



Teste de software

Semestre 1/2018

Profs. Ricardo Ajax



Verificação e Validação

Ferramentas para aferição da qualidade



Produtos de software, gerados pelos seus processos produtivos



- O que é Software
- O que pode não funcionar direito em um software
- Verificar x Validar: ambos podem ser avaliados por meio de testes (estáticos ou dinâmicos)
- Erro
- Defeito
- Falha

E agora? Quais os focos de qualidade são importantes para o usuário ao definir o software por ele pretendido?



Mas o que é qualidade?

IEEE:

- o grau no qual o sistema, componente ou processo satisfaz os requisitos especificados
- O Grau no qual o sistema, componente ou processo satisfaz as necessidades ou expectativas do cliente (usuário)



Mas o que isso significa?

Conformidade com os requisitos (Crosby, 1979)

Satisfazer as necessidades e expectativas do cliente (usuário), a fim de prover a sua satisfação



Mas o que isso significa?

Conformidade com os requisitos (Crosby, 1979)

Satisfazer as necessidades e expectativas do cliente (usuário), a fim de prover a sua satisfação

Qualidade consiste na ausência de deficiências (Juran, 1988)



Pressman expande o conceito de qualidade de software

Conformidade aos requisitos funcionais e de desempenho, padrões explicitamente documentados e características implícitas que são esperadas por todos os profissionais de desenvolvimento de software



Pressman expande o conceito de qualidade de software

Conformidade aos requisitos **funcionais** e de **desempenho, padrões explicitamente documentados** e características implícitas que são **esperadas** por todos os profissionais de desenvolvimento de software



Significando:

- Requisitos funcionais específicos, referentes principalmente às saídas providas pelo sistema de software
- Aos padrões de qualidade mencionados nos contratos (Ex; Indicadores de densidade de defeitos)
- Às boas práticas de engenharia de software que refletem o estado da arte do desenvolvimento, a serem satisfeitas pelos desenvolvedores mesmo que não explicitamente declaradas nos contratos

(Galin, 2004)



Entende-se software como sinônimo de produto de software que é o conjunto de **programas** de computador, procedimentos e possível **documentação**, além de **dados associados**

Tudo que possa dar problema em um software é foco de ser testado, mas tudo que É expectativa do cliente em termos de o satisfazer é também foco de ser testado, pois Testes verificam se está correto em termos técnicos e em termos negociais.

Então:

Testes é uma questão de Qualidade e avalia o que foi requerido pelo usuário (Cliente)



Conceitos:

Verificar X Validar

Erro X defeito X falha

Técnicas estáticas de Verificação e Validação X Técnicas dinâmicas (testes de software)

Estáticos:

- Revisões
- Inspeções
- WalkThrougs
- Auditorias

Como a antecedência com a qual o defeito é encontrado torna o seu reparo mais economicamente barato, então usar técnicas estáticas, mesmo antes de haver um código executável é uma boa política para melhorar a qualidade do produto gerado.



Antes de detalhar, alguns aspectos:

- Desk Checking é um processo manual não computadorizado usado para verificar a lógica e o algoritmo antes que o programa seja iniciado. Ajuda os programadores a identificar bugs e erros que possam prevenir defeitos. Atualmente modernas aplicações e ferramentas de debugging tem feito o desk checking menos relevante do que era antigamente. Ele foi o primórdio do Walkthrough
  
- Avaliações Estruturais versus Funcionais
  - O conjunto de testes baseados na análise estrutural tende a descobrir erros que ocorrem durante "a codificação" do programa,
  - O conjunto de testes com base na análise funcional tende a descobrir erros que ocorrem no levantamento de requisitos ou especificações de projetos. Ou seja, o teste funcional assegura que as exigências sejam propriamente satisfeitas na aplicação do sistema.



## Nome: Revisões em pares (peer reviews)

Tipo		
X: Manual   _Automática	_Dinâmica   X: Estática	X: Estrutural   _Funcional

**Descrição:** É uma análise em dupla onde os envolvidos realizam as mesmas funções durante a revisão (papeis análogos no desenvolvimento do software). Pode ser conduzido por uma ou mais duplas e usualmente observa a eficiência, estilo e aderância aos padrões adotados

**Sugestão de Utilização:** Deve ser um processo formal empregado a todo setor do desenvolvimento de software considerado como crítico. Quando esta técnica é usada, geralmente ela é definida pelo processo de desenvolvimento da organização, para ser usada em todos os seus projetos (salvo justificativas específicas para não serem usadas). Os pares devem receber diretrizes durante as revisões para garantir a consistência dos subprodutos revisados. Seus resultados são alvo de relatórios (manuais ou automatizados conforme a organização) e os destinatários são os desenvolvedores do produto.

**Custo de Uso:** Revisão em pares é um trabalho intensivo em que o custo depende do número de pares conduzindo a revisão e do escopo das revisões. Normalmente ela é uma técnica de custo médio.

**Habilidades para Uso:** As duplas devem ter o mesmo nível de conhecimento sobre os principais tópicos.

**Vantagens:** Revisão em pares aplica uma dupla análise para garantir um produto de qualidade. Freqüentemente, essa análise em dupla pode incentivar os membros da equipe quando não existe um supervisor. Também serve para homogeneizar e difundir conhecimentos na organização.

**Desvantagens:** Dificuldade em Criticar colegas de trabalho (ou criticas em excesso), causando problemas de ordem humana e relacionamentos. Neste sentido a gerência deve frizar o aspecto de qualidade perseguido e um subproduto pode ser a necessidade de treinamentos identificada.



## Nome: Inspeções

Tipo		
X: Manual   _Automática	_Dinâmica   X: Estática	X: Estrutural   X: Funcional

**Descrição:** Possuem um nível de formalismo maior, envolvendo outros papéis além dos autores. Existem critérios melhor definidos e predeterminados (envolvendo métodos e técnicas usadas, além da forma do artefato). O processo de inspeção exige revisões minuciosas das entradas e saídas usadas/geradas na elaboração do produto.

**Sugestão de Utilização:** constitui-se em um processo formal empregado ao desenvolvimento de software e não somente a um único projeto. Assim deve possuir critérios claros que justifiquem o seu não uso como definido no processo organizacional, além de diretrizes para serem executados. Os resultados são relatórios (manuais ou não) visando apresentar os resultados de forma clara e concisa aos autores para correção dos artefatos inspecionados.

**Custo de Uso:** É um processo de trabalho intensivo, com o preço dependendo do tamanho da equipe e a profundidade da inspeção. Geralmente ela tem um alto preço.

**Habilidades para Uso:** Além de conhecimentos sobre o artefato sob inspeção (código, análise, projeto e gestão), a equipe deve ter conhecimento sobre o processo a ser seguido.

**Vantagens:** O formalismo e detalhamento das inspeções, embora tenham custos elevados, fornecem uma alta probabilidade de descobrir os problemas, sendo considerada como bastante efetiva.

**Desvantagens:** Justamente a vantagem considerada acima, pode se tornar uma desvantagem por consumir tempo e recursos do projeto. Porém, pela sua característica eminentemente técnica constuma ser bem respeitada pelos envolvidos que confiam nos seus resultados.



## Nome: Walkthroughs

Tipo		
X: Manual   _Automática	X: Dinâmica   _Estática	X: Estrutural   X: Funcional

**Descrição:** Podem ser feitas simulações manuais do sistema usando dados de testes pelo autor do produto ou líder da equipe. Os dados de teste devem ser simples para considerar as limitações de possíveis simulações humanas. Além disso é papel da equipe questionar o produto (e seu funcionamento) levantando hipóteses sobre os impactos de erros que possam gerar defeitos no software oriundos do artefato sobre walkthrough.

**Sugestão de Utilização:** É uma evolução da técnica de Desk Checking (\*). As equipes desta técnica são geralmente constituídas de peritos sobre o(s) assunto(s) da alcada do artefato, além de conhcedores de análise de sistemas e testes .Além disso, existem diretrizes sobre sua condução. Os participantes devem ser treinados no seu uso e os resultados comprehendem identificação de problemas com recomendações sobre suas soluções

**Custo de Uso:** Esta técnica usa intensivamente pessoas, cujos custos variam de acordo com o número de participantes, além do tamanho e complexidade do projeto e artefatos envolvidos. Porém esta técnica tem sido classificada coo de médio custo (nem alto nem baixo).

**Habilidades para Uso:** dependendo do artefato e do objetivo do walkthrough (ex.: validação) pode requerir habilidades do âmbito do usuário no uso do sistema. Em outras ocasiões (avaliações de baixo-nível como o código, ou arquiteturais) podem requerir a presença de representantes deste papel.

**Vantagens:** Fornece uma base para questionar a lógica do programador e ajudar o desenvolvedor a identificar problemas no sistema.

**Desvantagens:** É relativamente desestruturado. A falta de rigor faz com que seja importante a habilidades dos membros da equipe.



## Nome: Auditorias

Tipo		
X: Manual   _Automática	_ Dinâmica   X: Estática	X: Estrutural   X: Funcional
<p><b>Descrição:</b> Realizada principalmente por elementos externos da equipe de desenvolvimento, examinando o produto independentemente, afim de garantir um alto grau de confiança nas especificações, padrões e relatórios.</p>		
<p><b>Sugestão de Utilização:</b> Deve ser independente, realizada sob um ponto de vista ético e com o devido cuidado profissional, seguindo um planejamento, onde o objetivo e escopo é traçado. Costuma ser apoiada por listas de itens (checklists) a serem avaliados, além de questionários aplicados aos participantes. Não conformidades e oportunidades de melhorias são seus principais resultados.</p>		
<p><b>Custo de Uso:</b> É um trabalho intenso envolvendo a participação de terceiros. Ou seja, uma equipe com experiência em auditoria deve ser alocada para manter a independência das avaliações implicando em um custo elevado.</p>		
<p><b>Habilidades para Uso:</b> Os auditores necessitam ter habilidade em auditoria executando boas práticas como motivar a identificação de problemas e não atacar pessoas e sim fatos concretos. Conhecimentos especializados também em desenvolvimento de software são requeridos.</p>		
<p><b>Vantagens:</b> Auditoria fornece uma avaliação independente das conformidades do software e podem gerar boas oportunidades de melhoria ao processo com vantagens para a qualidade do produto final</p>		
<p><b>Desvantagens:</b> Auditores tendem a ficar defasados tecnologicamente em relação ao ambiente computacional da organização devido à crescente complexidade do ambiente computacional. Pode ser complicado encontrar auditores experientes</p>		



## Conclusões:

- Verificar tudo é inviável pelo custo que adiciona ao desenvolvimento do software
- Neste sentido, balancear artefatos críticos e custos de verificação é fundamental para o desenvolvimento geral produtos de alta qualidade
- Considerando que projetos são eventos únicos, gerando produtos também únicos, além de possuirem recursos limitados, a qualidade é (também) uma questão econômica.
- Só é possível identificar o que é crítico no desenvolvimento de software, se existe uma “rotina” a ser seguida, ou seja, se existe um processo
- O uso de técnicas requer processo (e treinamento), consome recursos e gera gastos. Porém, qual o retorno deste investimento
- ROI (Return of Investment): área de pesquisa na engenharia de software, cujo pressuposto reside na máxima defendida pela literatura que, quanto mais se demora a identificar e retirar defeitos, mais caro se torna reparar a falha.



## Dimensões da qualidade

Inicialmente → Norma ISO/IEC 9126 (2003), contendo os focos:

- Qualidade interna
- Qualidade Externa
- Qualidade em uso

Apesar de não atingir todas as empresas, foi uma das normas mais utilizadas (Marinho e Souza, 2010)

MARINHO, D. da S.; SOUZA, E. J. de. Pesquisa de qualidade de software brasileiro 2009. *Brasília: SECRETARIA DE POLÍTICA DE INFORMÁTICA, MINISTÉRIO DA CIÊNCIA E TECNOLOGIA, 2010.*



Objetivos da norma:

- validar a completitude de uma definição de requisitos;
- identificar requisitos de software;
- identificar objetivos de projeto de software;
- identificar objetivos para teste de software;
- identificar critérios para garantia de qualidade;
- identificar critérios de aceitação para produtos finais de software.



## Qualidade Interna

Especificam o nível de qualidade requerido sob o ponto de vista interno do produto.

Os requisitos de qualidade interna são usados para especificar as propriedades dos produtos intermediários. Esses produtos podem incluir: modelos estáticos e dinâmicos e código fonte

A qualidade interna está associada ao processo de produção do software



## Qualidade Externa

Especificam o nível de qualidade requerido sob o ponto de vista externo. Eles incluem requisitos derivados das necessidades de qualidade dos usuários, incluindo os requisitos de qualidade em uso

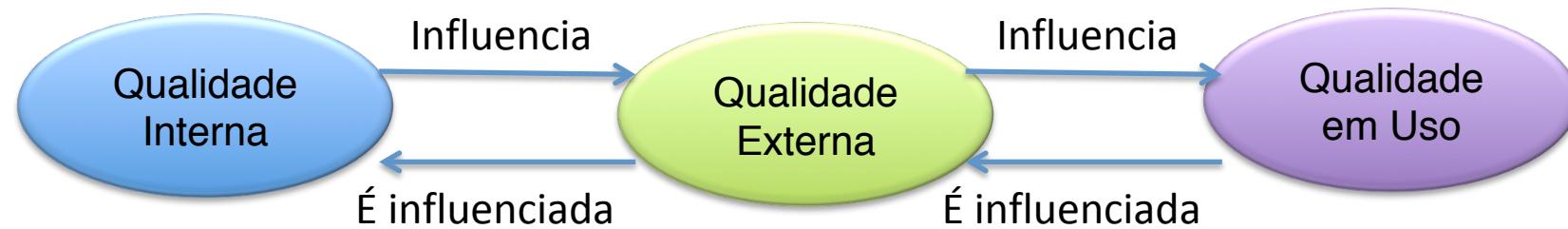
A qualidade externa está associada ao produto de software gerado pelo processo de desenvolvimento / manutenção de software

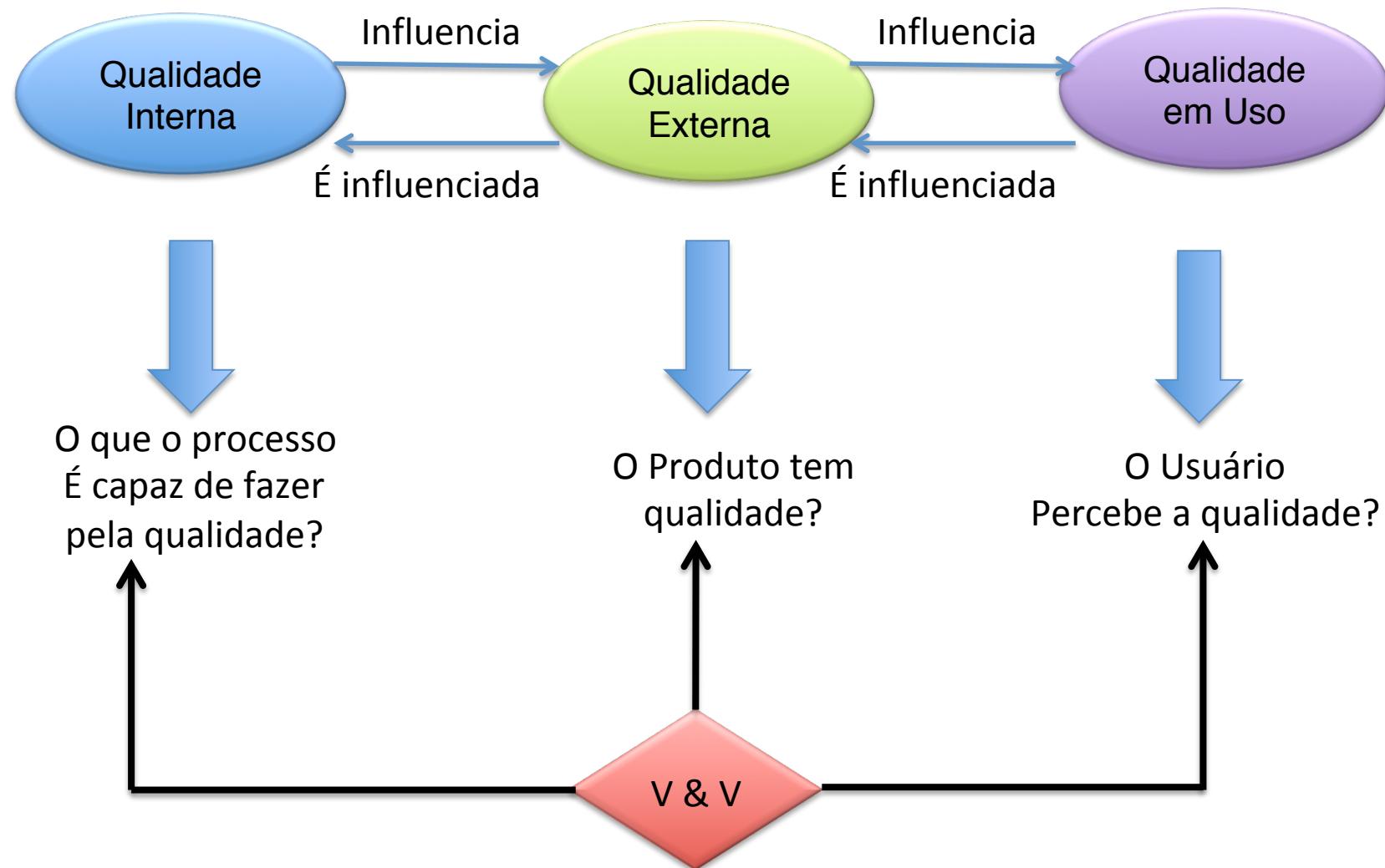


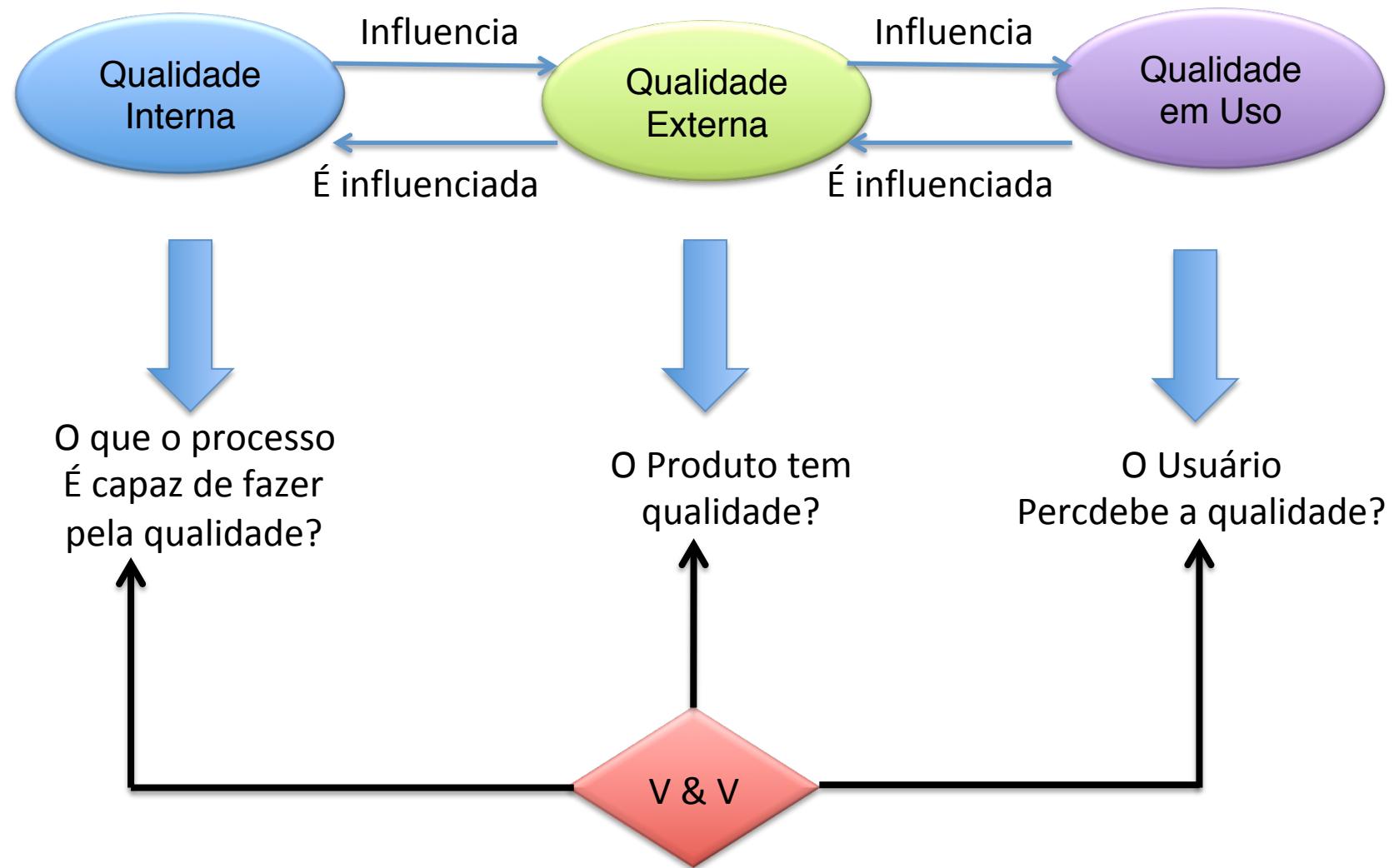
## Qualidade em Uso

É a visão da qualidade do produto de software do ponto de vista do usuário, quando este produto é usado em um ambiente e contexto de uso especificados.

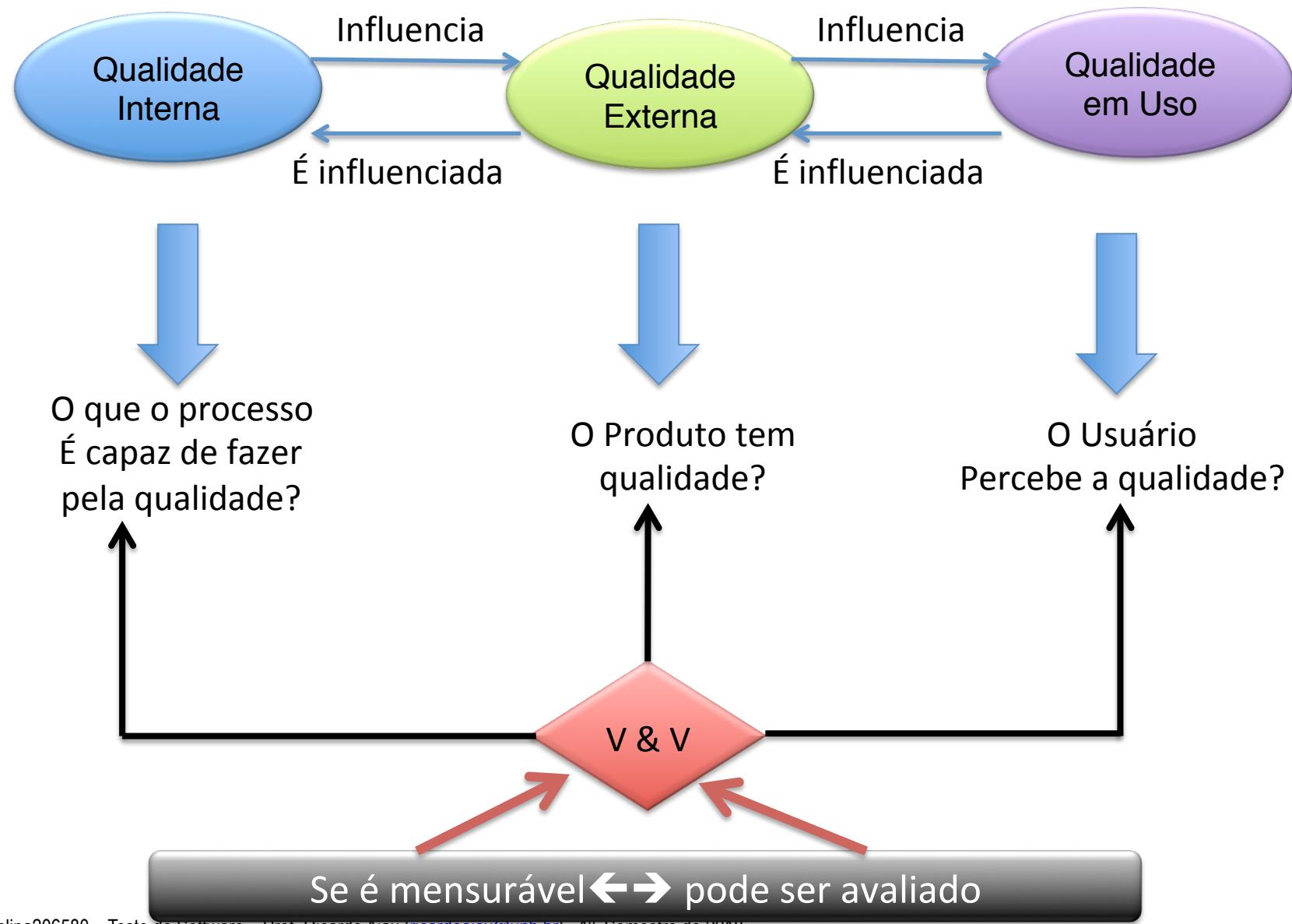
A qualidade em uso está associada ao quanto os usuários podem atingir seus objetivos num determinado ambiente e não as propriedades do software em si

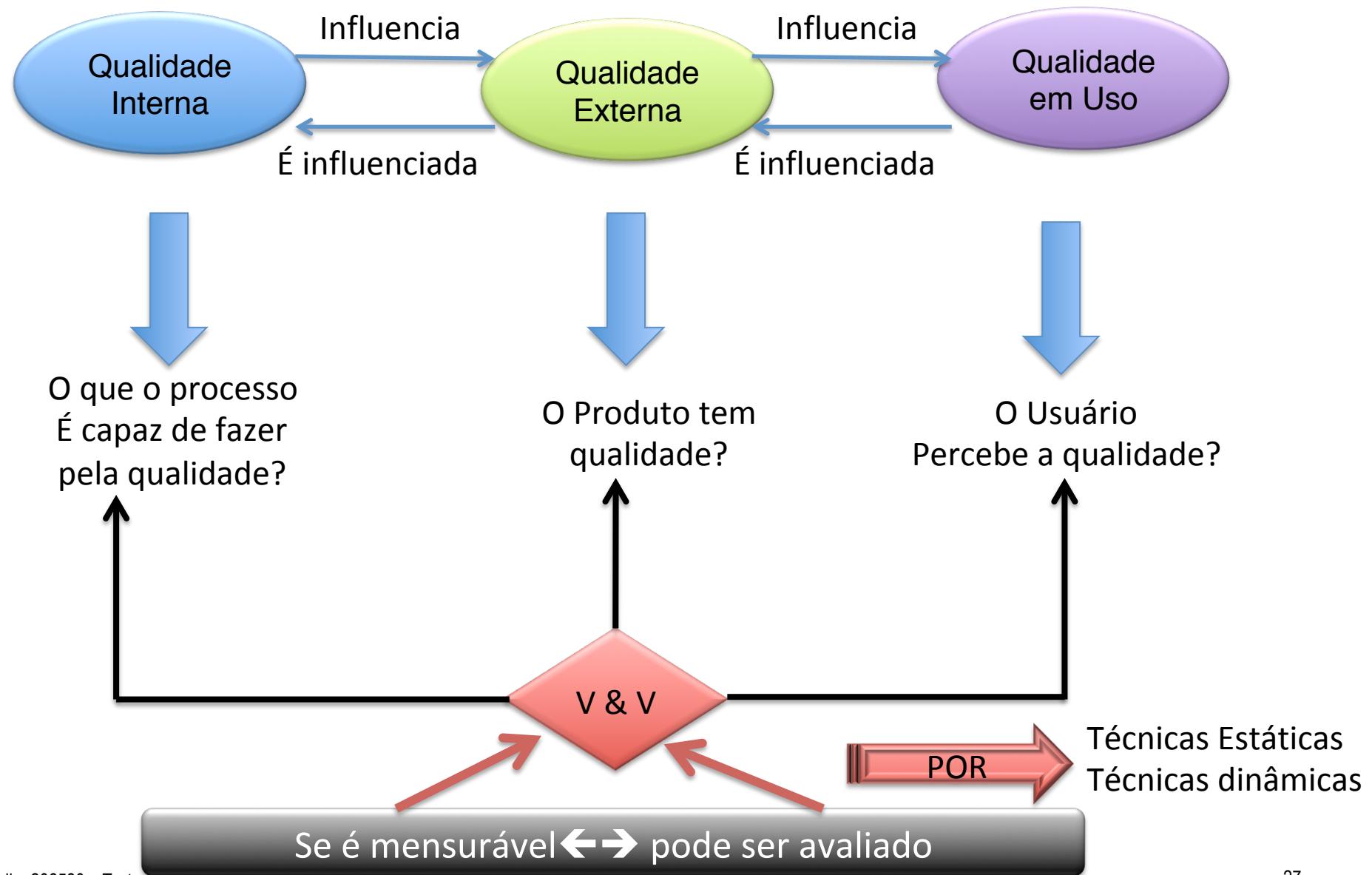


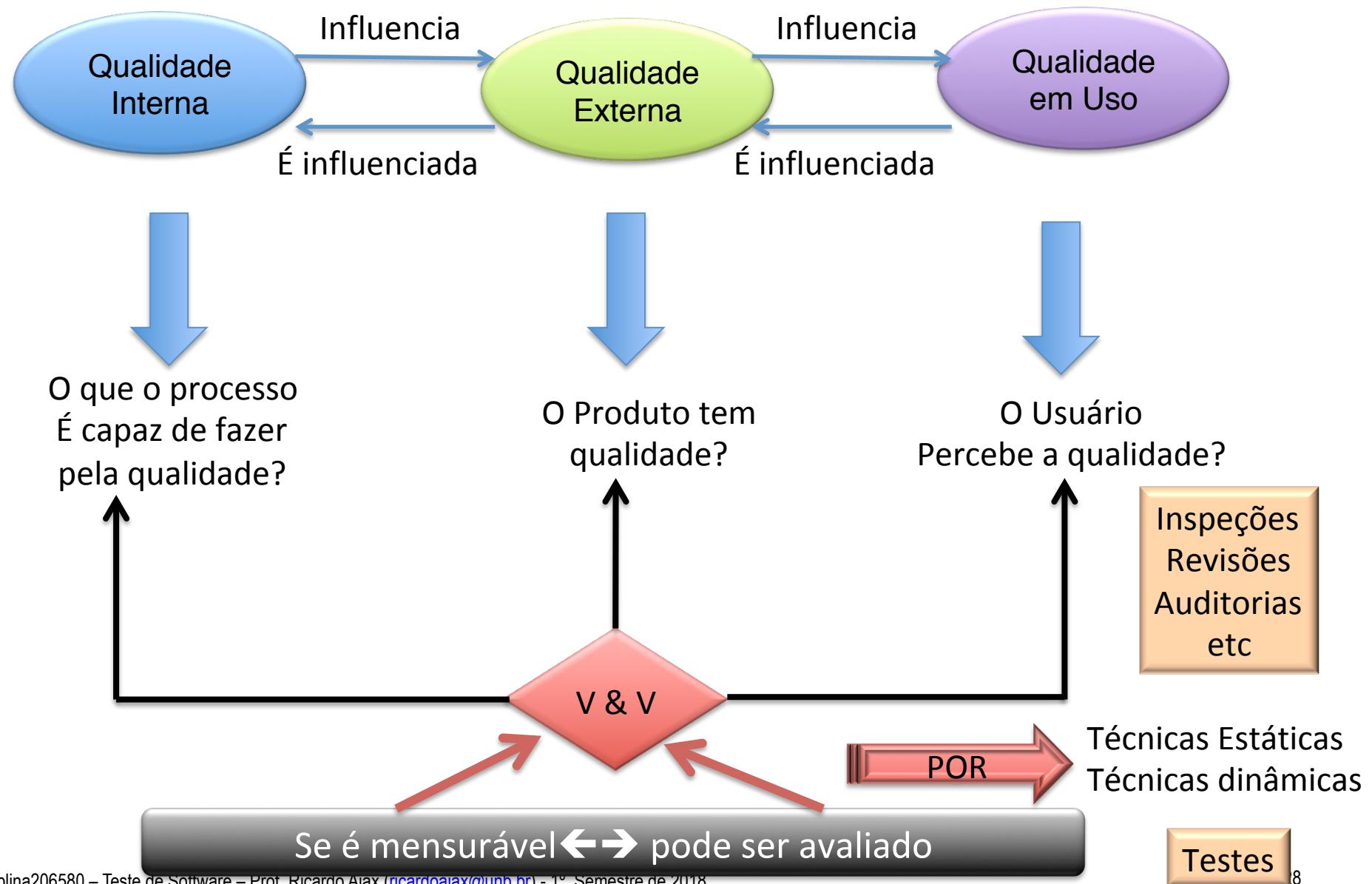


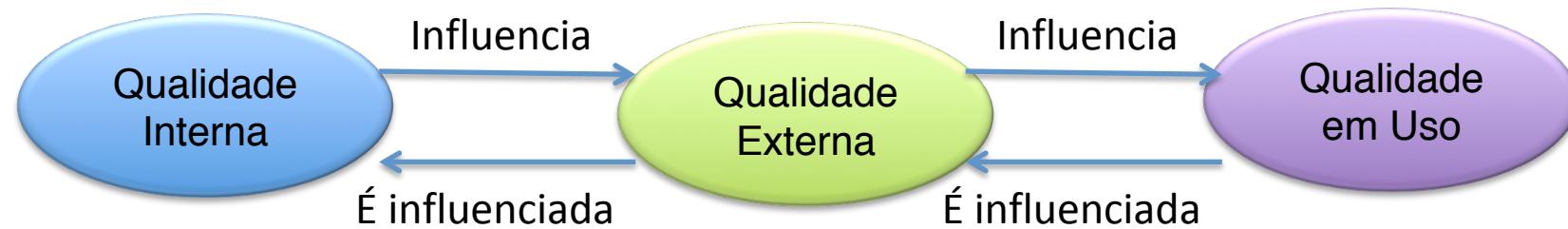


Se sim; É mensurável?

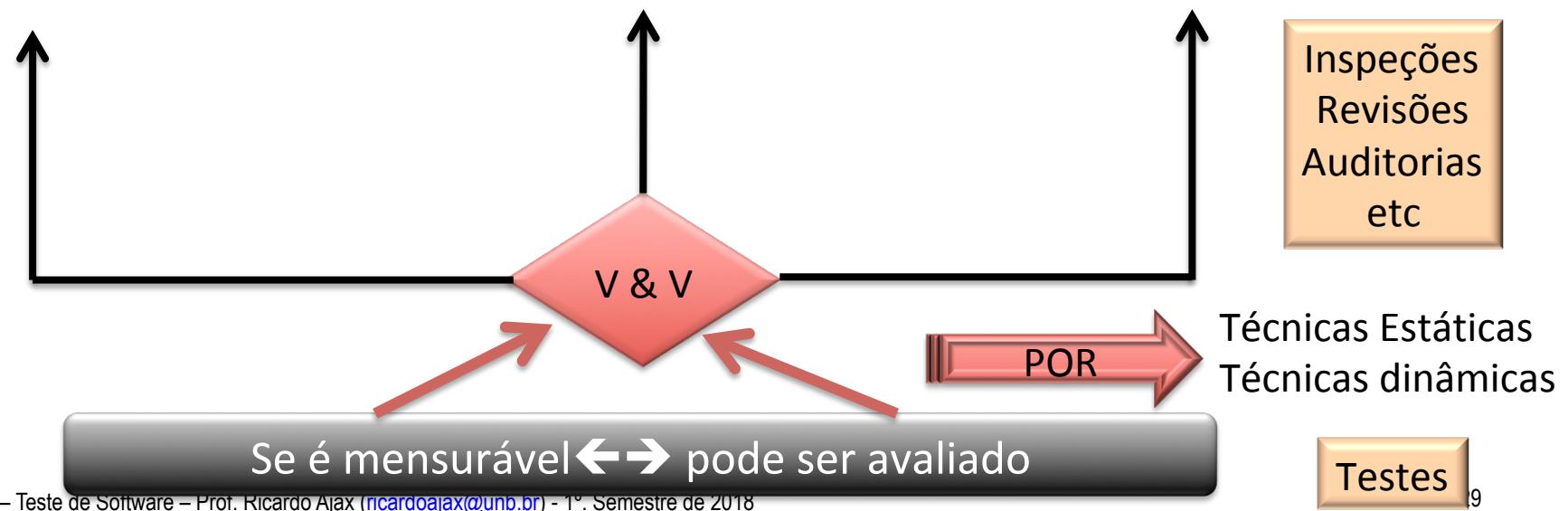


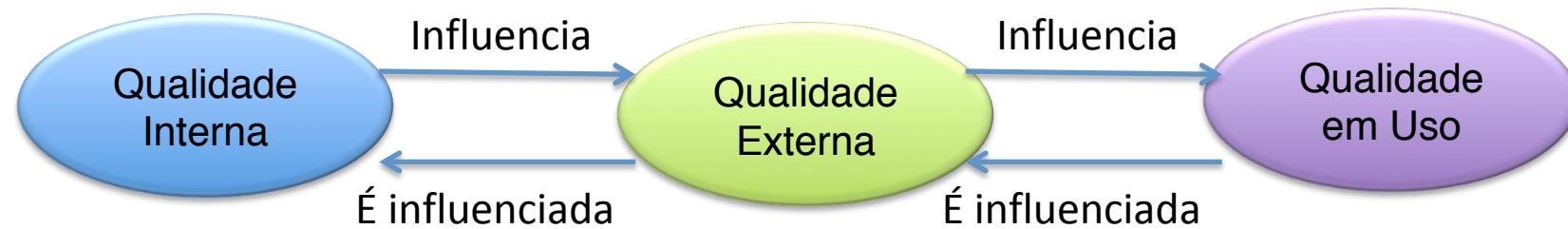






**Problema?**  
**Definições claras, objetivas, conclusivas**





**Problema?**  
**Definições claras, objetivas, conclusivas**

**COMO???**

Requisitos (Funcionais, não funcionais, de qualidade)

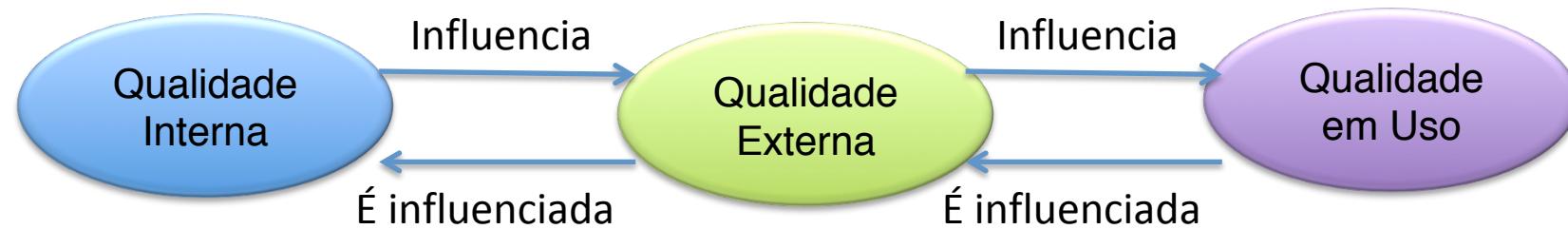


Inspeções  
Revisões  
Auditorias  
etc

Testes

**Antes:** **Controle** de qualidade

**Agora:** **Garantia** da qualidade

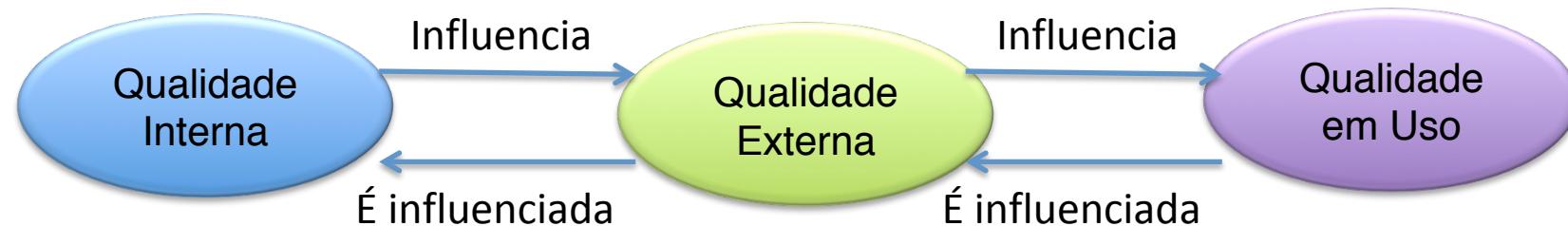


**Então: Quais são as dimensões da qualidade???**

A saber

**F U R P S**

QUE NÃO SATISFAZ?  
É UM DEFEITO QUE PODE SER UMA FALHA  
QUE PODE GERAR UMA REJEIÇÃO DO PRODUTO



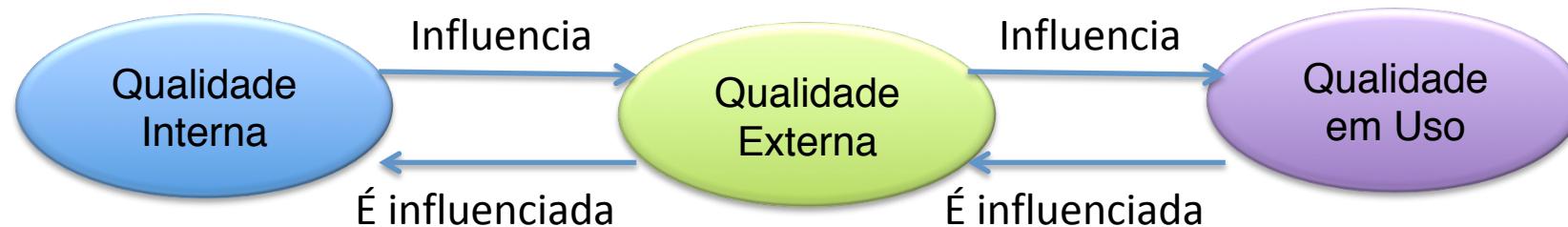
QUE NÃO SATISFAZ?  
É UM DEFEITO QUE PODE SER UMA FALHA  
QUE PODE GERAR UMA REJEIÇÃO DO PRODUTO

DIMINUINDO A SATISFAÇÃO DO CLIENTE QUANTO ÀS SUAS EXPECTATIVAS  
E....

DIMINUINDO A QUALIDADE E...

AUMENTANDO OS CUSTOS (PARA REPARAR), DIMINUINDO O LUCRO,  
AUMENTANDO OS PRAZOS E...

DETONANDO A COMPETITIVIDADE DA ORGANIZAÇÃO

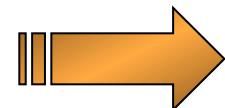


**A QUALIDADE  
É  
IMPORTANTE ?**



## Qualidade está relacionada com FURPS

Funcionalidade (**F**unctionality)  
Usabilidade (**U**sability)  
Confiabilidade (**R**eliability)  
Desempenho (**P**erformance)  
Suportabilidade (**S**upportability)



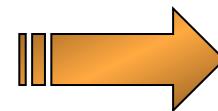
As funcionalidades que o Software deve prover aos Seus usuários para satisfazê-los Nas suas necessidades

**QUALIDADE**



Qualidade está relacionada com FURPS

Funcionalidade (**F**unctionality)  
Usabilidade (**U**sability)  
Confiabilidade (**R**eliability)  
Desempenho (**P**erformance)  
Suportabilidade (**S**upportability)



Estética, Aprendizagem e uso facilitado, consistência da interface e da documentação  
Materiais de treinamento

**QUALIDADE**



Qualidade está relacionada com FURPS

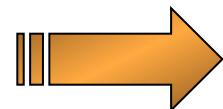
Funcionalidade (**F**unctionality)

Usabilidade (**U**sability)

Confiabilidade (**R**eliability)

Desempenho (**P**erformance)

Suportabilidade (**S**upportability)



Recuperabilidade, previsibilidade  
exatidão.

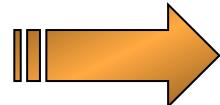
Freqüência e severidade de falhas

**QUALIDADE**



Qualidade está relacionada com FURPS

Funcionalidade (**F**unctionality)  
Usabilidade (**U**sability)  
Confiabilidade (**R**eliability)  
Desempenho (**P**erformance)  
Suportabilidade (**S**upportability)



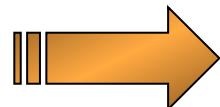
Taxa de transação  
Velocidade de processamento  
Disponibilidade  
Tempo de resposta  
Tempo de recuperação  
Impõem restrições às funções do software

**QUALIDADE**



Qualidade está relacionada com FURPS

Funcionalidade (**F**unctionality)  
Usabilidade (**U**sability)  
Confiabilidade (**R**eliability)  
Desempenho (**P**erformance)  
Suportabilidade (**S**upportability)



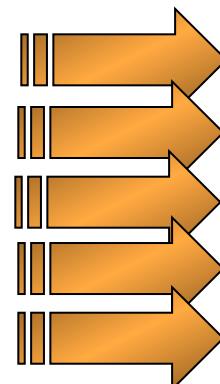
Testabilidade,  
Manutenibilidade  
Sistema atualizado depois da sua  
Implantação → Facilidade de  
Mudanças  
O processo de criação do sistema  
Impondo condições ao desenvolv.

**QUALIDADE**



Qualidade está relacionada com FURPS

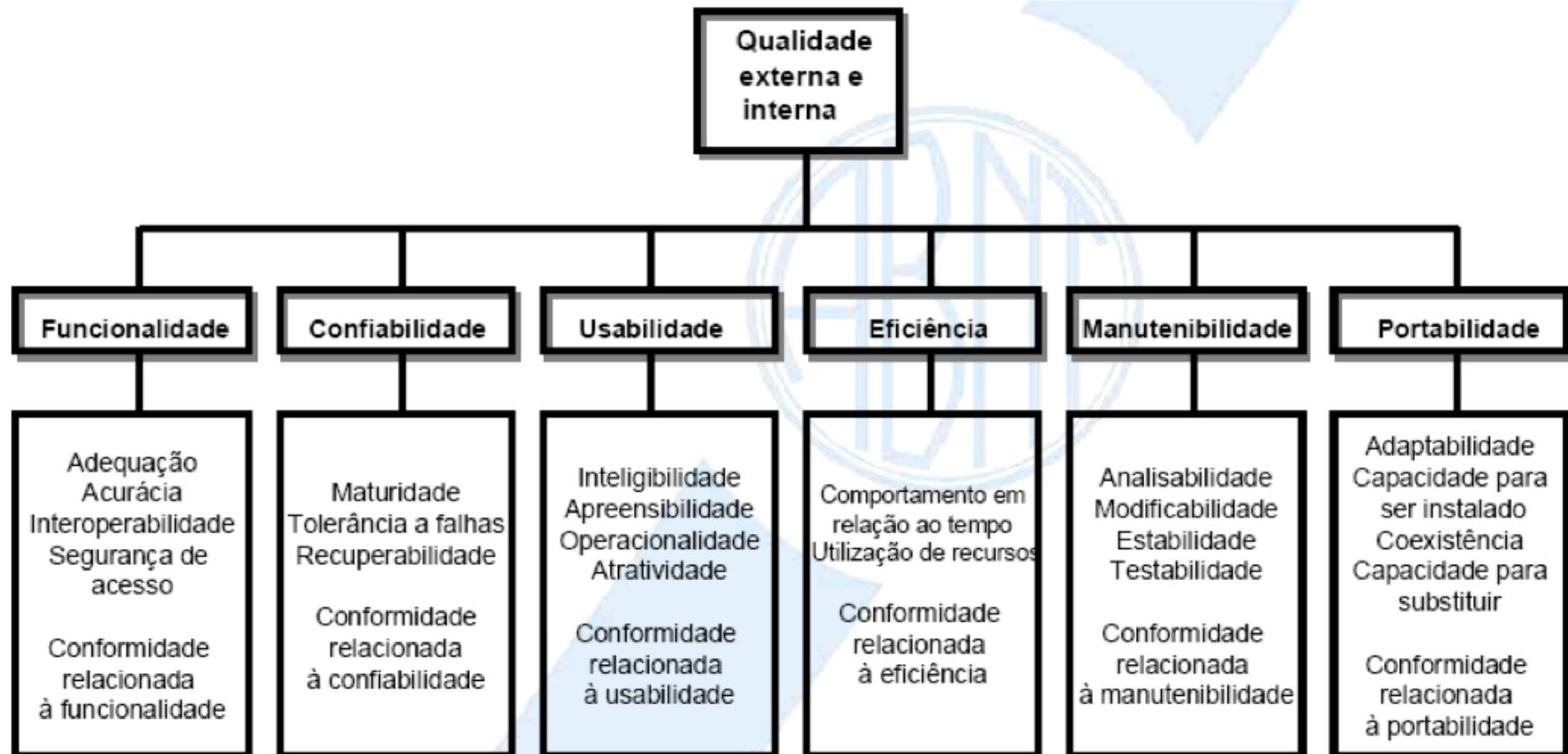
- Funcionalidade (**F**unctionality)
- Usabilidade (**U**sability)
- Confiabilidade (**R**eliability)
- Desempenho (**P**erformance)
- Suportabilidade (**S**upportability)



**QUALIDADE**



**ISO 9126**





Característica	Subcaracterística	Pergunta Chave para a subcaracterística
Funcionalidade (satisfaz as necessidades?)	Adequação	Propõe-se a fazer o que é apropriado?
	Acurácia	Faz se o que foi proposto de forma correta?
	Interoperabilidade	Interage com os sistemas especificados?
	Conformidade	Está de acordo com as normas, leis, etc?
	Segurança de acesso	Evita acesso não autorizado aos dados?



Confiabilidade  (é imune a falhas?)	Maturidade	Com que frequência apresenta falhas?
	Tolerância a falhas	Ocorrendo falhas, como ela reage?
	Recuperabilidade	É capaz de recuperar dados em caso de falha?



Usabilidade (é fácil de usar?)	Inteligibilidade	É fácil aprender o conceito e a aplicação?
	Apreensibilidade	É fácil aprender a usar
	Operacionalidade	É fácil aprender a controlar?



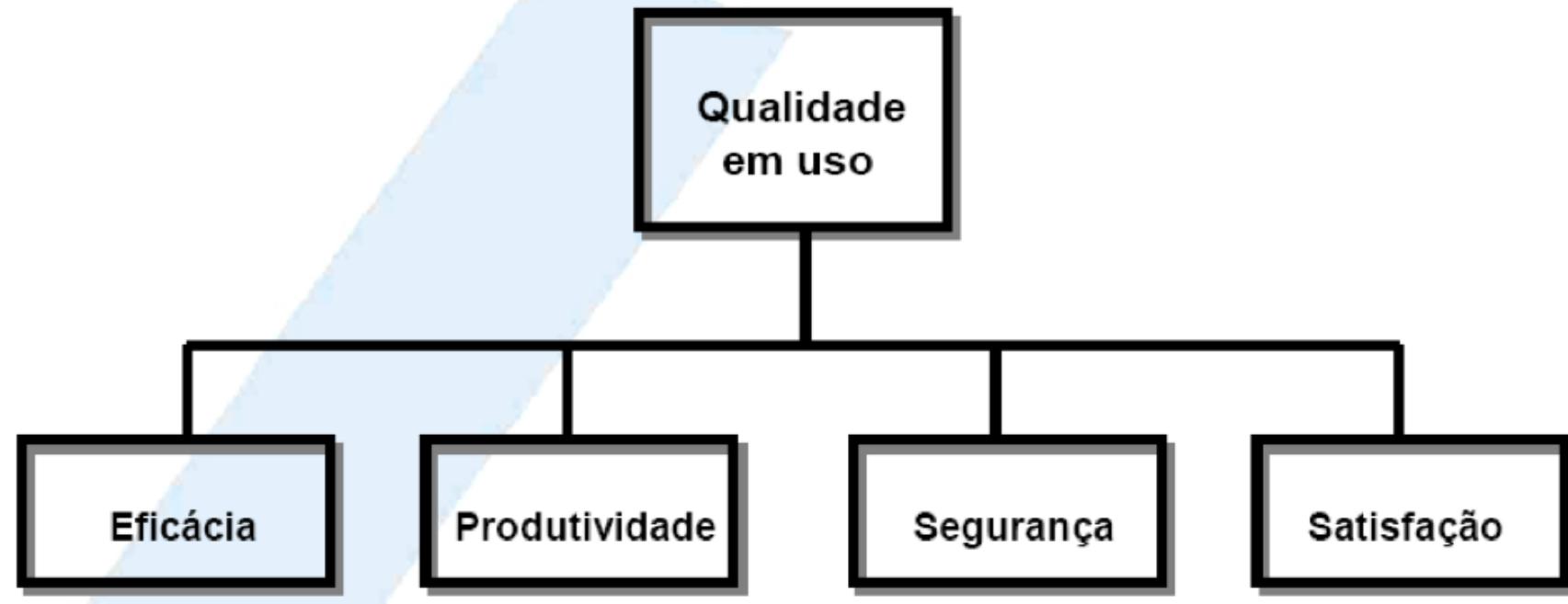
Eficiência (é rápido e enxuto)	Tempo	Qual é o tempo de resposta, a velocidade de execução?
	recursos	Quanto recurso usa? Durante quanto tempo?



Manutenibilidade  (É fácil de modificar)	Analisabilidade	É fácil de encontrar uma falha quando ocorre?
	Modificabilidade	É fácil modificar e adaptar?
	Estabilidade	Há grande risco quando se faz alterações?
	Testabilidade	É fácil testar quando se faz alterações?



Portabilidade (é fácil de usar em outro ambiente?)	Adaptabilidade	É fácil adaptar a outros ambientes
	Capacidade para ser instalado	É fácil instalar em outros ambientes?
	Conformidade	Está de acordo com padrões estabelecidos?
	Capacidade para substituir	É fácil usar para substituir outro?



Capacidade do produto de software de permitir que usuários especificados atinjam metas especificadas com eficácia, produtividade, segurança e satisfação em contextos de uso especificados.



**Eficácia:** Capacidade do produto de software de permitir que usuários atinjam metas especificadas com acurácia e completitude, em um contexto de uso especificado.

**Produtividade:** Capacidade do produto de software de permitir que seus usuários empreguem quantidade apropriada de recursos em relação à eficácia obtida, em um contexto de uso especificado.

**Segurança:** Capacidade do produto de software de apresentar níveis aceitáveis de riscos de danos a pessoas, negócios, software, propriedades ou ao ambiente, em um contexto de uso especificado

**Satisfação:** Capacidade do produto de software de satisfazer usuários, em um contexto de uso especificado

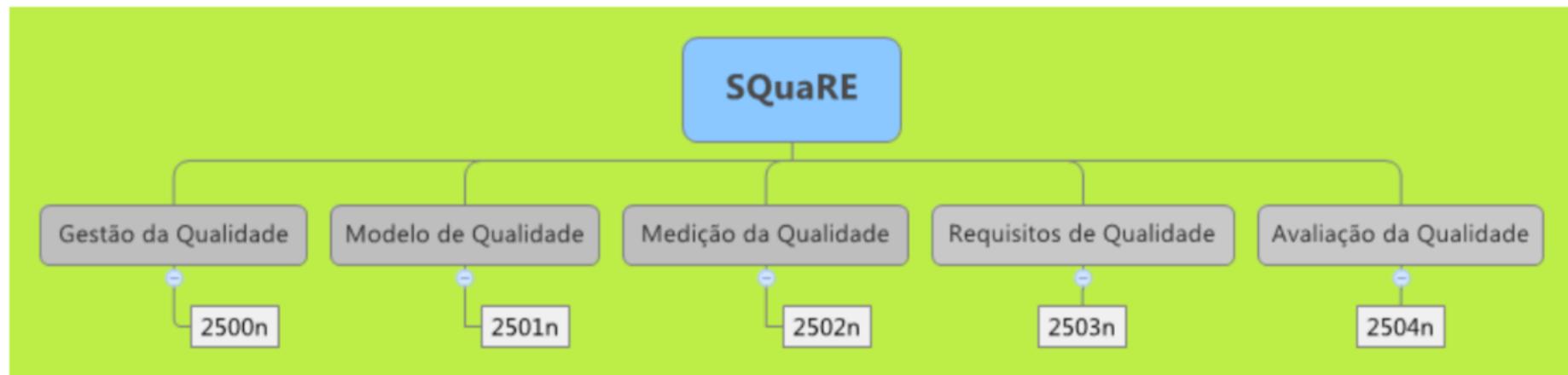


Atualmente a norma ISO/IEC 9126, foi substituída pela norma SQuaRE (Série de normas ISO/IEC 25000) com os seguintes benefícios para a área de desenvolvimento de software:

- Coordenar a orientação sobre a mensuração da qualidade dos produtos de software e avaliação da qualidade de software.
- Oferecer uma orientação para a especificação de requisitos de qualidade do produto de software.
- Harmonizar com a norma ISO/IEC 15939-Qualidade do processo de software ISO/IEC 15939 (2007), sob a forma de produto de Software de Qualidade de medição de referência.



A norma SQuaRE é um conjunto de normas com finalidades específicas.



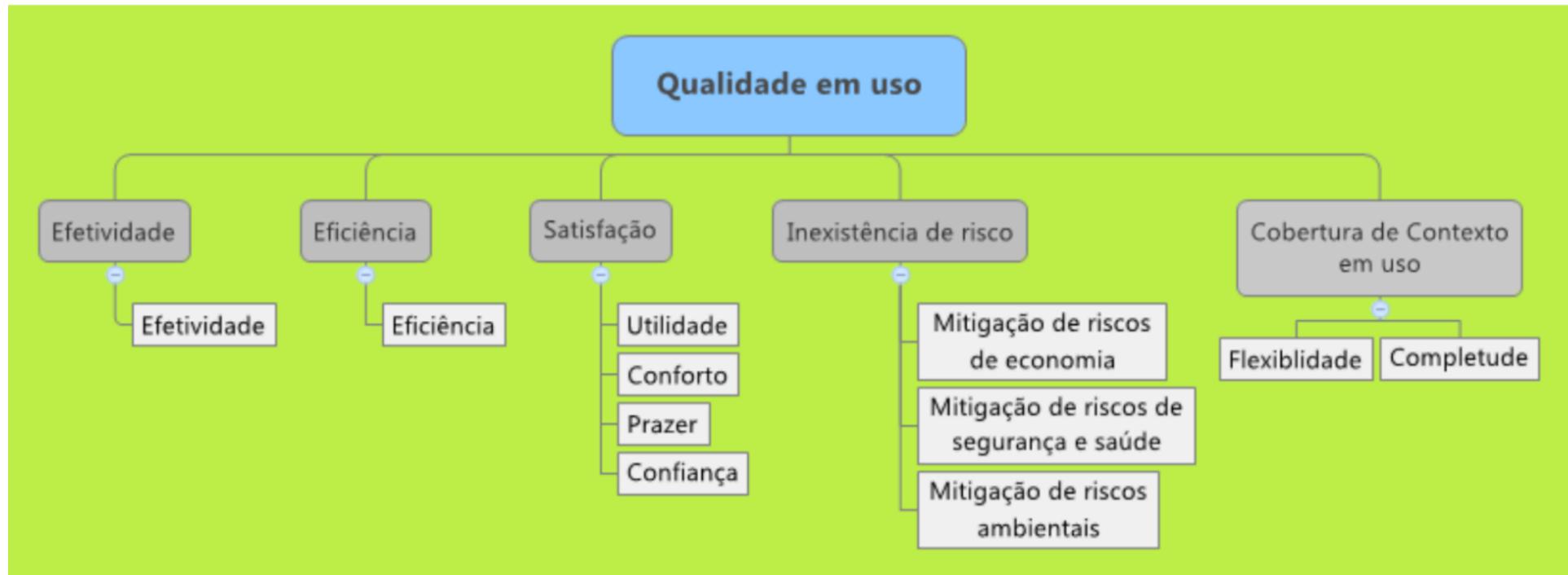


Embora a SQuaRE reconheça a influência do processo no produto, o foco na qualidade do produto foi ressaltado nesta série de normas.





Por serem características de alto impacto na qualidade em uso do produto de software que, por sua vez, impacta na percepção da qualidade que o cliente tem sobre o produto de software adquirido.





Onde:

- **Efetividade:** é a característica que diz respeito a capacidade que o software possui de atender a metas específicas sob condições particulares de uso levando em conta a exatidão e a integridade.
- **Eficiência:** é a característica que diz respeito a capacidade que o software possui de apresentar recursos que foram gastos ao atingir metas específicas sob condições particulares de uso levando em conta a exatidão e a integridade.
- **Satisfação:** é a característica que diz respeito a capacidade que o software possui de agradar seus clientes diante de um contexto de uso específico.
- **Inexistência de risco:** é a característica que diz respeito a capacidade que o software possui de minimizar riscos econômicos, humanos, para a vida humana e ambientais em um diante de um contexto de uso específico.
- **Cobertura de Contexto em Uso:** é a característica que diz respeito a capacidade que o software tem de possuir eficiência, integridade e satisfação do cliente diante de um contexto de uso específico.



## Conclusão:

**Se...**

Software é composto por programas, documentos e dados e tudo isso é passível de erro, que podem se transformar em defeitos e, se não encontrados, em falhas; E se...

Qualidade é satisfazer as expectativas do usuário (cliente que contrata o desenvolvimento de um software) e se esta expectativa pode ser subdividida em várias subcaracterísticas; E se...

É importante verificar e validar para avaliar a qualidade do produto de software...

**Então:....**

Existirão tantos tipos de testes quanto tipos de expectativas que possam existir para os usuários.

**Porém ...**

Todas elas serão, igualmente importantes ao mesmo tempo para um determinado usuário (Cliente contratante do software)

Esse raciocínio direciona os inúmeros tipos de testes de software que existem (além do uso de técnicas estátivas de verificação e validação).



- Ricardo Ajax

[Ricardoajax@unb.br](mailto:Ricardoajax@unb.br)

