MBA em Engenharia de Software à Distância

Verificação, Validação e Teste de Software

Profª. Cleziana de Freitas Costa
Cleziana.costa@gmail.com

MBA em Engenharia de Software à Distância

Você está aqui!

- Aula 1 • Fundamentos em Teste
- Aula 2 • Verificação de Software
- Aula 3 • Validação de Software
- Aula 4 • Dimensões do Teste
- Aula 5 • Processo de Teste
- Aula 6 • Ferramentas de Teste

MBA em Engenharia de Software à Distância

Objetivos da Aula

- ✓ Narrar a história do Teste
- ✓ Conceituar Teste
- ✓ Explicar as visões do teste
- ✓Descrever “Bug”
- ✓Explicar porque um “Bug” ocorre
- ✓Listar os vários tipos de Defeitos
- ✓Identificar onde estão os defeitos
- ✓Descrever porque o teste é importante
- ✓Explicar quanto teste é suficiente.

História do teste

✓ **Teste nas décadas de 1960 e 1970 – Eram feitos pelos desenvolvedores**
As atividades de teste eram consideradas um mal necessário para provar aos usuários que os produtos funcionavam.

✓ **Teste a partir dos anos 1980 – Processo formal**
As atividades de teste passaram a ser tratadas como processo formal
Surgiram as metodologias de teste que evoluíram até os dias de hoje.

✓ **Teste hoje – Alta importância**
Os fatores segurança e performance passam a ser de alta importância, tornando a atividade de testar cada vez mais especializada.

Teste

Definição de Teste

Testar é verificar se o software está fazendo o que deveria fazer, de acordo com os seus requisitos, e não está fazendo o que não deveria fazer.
(Rios e Moreira - 2002);

Testar é o processo de executar um programa ou sistema com a intenção de encontrar defeitos (teste negativo).
(Glen Myers - 1979);

Testar é qualquer atividade que a partir da avaliação de um atributo ou capacidade de um programa ou sistema, seja possível determinar se ele alcança os resultados desejados.
(Bill Hetzel - 1998).

Teste

Teste de software é o processo que visa executar o software de forma controlada, com o objetivo de avaliar o seu comportamento, baseado no que foi especificado.

Teste

- **Teste prova que tudo está bem e funcionando adequadamente?**
- **Desenvolvedores querem provar que "algo funciona".**
- **Analistas de teste querem provar que "algo não funciona".**

Analisando o esforço dos testes podemos perceber que é mais fácil provar que "**algo funciona**" do que provar que "**algo não funciona**".

Teste

“TESTES SERVEM PARA DIMINUIR O RISCO”

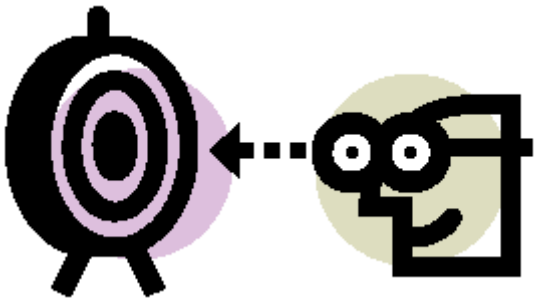
Visão do teste

Profissional de teste

“Inimigo dos desenvolvedores, dos analistas de requisitos”

“Trabalha para apontar erros dos outros”

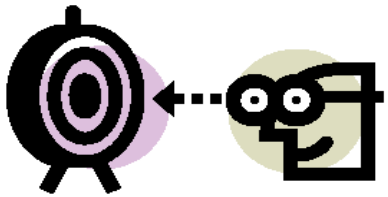
Todos que trabalham com o projeto têm que ter um único objetivo produzir produtos melhores para satisfazer o cliente.



Visão do teste

Visão do analista de sistemas

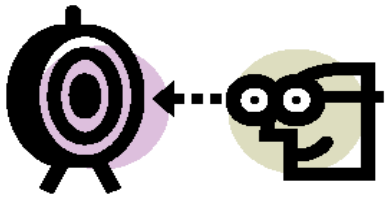
- Objetivo provar que as coisas estão funcionando;
- Testa os cenários positivos.



Visão do teste

Visão do analista de teste

- **Objetivo de provar a não adequação de algo**;
- **Testa os cenários positivos e negativos.**

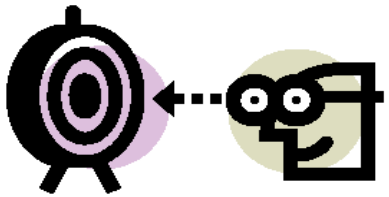


Visão do teste

Como evitar a “falta de visão meta”?

- **Os requisitos de testes devem estar claros;**
- **O plano de teste deve ser elaborado de forma clara;**
- **A equipe de teste deve ser “multiplataforma”;**
- **Os prazos para os testes devem ser devidamente negociados;**
- **Os testes devem ser executados por profissionais de teste**

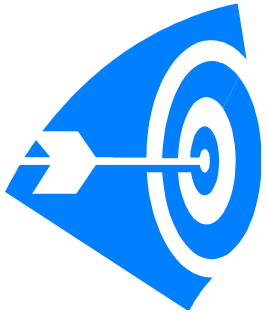
Testar pode ser aparentemente simples, porém quando olhamos com “lupa”, vemos que existe um novo universo logo ali, e no caso em questão, um “projeto dentro de outro projeto”.



Principal meta de um bom testador

A principal meta de um bom testador é:

- Encontrar bugs;
- Registrar os bugs;
- Encaminhar aos responsáveis para consertar.



“A meta de um testador de software é achar “bugs” o mais cedo possível, e fazer com que sejam consertados com certeza.”

“Bug” esquecido ainda é “bug”.

Qual perfil de um bom profissional de teste?

O profissional de teste deve ser:

- Explorador
- Resolvedor de problemas
- Incansável
- Criativo
- Perfeccionista
- Deve Exercitar o julgamento
- Diplomático
- Persuasivo

“Bug”

História do “Bug”

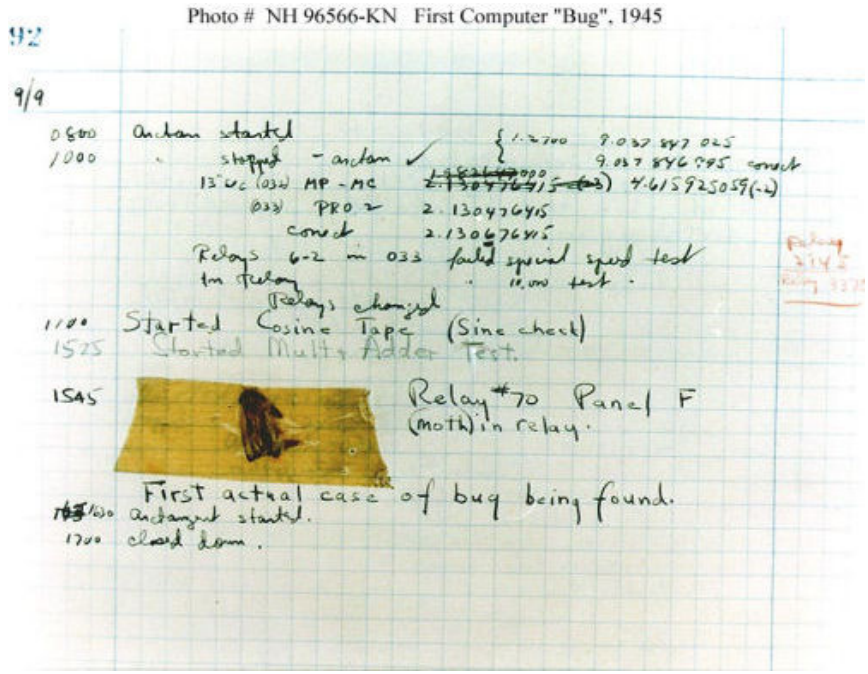
O termo bug foi associado a interferências e mal funcionamento bem antes de que existissem os computadores modernos, sendo Thomas Edison um dos primeiros a usar este significado. Mas foi uma mulher, Grace Murray Hopper, quem em 1945 documentou o primeiro bug da informática.

Bug = Inseto

Para informática adquire outro significado, relacionados a elementos e circunstâncias no software ou hardware, involuntários e indesejados, que provocam um mal funcionamento.

“Bug”

História do “Bug”



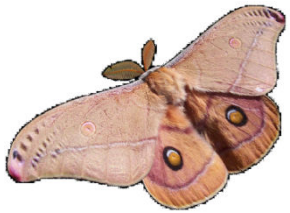
A partir de então, cada vez que algum computador dá problemas dizemos que tem **bugs**.

“Bug”

O que é um “Bug”?

O “Bug” é falha, ou erro em um programa computacional, que o impede de se comportar como pretendido.

Erros mais graves podem fazer com que o programa deixe de funcionar ou congele e conduza a negação de serviço.



“Bug”

Situações de “Bug”

Desde o primeiro bug relatado, sucederam inúmeros incidências relacionados as falhas de programação , Podemos relatar os piores desastres:

- **Desastre de 28 de Julho de 1962: Falha na sonda Mariner 1.**
- **Desastre de 1982: Explosão num gasoduto soviético.**
- **Desastre de 1985-1987: Acelerador médico Therac-25.**

“Bug”

Situações de “Bug”

- Desastre de 1988: O Worm de Morris.
- Desastre de 1988-1996: Gerador de números aleatórios de Kerberos
- Desastre de 15 de Janeiro de 1990: Queda da rede de AT&T.

“Bug”

Situações de “Bug”

- Desastre de 1995/1996: O Ping da Morte.
- Desastre de 4 de Junho de 1996: Desintegração do Ariane 5.
- Desastre de novembro 2000: Sobredosagem radiológica no Instituto Nacional do Cancro da Cidade do Panamá.

“Bug”

Situações de “Bug”

“Bug do Milênio”

Existem vários outros exemplos, um dos mais conhecidos de todos é o **“Bug do Milênio”.**

Quem não se lembra da onda de pânico coletivo gerada, principalmente nos países onde os computadores eram mais populares.

“Bug”

Por que um Bug ocorre?

Porque programadores erram na lógica de programação. Essa frase nem sempre é verdade, na maioria das vezes a culpa não é dos programadores.

Em pesquisas realizadas comprovou-se que a maioria dos erros encontrados são devido a **requisitos mal elaborados.**

Se a funcionalidade desejada não é especificada ou comunicada de forma correta, seria como especificar algo incerto ou até incorreto.

Testes com precisão

Para testar devemos planejar os testes com precisão!

O teste deve ser possuir:

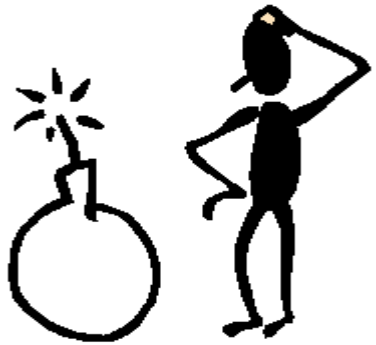
- **Planejamento;**
- **Execução dos testes;**
- **Avaliação dos testes.**

“Cuidado! Quando não se encontra “bug” na aplicação, o teste pode estar viciado”.

Defeitos

O Teste reduz a probabilidade que os defeitos permaneçam em um software,

mas mesmo se nenhum defeito for encontrado, não prova que ele esteja perfeito.



Defeitos

Os defeitos podem ser chamados de erros, problemas, falhas, ocorrências, incidentes, não conformidades e inconsistência, bugs, crash, abends.

- Falha, defeito representam de forma semelhante "**o software falhou em não funcionar corretamente ou em não fazer a coisa correta**".
- Anomalia, incidência, variança representam um **"funcionamento eventual ou involuntário do software"**.
- Problema, erro, **"bug"** são **termos genéricos.**

Defeitos

Conforme Glossário ISTQB:

- **Defeito** - Falha em um componente ou sistema que pode fazer com que o componente ou sistema falhe ao desempenhar sua devida função.
- **Erro** - Ação humana que produz resultados incorretos.
- **Falha** - Desvio do componente ou sistema da entrega, resultado ou serviço esperado.

Defeitos

Conforme Syllabus:

“O ser humano está sujeito a cometer um erro, que produz um defeito, no código, em um software ou sistema ou em um documento. Se um defeito no código for executado, o sistema falhará ao tentar fazer o que deveria, (ou algumas vezes, o que não deveria), causando uma falha. Defeitos no software, sistemas ou documentos resultam em falhas, mas nem todos os defeitos causam falhas.”

Defeito segundo QAI

O QAI resume os defeitos em três categorias:

- **Errado**
- **Perdido**
- **Extra**

Afirma também que **nem todo defeito provoca uma falha, o defeito encontrado em um documento poderia ser consertado antes que provocasse uma falha.**

O que é um defeito?

"**É uma não-conformidade em relação ao que o software se propõe a fazer, que diz que faz e não faz**."

Leonardo Molinari

"**Resultado de um erro existente num código ou num documento**".

Anderson Bastos, Emerson Rios, Ricardo Cristalli, Trayahu Moreira.

Tipos de Defeitos

Existem vários tipos de defeitos:

- **Defeitos de interface com os usuários**
- **Defeitos no manuseio de defeitos**
- **Defeitos de limites**
- **Defeitos de cálculo**
- **Defeitos de iniciação ou fechamento**
- **Defeitos de controle de fluxo**
- **Defeitos de manuseio ou de interpretação de dados**
- **Defeitos de condições de disputa**
- **Defeitos de carga**
- **Defeitos de hardware ou software**
- **Defeitos de controle de versões**

Tipos de Defeitos

- **Defeitos de interface com os usuários**

  - **Defeitos da funcionalidade** – Ocorre quando o software executa uma ação não especificada nos seus requisitos ou quando o requisito especifica uma coisa e o software faz outra.

  - **Defeitos de usabilidade** – São defeitos decorrentes da dificuldade para se navegar em uma aplicação.

Tipos de Defeitos

- **Defeitos de interface com os usuários**

  - **Defeitos de desempenho** – O programa não atende com rapidez necessária às solicitações do usuário, especialmente no caso dos sistemas interativos.

  - **Defeitos de saída** – Os resultados apresentados pelo software não estão conforme definido pelas especificações fornecidas pelo usuário ou foram mal projetadas dificultando a sua análise.

Tipos de Defeitos

- **Defeitos no manuseio de defeitos**

  - **Prevenção dos defeitos** – O programa não se protege das entradas de dados não previstas que posteriormente são processadas de forma inadequada.

  - **Detecção dos defeitos** – O programa não trata as indicações de defeitos resultados das suas operações.

  - **Recuperação dos defeitos** – Mesmo detectando o defeito, o programa falha porque não trata adequadamente a situação.

Tipos de Defeitos

- **Defeitos de limites**

  O programa não consegue tratar ou trata inadequadamente valores extremos ou fora dos limites estabelecidos.

- **Defeitos de cálculo**

  O programa efetua um cálculo e produz um resultado errado.

- **Defeitos de inicialização ou fechamento**

  Quando o programa ou rotina é usado pela primeira vez ou é executado, cria um arquivo em disco, a ausência deste arquivo poderá causar problemas nas etapas seguintes do processamento.

Tipos de Defeitos

- **Defeitos de controle de fluxo**

  Controla, em determinadas circunstâncias, o que o programa fará a seguir. Um defeito deste tipo leva o programa fazer coisas erradas, parando ou tendo comportamento inesperado.

- **Defeitos de manuseio ou de interpretação de dados**

  Ocorre quando um programa tem que passar um grupo de dados para outro programa. No processo, os dados podem ser corrompidos ou interpretados incorretamente.

- **Defeitos de condições de disputa**

  Ocorre quando o programa espera pela resposta dos eventos A e B, sendo que é suposto que A sempre termine primeiro. Se por algum problema B terminar primeiro, o programa pode não estar preparado para esta situação e apresentar resultados inesperados.

Tipos de Defeitos

- **Defeitos de carga**

  O programa não pode suportar um pico de serviço em um determinado momento ou uma carga alta de serviço por um tempo muito prolongado.

- **Defeitos de hardware ou software**

  São defeitos decorrentes da incompatibilidade entre o programa e o ambiente onde está sendo processado, gerando falhas de comunicação entre ambos.

- **Defeitos de controle de versões**

  Quando a versão de um programa é ligada com versões diferentes de outros componentes.

Tipos de Defeitos

O defeito está associada a vários fatores como por exemplo:

- **o processo de software;**
- **a execução das atividades;**
- **o ambiente de trabalho;**
- **o hardware;**
- **o software;**

Por que um defeito ocorre?

Para evitar que defeitos ocorram, temos que nos concentrar nas **fases iniciais dos projetos, nas especificações,** é nessa fase que a comunicação tem que ser clara para que a especificação seja feita de forma correta.

Por que um defeito ocorre?

O defeito ou um “Bug” ocorre quando o software:

- Não faz o que a especificação diz para fazer;
- Faz algo que na especificação diz para não fazer;
- Faz algo que a especificação do produto não menciona;
- Não faz algo que a especificação do produto menciona;
- É difícil de entender, difícil de usar, lento, ou simplesmente aos olhos do testador o “usuário não gostará”.

Onde estão os defeitos?

Os defeitos podem ser encontrados em toda a fase do processo de desenvolvimento do software, causando:

- retrabalho
- aumentando os custos do projeto
- dilatando os prazos de entrega do software
- diminuindo a produtividade
- aumentando a insatisfação do cliente.

O custo da propagação dos defeitos

Segundo estudos de Myers

Quanto mais tardiamente descobrimos os erros, mais caros estes se tornam.

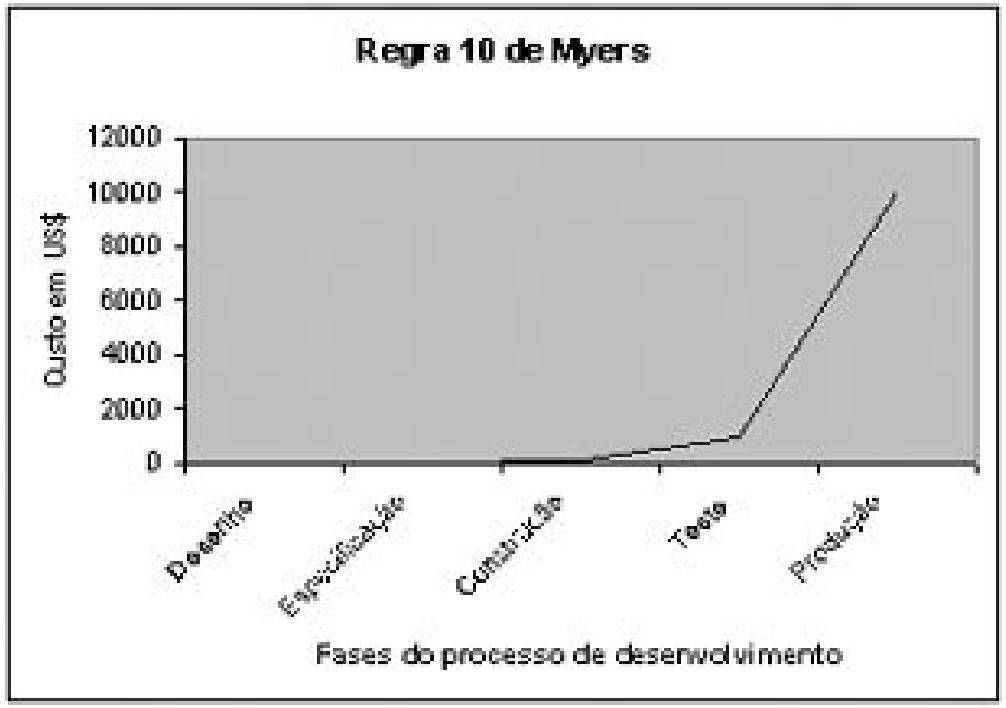
Myers aplica a chamada "**regra 10**" aos custos da correção dos erros. Significa:

Quando um erro não é identificado, os custos de sua correção multiplicam-se por 10 para cada fase em que o erro migra.

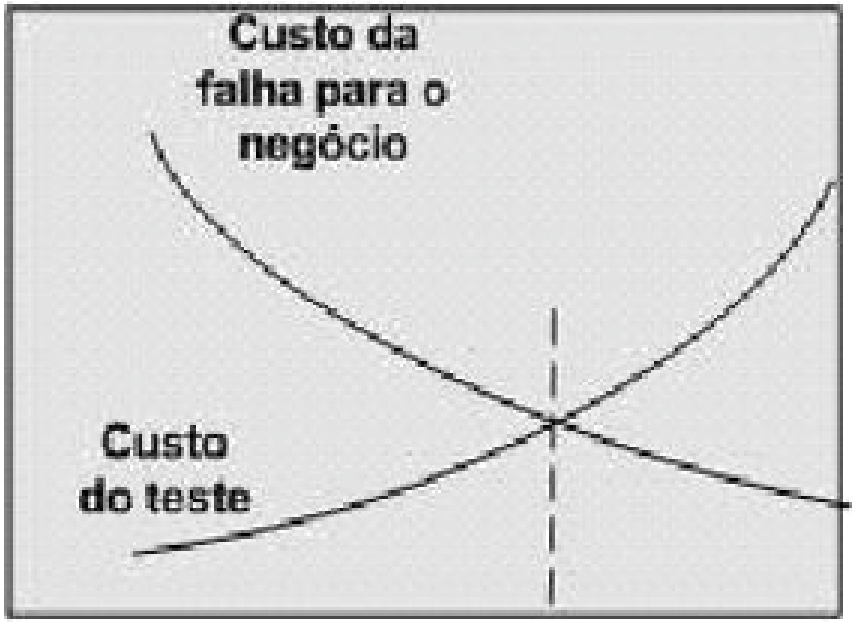
O custo da propagação dos defeitos

“Erros na produção são extremamente caros.”

O custo da propagação dos defeitos



O custo da propagação dos defeitos



Por que as empresas não testam?

As empresas não testam porque:

O Software é complexo.
O Software é intangível.
O Software é altamente modificável.

Teste lida com pessoas.

Por que as empresas não testam?

O teste de software exige:

- **Conhecimento**
- **Planejamento**
- **Projeto**
- **Execução / Acompanhamento**
- **Integração com outras áreas**
- **Recursos.**

Porque o teste é necessário?

Contexto dos sistemas de softwares

Problemas que pode ocorrer devido a falta de teste:

- ***Perda de dinheiro;***
- ***Perda de tempo;***
- ***Má reputação da empresa;***
- ***Morte de pessoas;***
- ***Ferimentos de pessoas***

Porque o teste é necessário?

Testar é investir

“No futuro, para reduzir os riscos de que falhas aconteçam”

O desenvolvedor testa alguma coisa que deveria falhar e ela simplesmente funciona.

Sempre que um teste detecta uma falha logo após ela surgir no sistema, a equipe ganha tempo, pois tem a oportunidade de corrigir rapidamente.

Porque o teste é necessário?

Causas dos defeitos de software

O ser humano está sujeito a cometer erro, que pode produzir defeito:

- No código
- No software
- No sistema
- No documento

Defeito no código, o sistema falhará ao tentar fazer o que deveria, causando uma falha.

Defeitos nos softwares, sistemas ou documentos podem ou não resultar em falhas.

Porque o teste é necessário?

Causas dos defeitos de software

Os defeitos ocorrem porque:

- ***Os seres humanos são passíveis de falhas;***
- ***Existe pressão no prazo;***
- ***Os códigos são complexos;***
- ***A infra-estrutura é complexa;***
- ***Há mudanças na tecnologia;***
- ***Há muitas interações no sistema.***

Porque o teste é necessário?

Função do teste no desenvolvimento, manutenção e operação de software.

Problemas que levam a abandonar os testes:

- ***Teste é uma atividade chata e difícil;***
- ***Teste consome tempo;***
- ***Teste tem pouca importância;***
- ***Elevado o número de teste.***

Porque o teste é necessário?

Função do teste no desenvolvimento, manutenção e operação de software.

Teste requer:

- **Vários detalhes;**
- **Muitas variações**
- **Várias possibilidades para realizar o testes**

Porque o teste é necessário?

Função do teste no desenvolvimento, manutenção e operação de software.

O teste é necessário porque:

- **Reduz os riscos de ocorrência de problemas;**
- **Contribui para a qualidade dos sistemas de software;**
- **Defeitos corrigidos antes da implantação do software;**

Teste deve ser algo simples de ser feito e não deve consumir um tempo exorbitante.

Porque o teste é necessário?

Teste e Qualidade

Com o teste é possível medir a qualidade do software em termos de defeitos encontrados

- **Teste pode dar a confiança na qualidade do software;**
- **Teste planejado adequadamente reduz os riscos de erros no sistema.**

Quando os testes encontram defeitos, a qualidade do sistema aumenta quando estes são corrigidos.

Porque o teste é necessário?

Teste e Qualidade

Lições devem ser aprendidas de projetos anteriores, assim melhoramos a qualidade dos sistemas futuros.

Porque o teste é necessário?

Quanto teste é suficiente?

Devemos levar em consideração:

- **O nível do risco, incluindo risco técnico**
- **O nível do negócio e do projeto**
- **O tempo do projeto**
- **O orçamento do projeto**

O teste deve prover informações suficientes aos interessados para tomada de decisão sobre a distribuição do software ou sistema, para as próximas fases do desenvolvimento ou implantação nos clientes.

MBA em Engenharia de Software à Distância

Baseado no livro Testes de Software

Leonardo Molinari

MBA em Engenharia de Software à Distância

Parabéns!

Até a próxima aula!