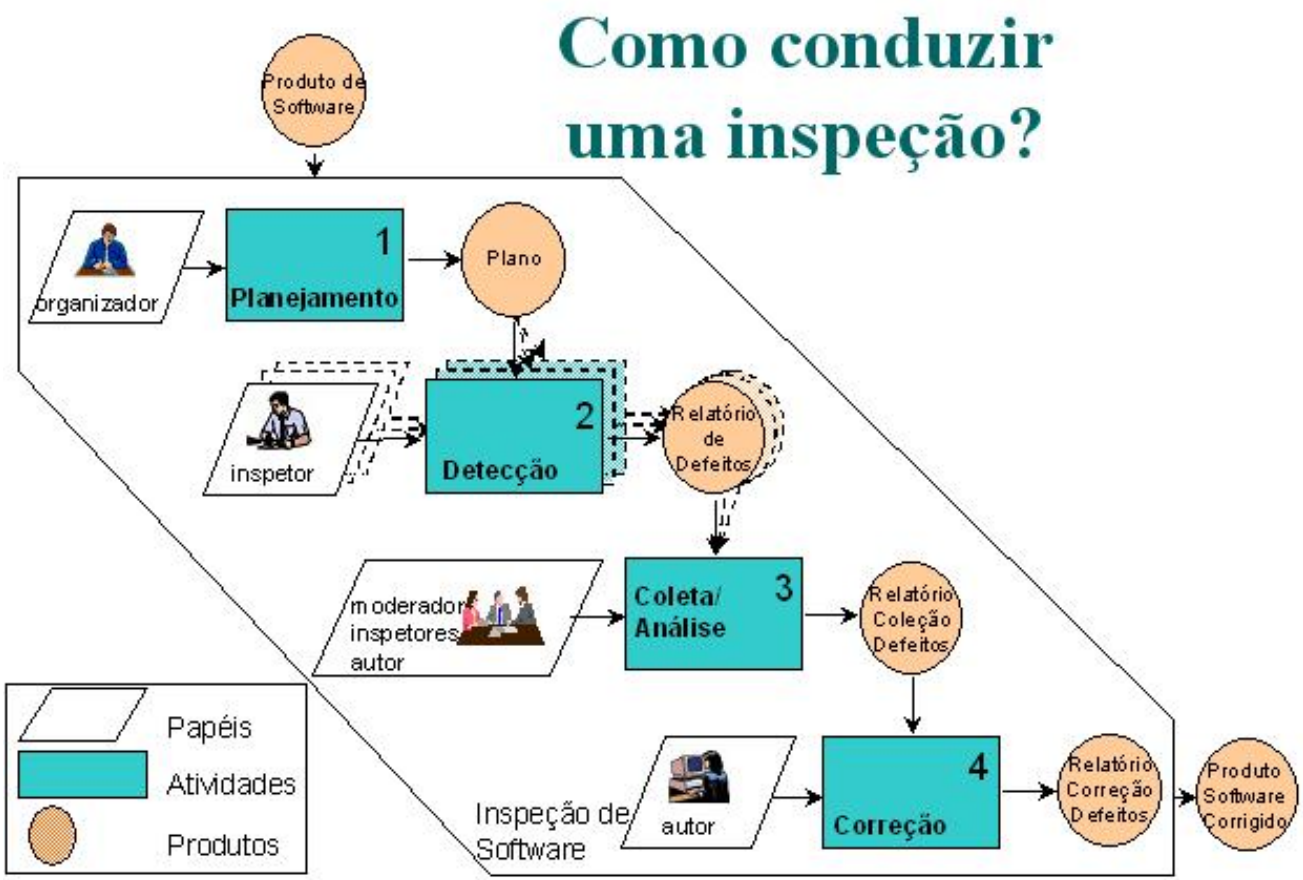
Diretrizes para Revisão

- Revise o produto, não o produtor.
- Fixe e mantenha uma agenda.
- Limite o debate e a refutação.
- Relacione as áreas problemáticas.
- Faça anotações por escrito.
- Limite o número de participantes e insista em uma preparação antecipada.
- Desenvolva uma lista de conferência (checklist) para cada produto que provavelmente será revisto.
- Atribua recursos e uma programação de tempo para as revisões.
- Realize um treinamento significativo para todos os revisores.
- Reveja suas antigas revisões.

Inspeção de Software

- Método de análise estática para verificar a qualidade de um produto de software.
- Pode-se inspecionar tanto produtos de software como também projetos de software.
  - Diferencial está na seleção dos aspectos que devem ser considerados durante a revisão.
- **Inspeção em Documentos de Requisitos**.

Etapas da Inspeção



Vantagens da Inspeção

- Detecção antecipada de defeitos (inspeção de requisitos).
- Aprende-se pela experiência.
  - Participantes aprendem os padrões e o raciocínio utilizado na detecção de defeitos.
  - Participantes aprendem bons padrões de desenvolvimento.
- A longo prazo...
  - A inspeção convence os participantes a desenvolverem produtos mais compreensíveis e mais fáceis de manter.

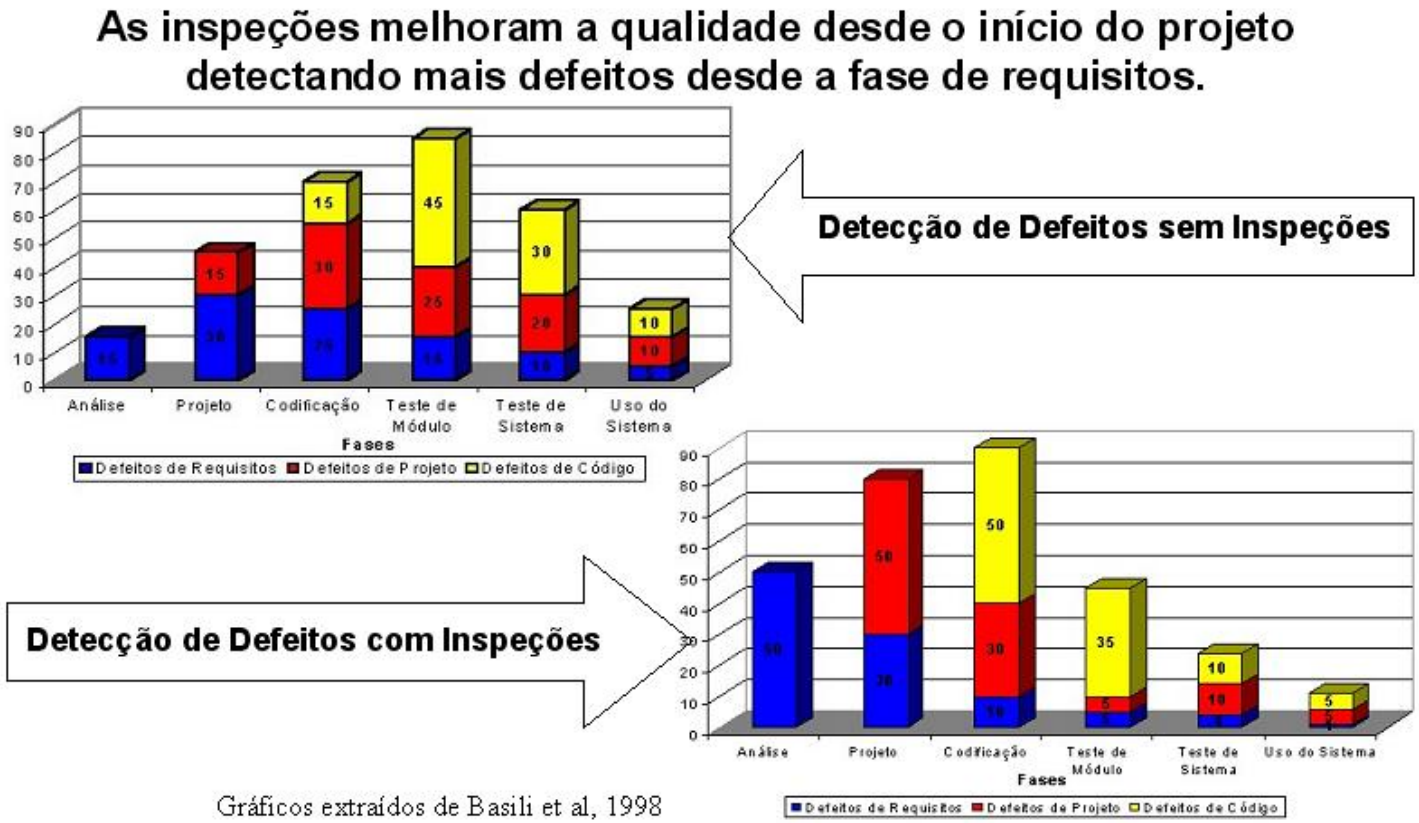
As inspeções ajudam a integrar o processo de **prevenção** de defeitos com o processo de **detecção** de defeitos.

Inspeção em Documento de Requisitos

- Benefícios da inspeção em Documento de Requisitos:
  - Detecção antecipada de defeitos.
  - Produtividade e diminuição do custo.

Inspeção em Documento de Requisitos

- Detecção antecipada de defeitos.



Gráficos extraídos de Basili et al, 1998

Inspeção em Documento de Requisitos

- Produtividade e diminuição do custo.

Inspeções melhoram produtividade por acharem defeitos quando eles são mais baratos para corrigir.

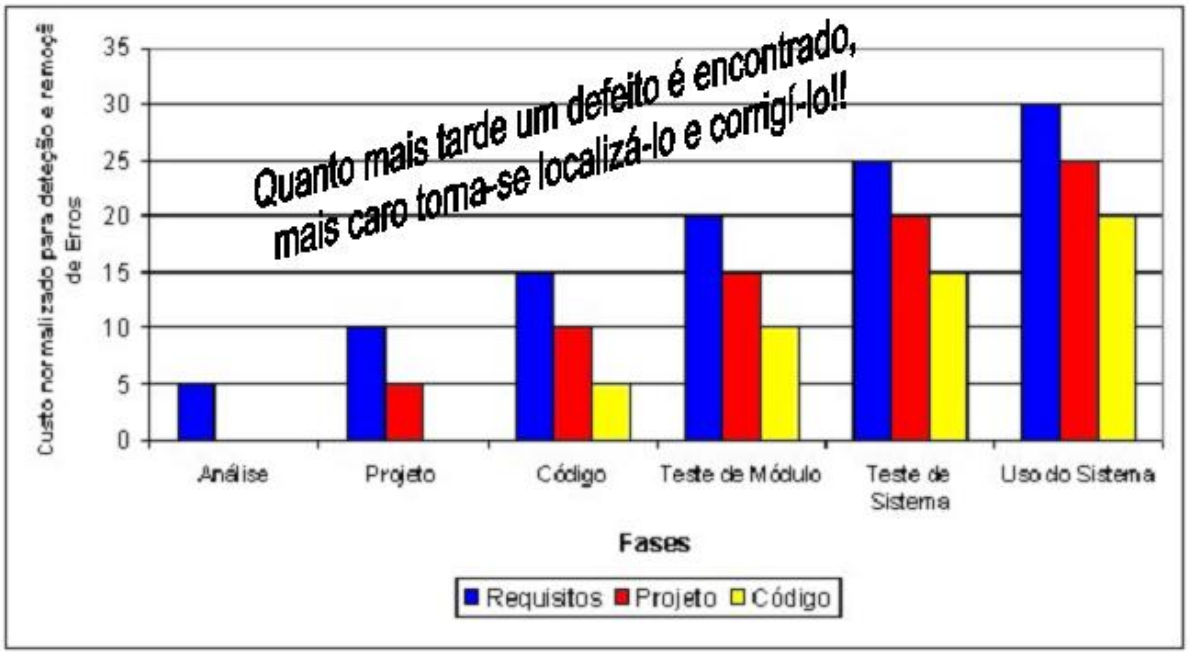


Gráfico extraído de Basili et al, 1998

Taxonomia de Defeitos em Documento de Requisitos I

SCE306 – Engenharia de Software I

Garantia de Qualidade (SQA)

Revisões de Software

Inspeção de Software

Inspeção em DR

Taxonomia de Defeitos em DR

Técnicas de Leitura

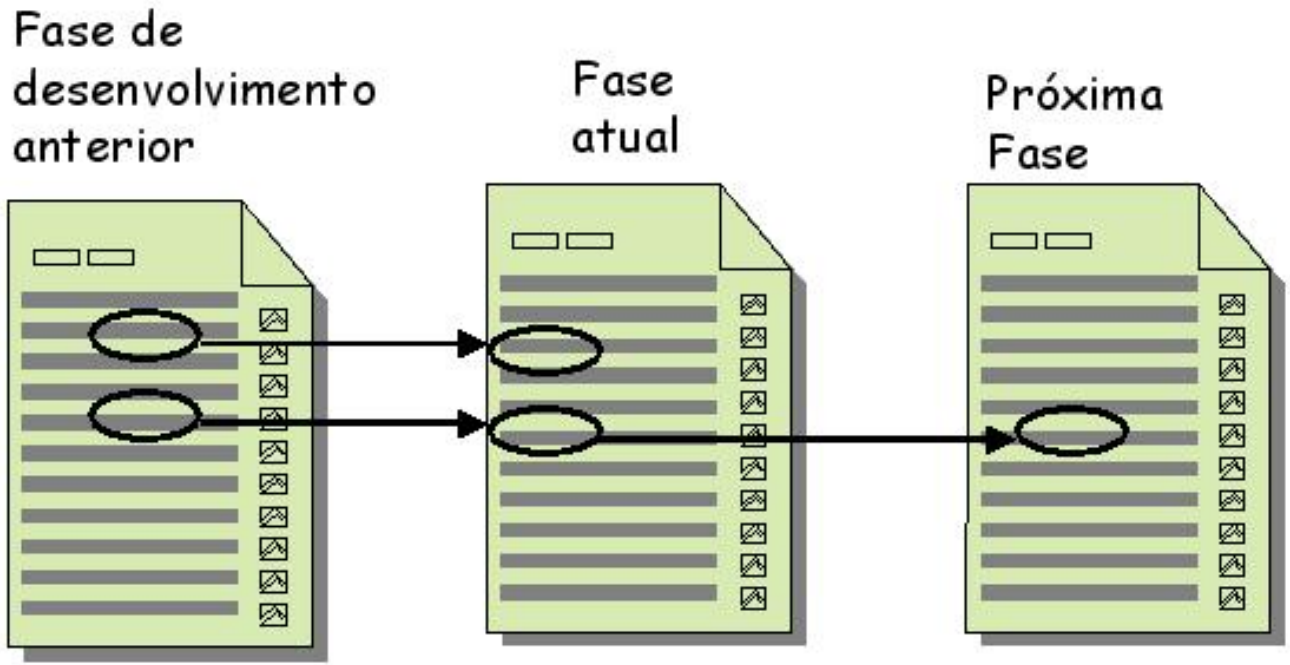
Ad-hoc

Checklist

Leitura Baseada em Perspectiva (PBR)

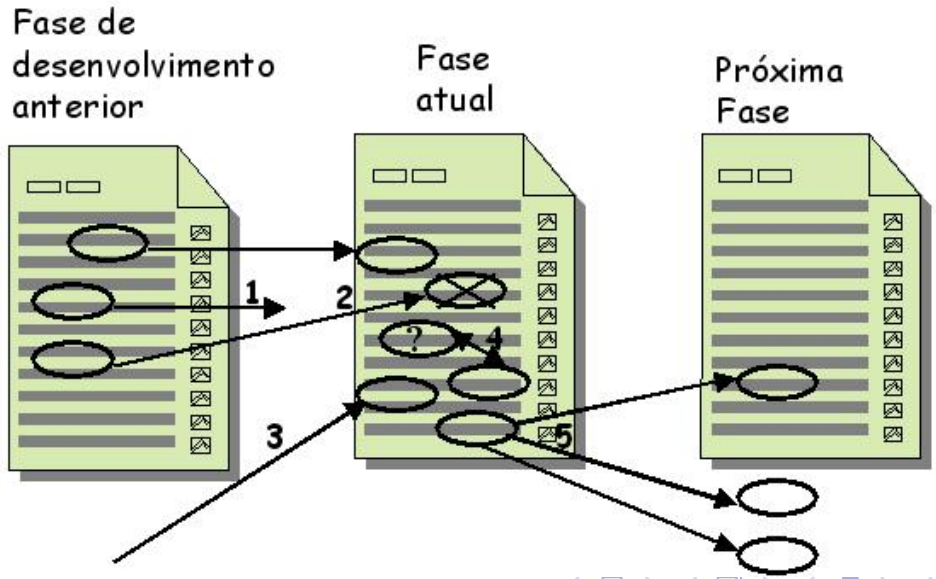
Exercício de Fixação

- Utilizada para classificação dos defeitos encontrados.
- Situação ideal:
  - A informação é transformada corretamente.



Taxonomia de Defeitos em Documento de Requisitos II

- Tipos de erros:
  1. A informação é perdida durante a transformação.
  2. A informação é transformada incorretamente.
  3. Informação estranha é introduzida.
  4. A mesma informação é transformada em diversas ocorrências inconsistentes.
  5. A mesma informação possibilita diversas transformações inconsistentes.



Taxonomia de Defeitos em Documento de Requisitos III

- Classes de Defeitos:
  - Defeitos de Omissão
  - Defeitos de Fato Incorreto
  - Defeitos de Inconsitência
  - Defeitos de Ambigüidade
  - Defeitos de Informação Estranha

Defeitos de Omissão I

- Defeito de Omissão (O): qualquer informação necessária que tenha sido omitida.
- Exemplo 1 (Omissão de Funcionalidade). Considere um sistema de biblioteca e os seguintes requisitos funcionais (RF):
  - RF2: O sistema deve solicitar a informação necessária para inserir um item bibliográfico: título, autor, data, lugar, assunto, resumo, número, editor, periódico, congresso.
  - RF3: O sistema deve dar uma mensagem de alerta quando o usuário tentar inserir um item incompleto. Essa mensagem deve questionar o usuário se ele deseja cancelar a operação, completar a informação ou concluir a inserção como está.
    Qual informação é necessária para possibilitar uma inserção incompleta?

Defeitos de Omissão II

- Exemplo 2 (Omissão de Desempenho:)
  - RNF1: O sistema deve fornecer os resultados tão rápido quanto possível.

Tão rápido quanto possível?

Defeito de Fato Incorreto

- Defeito de Fato Incorreto (FI): informação que consta do artefato mas que seja contraditória com o conhecimento que se tem do domínio de aplicação.
- Exemplo: Considere um sistema de empréstimo numa biblioteca e o seguinte RF:
  - RF30: O sistema não deve aceitar devolução de livros se o usuário não estiver com a carteirinha da biblioteca no momento.

Para devolução de livros não é necessário apresentar a carteirinha pois todas as informações já estão registradas no sistema !

Defeito de Inconsistência

- Defeito de Inconsistência (I): informação que consta do artefato mais de uma vez e em cada ocorrência ela é descrita de forma diferente.

- Exemplo: Considere um sistema de empréstimo numa biblioteca e o seguinte RF:
  - RF5: O sistema não deve permitir períodos de empréstimo maiores que 15 dias.
  - RF9: Professores podem retirar livros por um período de 3 semanas.

Qual período deve ser considerado?

Defeito de Ambigüidade

- Defeito de Ambigüidade (A): quando a informação pode levar a múltiplas interpretações.
- Exemplo: Considere um sistema de empréstimo numa biblioteca e o seguinte RF:
  - RF20: Se o número de dias que o usuário está em atraso é menor que uma semana, ele deve pagar uma taxa de R$ 1,00; se o número é maior que uma semana, a taxa é de R$ 0,50 por dia.

Qual a taxa a ser paga se o período for de uma semana?
No primeiro caso, a taxa deve ser calculada por dia?

Defeito de Informação Estranha

- Defeito de Informação Estranha (IE): qualquer informação que, embora relacionada ao domínio, não é necessária para o sistema em questão.
- Exemplo:
  - RF15: Quando um novo livro é adicionado ao acervo, ele permanece em uma prateleira especial por um período de um mês.

Essa informação não é necessária ao sistema !!

Técnicas de Leitura para Inspeção I

- Como detectar defeitos?
  - Lendo o documento.
  - Entendendo o que o documento descreve.
  - Verificando as propriedades de qualidade requeridas.
- Problema:
  - **Em geral não se sabe como fazer a leitura de um documento!!!**
- Razão:
  - Em geral, os desenvolvedores aprendem a escrever documento de requisitos, código, projeto, mas não aprendem a fazer uma leitura adequada dos mesmos.

Técnicas de Leitura para Inspeção II

- Solução:
  - Fornecer **técnicas de leitura** bem definidas.
- Benefícios:
  - Aumenta a relação custo/benefício das inspeções.
  - Fornece modelos para escrever documentos com maior qualidade.
  - Reduz a subjetividade nos resultados da inspeção.

Técnicas de Leitura para Inspeção III

SCE306 – Engenharia de Software I

Garantia de Qualidade (SQA)

Revisões de Software

Inspeção de Software

Inspeção em DR

Taxonomia de Defeitos em DR

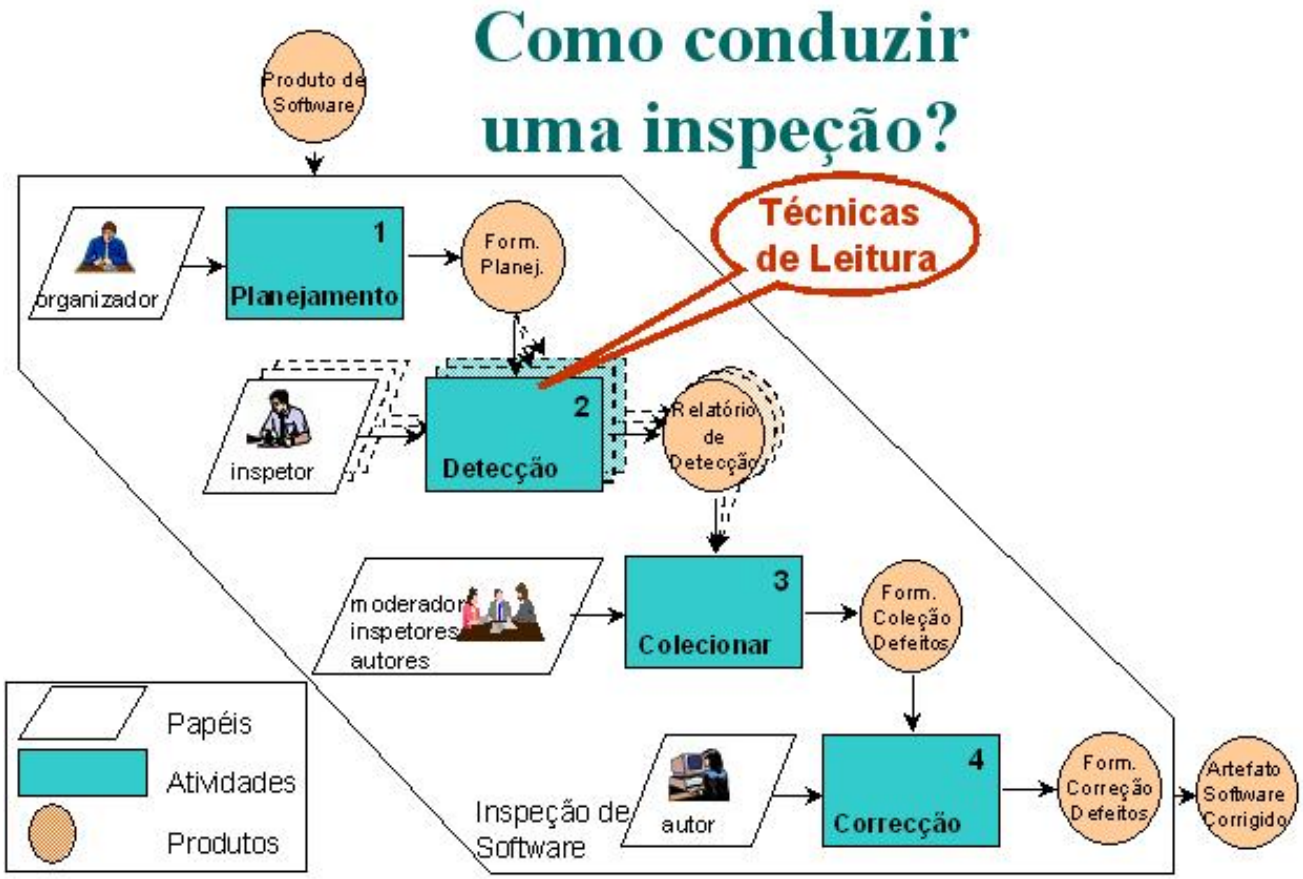
Técnicas de Leitura

Ad-hoc

Checklist

Leitura Baseada em Perspectiva (PBR)

Exercício de Fixação



Técnicas de Leitura para Inspeção IV

- O que é uma técnica de leitura?
  - Conjunto de instruções fornecido ao revisor dizendo como ler e o quê procurar no produto de software.
- Algumas técnicas de leitura:
  - Ad-hoc
  - **Checklist**
  - Leitura Baseada em Perspectiva (PBR)
  - Stepwise Abstraction

Ad-hoc

- Os revisores não utilizam nenhuma técnica sistemática de leitura.
- Cada revisor adota sua própria maneira de ler o documento.
- Desvantagens:
  - Depende da experiência do revisor.
  - Não é repetível.
  - Não é passível de melhoria pois não existe um procedimento a ser seguido.

Checklist

- Técnica que fornece diretrizes para ajudar o revisor alcançar os objetivos de uma atividade de revisão formal.
  - Verificar se o software está de acordo com os seus requisitos.
  - Assegurar que o software está representado de acordo com padrões pré definidos.
  - Cobrir erros de função, de lógica, de implementação em qualquer representação (artefato) de software.

Checklist

- Cada revisor recebe um **checklist**.
  - Os itens do checklist capturam lições importantes que foram aprendidas em inspeções anteriores no ambiente de desenvolvimento.
  - Itens do checklist podem explorar defeitos característicos, priorizar defeitos diferentes e estabelecer questões que ajudam o revisor a encontrar defeitos.

Checklist

- Inspetor segue uma lista de itens com características a serem revisadas.
- Resultado final mais direcionado:
  - Características de qualidade definidas a priori.
  - Produtividade individual.
  - Difícil garantir que inspetor leu o documento de forma adequada, mesmo tendo sido definidas as características de qualidade a se procurar.
- Cobertura do documento relacionada aos itens do checklist.
- Custo/eficiência depende do checklist e dos inspetores.
- Checklist pode ser adaptado ou construído para capturar uma determinada especificidade.

Checklist I

- **Questões Gerais**
  1. Os objetivos do sistema foram definidos?
  2. Os requisitos estão claros e não ambíguos?
  3. Foi fornecida uma visão geral da funcionalidade do sistema?
  4. Foi fornecida uma visão geral das formas de operação do sistema?
  5. O software e o hardware necessários foram especificados?
  6. Se existe alguma suposição que afete a implementação ela foi declarada?
  7. Para cada função, os requisitos foram especificados em termos de entrada, processamento e saída?
  8. Todas as funções, dispositivos e restrições estão relacionadas aos objetivos do sistema e vice-versa?

Checklist II

- **Checklist de Omissão - Funcionalidade**
  1. As funções descritas são suficientes para alcançar os objetivos do sistema?
  2. As entradas declaradas para as funções são suficientes para que elas sejam executadas?
  3. Foram considerados os eventos indesejáveis e as respostas a eles foram especificadas?
  4. Foram considerados o estado inicial e os estados especiais (por ex. inicialização do sistema, término anormal)?

Checklist III

- **Checklist de Omissão - Desempenho**
  1. O sistema pode ser testado, analisado ou inspecionado para mostrar que ele satisfaz seus requisitos?
  2. Os tipos de dados, unidades, limites e resolução foram especificados?
  3. A freqüência e volume de entrada e saída foram especificados para cada função?

- **Checklist de Omissão - Interface**
  1. As entradas e saídas para todas as interfaces são suficientes?
  2. Foram especificados os requisitos de interface entre hardware, software, pessoas e procedimentos?

Checklist IV

- **Checklist de Omissão - Recursos do Ambiente**
  1. Foram especificadas de forma apropriada as funcionalidades de interação entre hardware, software com o sistema?

- **Checklist de Informação Estranha**
  1. Todas as funções especificadas são necessárias para alcançar os objetivos do sistema?
  2. As entradas das funções são necessárias para executá-las?
  3. As entradas e saídas das interfaces são necessárias?
  4. As saídas produzidas por uma função são usadas por outra função ou transferidas para a interface externa?

Checklist V

- **Checklist de Ambigüidade**
  1. Cada requisito foi especificado de forma discreta, não ambígua e testável?
  2. Todas as transições do sistema foram especificadas de forma determinística?
- **Checklist de Inconsistência**
  1. Os requisitos estão consistentes entre si?
- **Checklist de Fato Incorreto**
  1. As funções especificadas são coerentes com o sistema e com os objetivos a serem alcançados?

Checklist VI

Lista de Defeitos

Nro Sequencial	Local no Doc. Requisitos	Página	Tipo do Defeito	Descrição
1	RF5	10	O	Não discriminadas as informações necessárias para que seja feito o cadastro da pessoa.
2	RF12	18	A	Não fica claro qual a taxa que deve ser paga, no caso de atraso de livro

de acordo com a taxonomia de erros

nro da seção ou do requisito no doc. de requisitos

uma explicação que dê para entender porque o inspetor considera que aquilo seja um defeito

Técnica de Leitura Baseada em Perspectiva (PBR) I

SCE306 – Engenharia de Software I

Garantia de Qualidade (SQA)

Revisões de Software

Inspeção de Software

Inspeção em DR

Taxonomia de Defeitos em DR

Técnicas de Leitura

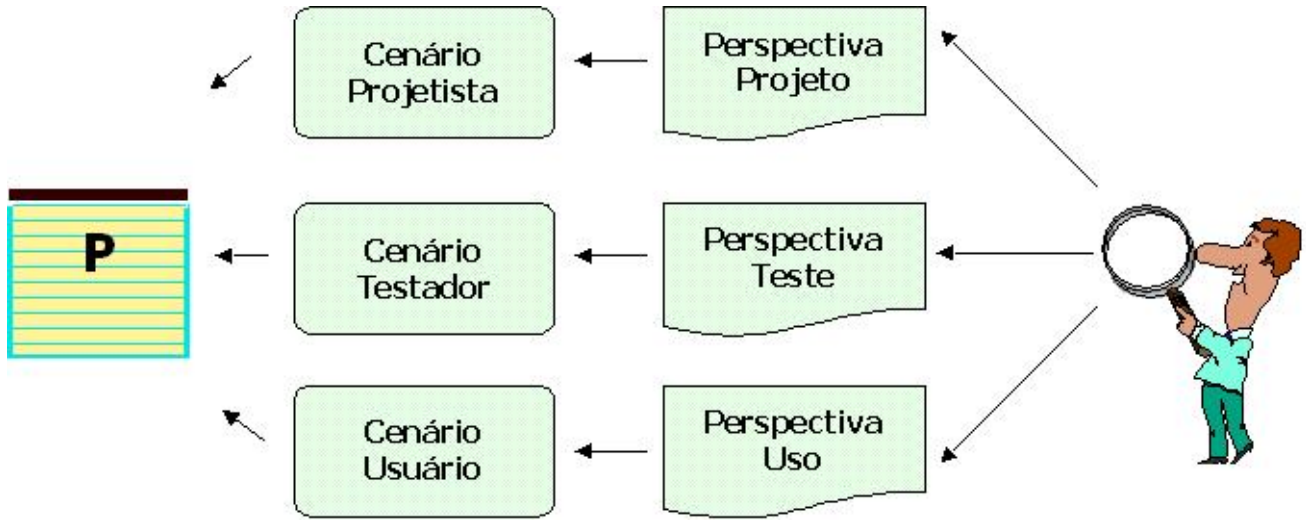
Ad-hoc

Checklist

Leitura Baseada em Perspectiva (PBR)

Exercício de Fixação

- Técnica de leitura proposta para detectar defeitos em **especificações de requisitos**.
- Várias leituras podem ser feitas no Documento de Requisitos.



Técnica de Leitura Baseada em Perspectiva (PBR) II

SCE306 – Engenharia de Software I

Garantia de Qualidade (SQA)

Revisões de Software

Inspeção de Software

Inspeção em DR

Taxonomia de Defeitos em DR

Técnicas de Leitura

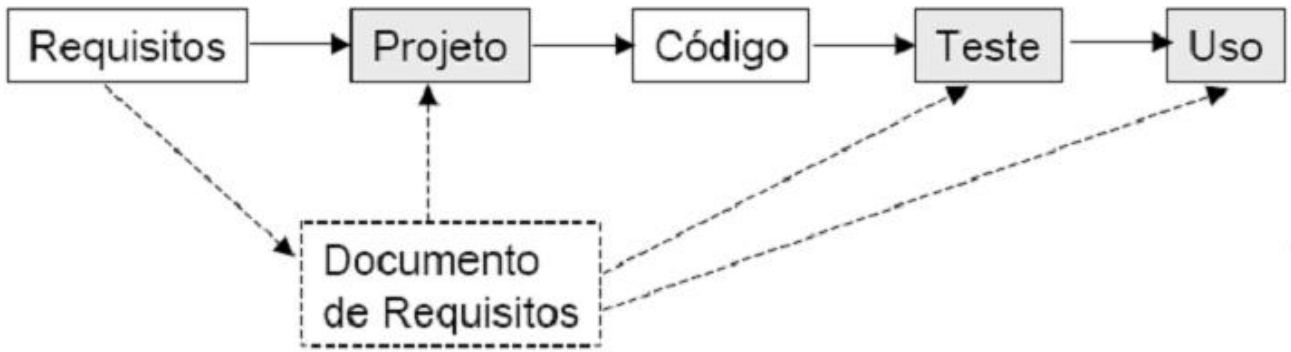
Ad-hoc

Checklist

Leitura Baseada em Perspectiva (PBR)

Exercício de Fixação

- Várias leituras podem ser feitas no Documento de Requisitos.
  - O projetista que usa o DR para gerar o projeto do sistema.
  - O testador que, com base no DR, deve gerar casos de teste para testar o sistema quando este estiver implementado.
  - O usuário que verifica se o DR está capturando toda funcionalidade que ele deseja para o sistema.



Técnica de Leitura Baseada em Perspectiva (PBR) III

- Faz com que cada revisor se torne responsável por uma perspectiva em particular.
  - Possibilita que o revisor melhore sua experiência em diferentes aspectos do documento de requisitos.
  - Assegura que perspectivas importantes sejam contempladas.
- Benefícios:
  - Determina uma responsabilidade específica para cada revisor.
  - Melhora a cobertura de defeitos.

Exercício de Fixação

SCE306 – Engenharia de Software I

Garantia de Qualidade (SQA)

Revisões de Software

Inspeção de Software

Inspeção em DR

Exercício de Fixação