Testes de sistema

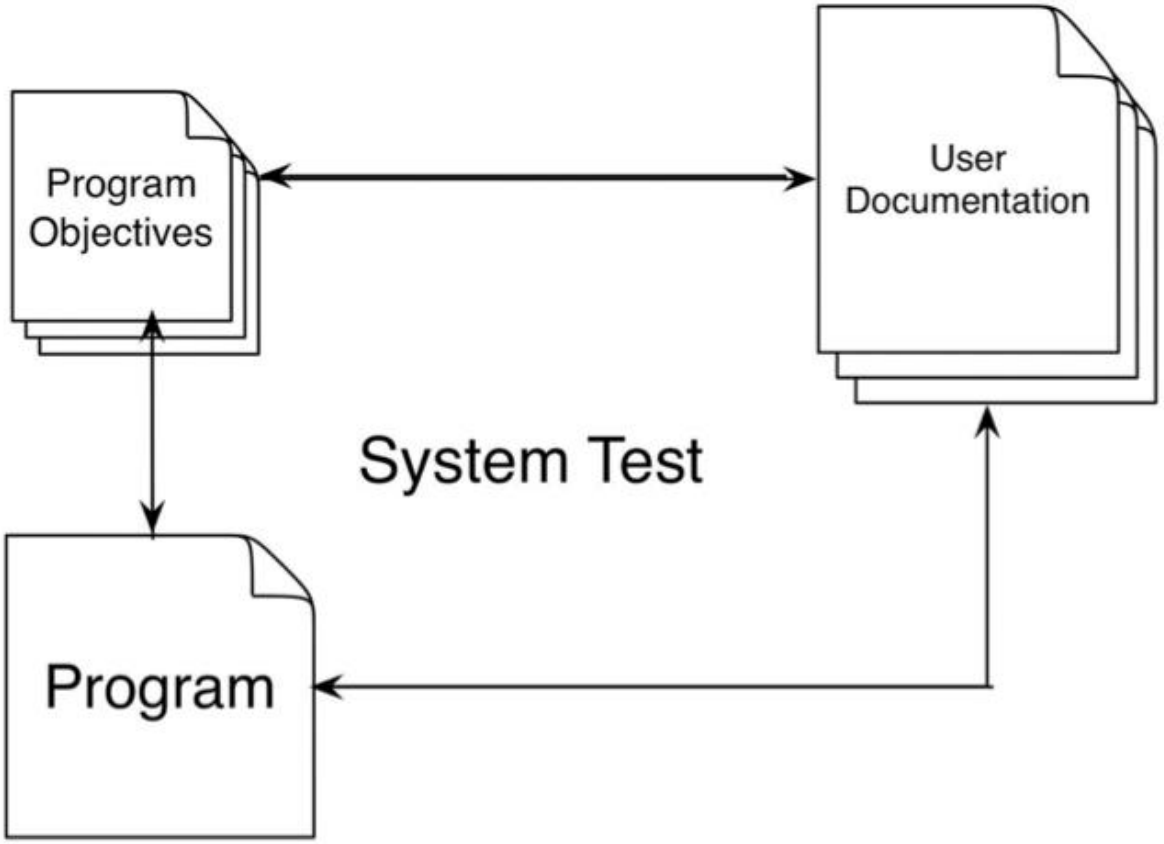
Testes de sistema

* Testamos os módulos individualmente do nosso sistema (testes unitários)
* Após os testes removemos os mocks e stubs e integramos todos os nosso módulos (testes de integração)
* Mas, será que nosso software está pronto para uso?

Objetivos

* Revelar falhas no sistema
* Demonstrar que o sistema em teste atende aos objetivos do projeto
* O sistema foi construído corretamente?

Testes de sistema



Exemplos de teste sistema

* Qualquer parte do sistema deve ser acessível em 3 cliques
* Operações devem ser realizadas em no máximo 2 segundos
* O sistema deve permitir o cadastro de dados de imposto e posteriormente o envio de dados para a receita

Exemplos de teste sistema

* O sistema acima deve suporta um volume de 2 tb de dados
* A transmissão dos dados para a receita está sendo criptografada corretamente?
* Para cada operação realizada o sistema deverá emitir uma resposta

Fontes de informações para testes

* Especificação dos requisitos
* Protótipo, layout, mock-ups
* Políticas da organização (negócio)
* Características de produtos descritos na literatura

Como testar nosso sistema?

Testes	Testes
Teste de carga	Teste de confiança
Teste de stress	Teste de manutenção
Teste de usabilidade	Teste de documentação
Teste de segurança	Testes de procedimento
Teste de performance	Teste de compatibilidade
Teste de armazenamento	Teste de instalação
Teste de configuração	Teste de recuperação

Testes de usabilidade

* Programas devem ser fáceis de serem utilizados
* Ao mesmo tempo que devem se adaptar a usuários mais avançados
* O sistema responde às ações do usuário?

Testes de usabilidade

* As saídas do sistema são fáceis de serem compreendidas?
* Mensagens de erro são fáceis de serem compreendidas?
* Há consistência em minha aplicação?
* São realizadas verificações necessárias?

Testes de usabilidade

* TestR (Teste de usabilidade remoto)
* Mouseflow (Mapa de calor)

Testes de segurança

* Verificar se o software atende aos requisitos de segurança desejados
* É preciso conhecer qual o nível de segurança desejado
* Identificar na literatura como os problemas ocorrem e tentar replicá-los em nossa aplicação

Testes de segurança

* Sonar
* Vega

Testes de armazenamento

* Aplicações precisam armazenar dados
* Este armazenamento está sendo feito corretamente?
  * Quanto de capacidade os dados estão consumindo?
  * E os arquivos temporários?

Testes de instalação

* A instalação de um programa é fundamental para que ele possa ser utilizado
* Não podem haver complicações nesse processo

Teste de confiança

* Nosso programa é confiável?
  * Quanto tempo ele está disponível para uso por nossos clientes?
* Se forem oferecidas metas nosso software deverá atendê-las

Teste de carga

* Nossas aplicações, em geral, serão executadas por dezenas ou até mesmo milhares de pessoas simultaneamente
* Será que estamos preparados? Como garantimos isto?
* Iremos simular uma grande quantidade de requisições ao nosso sistema

Teste de carga

* JMeter

Teste de stress

* Testes de carga simulam uma grande, mas real, quantidade de requisições
* Teste de stress irá avaliar sistema além das condições normais de uso
* Caso o sistema não pare nestas condições temos um bom sinal de estabilidade
* Podemos avaliar também até quais limites nosso sistema suporta

Teste de stress vs teste de carga

Como realizar testes de carga e stress

* Apache Benchmark
* jMeter

Teste de recuperação

* Programas são vulneráveis a diversas falhas
  * Banco de dados
  * Redes
  * Operações falhas
* O objetivo deste teste é verificar se, após uma dessas falhas, o sistema consegue voltar a um estado estável

Testes de compatibilidade

* Há casos onde nosso software irá executar em diferentes ambientes
* Será que ele irá executar bem no Windows? E no Linux ou Mac?
* Se for uma aplicação web será que ela funciona da mesma forma no Internet Explorer, Firefox e no Chrome?

Testes de compatibilidade

* Iremos avaliar como o sistema se comporta em diferentes ambientes
* Testaremos o software em diferentes sistemas operacionais
* Se for um sistema web testaremos o software em diferentes browsers
* Além disso também testaremos o software em diferentes versões de um mesmo browser

Estudo de caso

Teste de documentação

* A documentação do usuário deve ser condizente com o objetivo do software
* Ela deve ser validada para verificar sua acurácia e clareza

Testes funcional

* Iremos avaliar o sistema do ponto de vista do usuário
* O teste é concentrado nos requisitos funcionais do software
* Estamos interessados em encontrar divergências entre os requisitos funcionais e nosso software

Queremos identificar

* Funções incorretas ou ausentes
* Erros de interface
* Erros nas estruturas de dados ou acesso ao banco de dados
* Erros de desempenho
* Erros de inicialização e término

Diferenças entre testes unitários e funcionais

* Testes unitários focam em: **i)** garantir que as funções atendam aos requisitos; **ii)** e também em sua implementação
* Com testes funcionais estamos interessados apenas no ponto **i)**

Etapas do teste funcional

1. Identificar os casos de testes
2. Determinar os valores usados nos casos de testes
3. Executar os casos de teste
4. Avaliar os resultados