Testes de aplicações *Web*

Claudiney Vander Ramos

PUC Minas São Gabriel

Fundamentos de Testes de Software

O que são aplicações *Web*

- Uma aplicação que é executada num servidor web e pode ser acessada através de uma rede, podendo esta ser a *World Wide Web ou uma intranet.*

- Pode ser acessada a partir de qualquer lugar tendo um *browser* como cliente (também chamado de *thin client)*

- Tem a particularidade de se poder realizar atualizações e executar tarefas de manutenção sem a necessidade de distribuição e instalação em, possivelmente, milhares de clientes

Aplicações tradicionais Vs *Web*

Os testes de software são uma tarefa difícil, mas quais as principais diferenças para os testes de uma aplicação *Web*?

- **Interface dinâmica**
  - Personalização, páginas diferentes a cada chamada;
- **Ambiente de funcionamento interativo**
  - Browsers, Scripts, Proxy-servers, BD, etc;
- **Usuários heterogêneos**
  - Sem conhecimentos prévios e comportamento desconhecido;

Testes de Aplicações *Web*

Nas aplicações *Web* podemos definir dois grandes grupos de testes:

- **Testes não funcionais**

  Testes não relacionados com a funcionalidade da aplicação, ou seja, testes relacionados com a performance, usabilidade, segurança...

- **Testes funcionais**

  Testes diretamente ligados à funcionalidade da aplicação, verificam se todas as ações estão sendo executadas corretamente e se o tratamento de erros está sendo feito.

Testes Não Funcionais

- Testes de Desempenho
- Testes de Carga
- Testes de Stress
- Testes de Compatibilidade
- Testes de Usabilidade
- Testes de Acessibilidade
- Testes de Segurança

Testes de Desempenho

- Os testes de desempenho verificam certos parâmetros de desempenho do sistema:
  - Tempos de Resposta;
  - Disponibilidade do serviço;
- São geralmente executados simulando centenas de usuários acessando o sistema num dado intervalo de tempo;
- A informação sobre os acessos é gravada e depois analisada, estimando assim os níveis de carga, para os quais existe um esgotamento dos recursos;
- As falhas descobertas através de testes de desempenho são tipicamente devido a falhas no ambiente de execução:
  - Recursos escassos;
  - Recursos mal distribuídos;
  - Latência na rede;

Ferramentas: WAPT 5.0, RADView (WebLOAD), JMeter

Testes de Desempenho

Ferramentas:

- WAPT 5.0 (Web Application Testing)
  - http://www.loadtestingtool.com/
- RADView
  - http://www.radview.com/
- WebLOAD
  - http://www.webload.org/
- JMeter
  - http://jmeter.apache.org

Testes de Carga

- Os testes de carga diferem dos testes de desempenho na medida em que requerem que o desempenho do sistema seja medido a partir de um nível de carga pré-definido;
- Mede o tempo necessário para executar diversas ações sobre condições pré-definidas, que incluem uma configuração mínima e uma configuração máxima de atividade da aplicação;
- São simulados diversos usuários executando a aplicação, sendo a informação gravada. Sempre que uma ação falhar, são gerados relatórios para reportar o erro e as condições em que esse erro ocorreu.

Ferramentas: RADView, WebPerformance, Mercury LoadRunner

Testes de Carga

Ferramentas:

- RADView
  - http://www.radview.com/
- WebPerformance
  - http://www.webperformanceinc.com/
- HP (Mercury) LoadRunner
  - hhtp://www8.hp.com/br/pt/software-solutions/loadrunner-load-testing/

Testes de stress

- Avalia o comportamento do sistema num estado limite (ou superior) segundo a sua especificação;

- Usados para avaliar a resposta do sistema em picos de atividade que excedem os limites, verificando assim se o sistema apresenta *"crash"* ou se é capaz de se recuperar de tais condições.

Ferramentas: **AdventNet QEngine** , **DTM DB Stress**

Testes de stress

Ferramentas:

- **AdventNet QEngine**
  - http://www.manageengine.com/products/qengine/
- **DTM DB Stress**
  - http://www.sqledit.com/stress/

Testes de compatibilidade

- Determina se a aplicação é executada como esperado num ambiente que contém diferentes condições de hardware e software;

- Devido à enorme combinação de componentes existentes na web não se torna viável que todas elas sejam testadas;

- Usualmente, apenas as combinações mais comuns são verificadas, tendo como conseqüência, que apenas um subconjunto de possíveis falhas seja descoberto.

Ferramentas: **AnyBrowser**

Testes de Usabilidade

- Os testes de usabilidade, são de extrema importância, pois é a melhor forma de verificar se um sistema corresponde ao que é pretendido;
- Apesar de existirem algumas ferramentas que verificam as interfaces segundo um conjunto de normas de usabilidade, estas não são 100% confiáveis;
- Podem ser realizados dois tipos principais de testes de usabilidade:
  - Realizados pelo próprio programador, verificando se uma série de heurísticas de usabilidade são implementadas de forma eficaz:
  - Através de um grupo de usuários (seguindo o perfil definido inicialmente para o usuário típico das aplicações), que segue um guia de ações pré-definido, sendo os seus erros, tempos de resposta, expressões e comentários escritos e muitas vezes gravados em vídeo;

Ferramentas: WebXAct

Site com diversas ferramentas:www.usefulusability.com/24-usability-testing-tools/

Teste de Acessibilidade

- Verificar se um web site segue um conjunto de normas definidas como as melhores práticas de acessibilidade para um grupo definido de usuários (por exemplo, pessoas com deficiências auditivas, visuais, motoras, cognitivas…);

- Testes normalmente feitos manualmente, seguindo guias do tipo *check-point list,* podendo-se verificar a implementação destas *best practices;*

- Existem ferramentas no mercado que podem automatizar este tipo de testes, mas não sendo estas 100% confiáveis, não dispensam uma avaliação manual seguindo as *guidelines* e as *check-point lists.*

Ferramentas: t.a.w, WAVE

Teste de Acessibilidade

Ferramentas:

- t.a.w
  - http://www.tawdis.net/
- WAVE
  - http://wave.webaim.org/

- Página com diversas ferramentas de teste de usabilidade:
  - www.w3.org/WAI/RC/tools/complete

Testes de segurança

- Avalia a eficácia de resposta da aplicação contra usuários não autorizados ou acessos indesejados;
- Avalia a capacidade global da aplicação de permitir aos usuários autorizados o acesso aos serviços e recursos disponibilizados pela mesma;
- Também podemos enquadrar neste grupo os testes de invasão
  - Sql Injections;
  - Key Loggers;
  - ...
- Compromisso entre **proteção** vs **usabilidade**
  - Tornar um sistema demasiado seguro em detrimento da sua usabilidade, i.e., não tem interesse ter um sistema muito seguro, mas muito pouco usável;
  - A implementação de um sistema muito usável também não é viável quando esta decisão põe em risco a integridade e segurança do sistema.

Ferramentas: AlertSite, Snort, WinPCap

Testes funcionais

- Verificam se as funcionalidades e os comportamentos operacionais da aplicação cumprem os requisitos;

- Procuram falhas na aplicação geradas por defeitos nos requisitos funcionais da mesma, em vez de falhas devido ao ambiente de execução;

Tipos de Testes Funcionais

Os testes funcionais podem ser agrupados em três grandes grupos:

- Testes de Unidade: Verificam cada componente da aplicação de forma individual;
- Testes de integração: considera partes combinadas da aplicação e verifica a forma como elas comunicam ou funcionam em paralelo;
- Testes de sistema: Procura defeitos que são propriedades do sistema como um todo, e não os defeitos de componentes individuais.

Testes de Unidade

- Uma pequena porção da aplicação é testada de cada vez;

- Necessitam de ser executados no lado do servidor e no lado do cliente:

  - O componente cliente é aquele que contém a interface gráfica(links, texto, imagens,navegação...). Por vezes também inclui validação de entrada e funções simples;

  - O componente servidor é aquele que implementa a lógica de negócio da aplicação, coordenando a execução dos diferentes pedidos e controlando o armazenamento e seleção de informação para um "centro de dados" (BD, XML,TXT...).

Ferramentas: JWebUnit , selenium, JsUnit, NUnit

Testes de integração

- Testa um conjunto de páginas da aplicação, verificando como elas funcionam em conjunto;
- A documentação de desenho da aplicação, mostrando as relações entre as páginas, pode ser usada para definir grupos de páginas que podem ser testadas em conjunto;
- Podemos por exemplo considerar páginas ligadas entre elas, como uma unidade a ser testada;
- A este nível, tanto o comportamento como a estrutura da aplicação devem ser consideradas.

Ferramentas: Selenium

Testes de sistema

- Tenta encontrar defeitos relacionados com toda a aplicação;

- Testes de cobertura devem incluir:
  - Cobertura de funções e casos de uso do usuário;
  - Cobertura de páginas da aplicação (tanto cliente como servidor);
  - Cobertura de links.

Estratégias de teste

- Define a aproximação para o desenho dos casos de teste;
- Possibilidade de testes baseados em metodologias Capture/Replay;
- Três estratégias conhecidas:
  - Testes de caixa preta;
  - Testes de caixa branca;
  - Testes de caixa cinza;

Testes de caixa preta

- Também conhecidos como testes funcionais. Verificam os *outputs* esperados dado um conjunto de *inputs* pré-definidos;
- Os testes são realizados do ponto de vista do usuário, tendo como objetivo verificar se o comportamento da aplicação está correto (podemos considerar os testes de interface como testes de caixa preta);
- A cobertura de testes aplica-se ao teste com diferentes combinações de *inputs*, verificando depois se os *outputs* gerados são aceitáveis;
- Não examina o código fonte da aplicação.

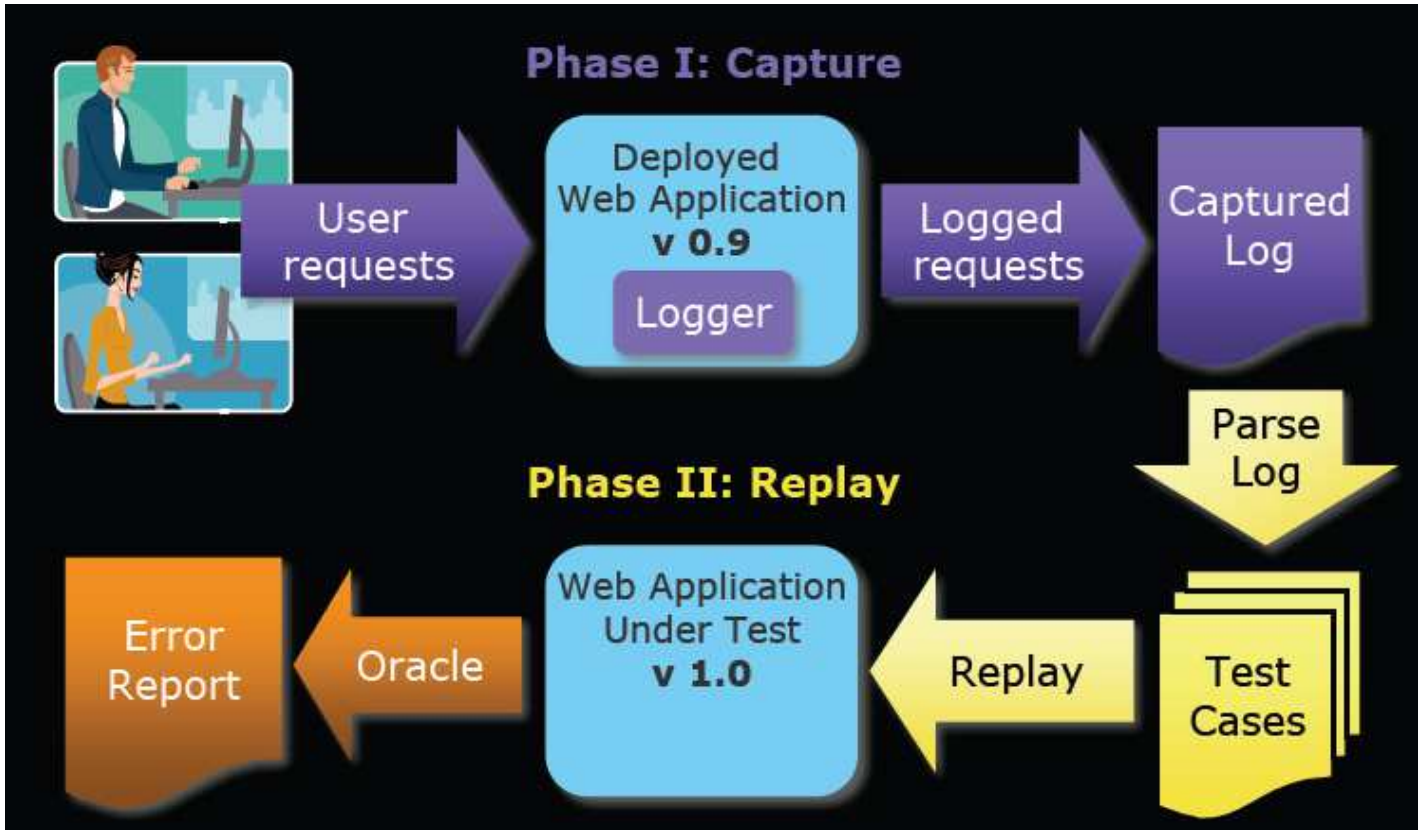
Testes de caixa preta
Técnicas de capture/replay

- Mais propriamente testes de interface com o usuário;

- Vantagens:
  - Reproduz falhas de *input* dos usuários;
  - Verifica configurações de atualizações do *software;*
  - Complementar as outras técnicas de teste;

- Benefícios para as aplicações web:
  - Baratos;
  - Portáveis (independentes da tecnologia);

Testes de caixa preta
Estratégia capture/replay



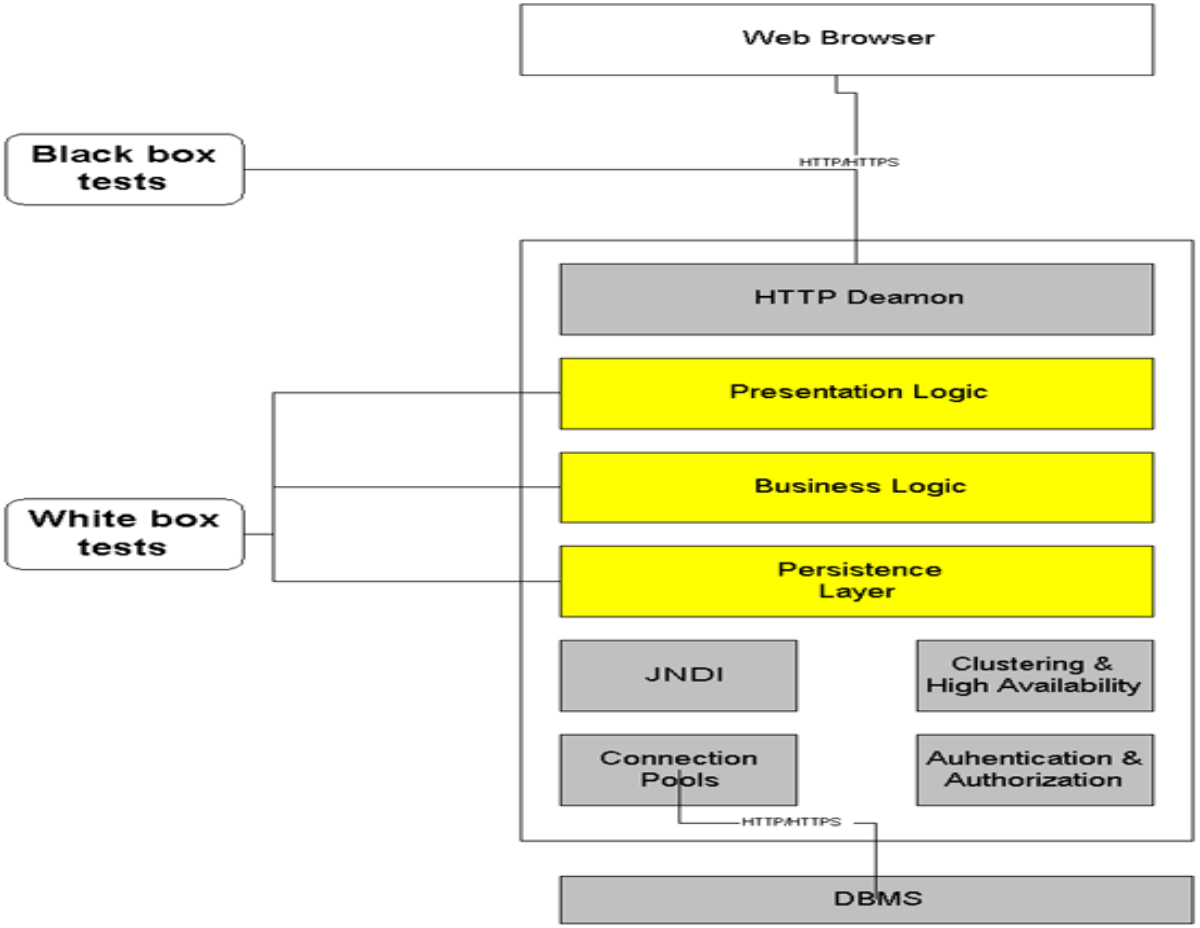
Teste de caixa branca

- Baseados num critério que leva em conta a estrutura da aplicação ou dos seus componentes;
- Modelos representando a aplicação ou a estrutura dos seus componentes, critérios de cobertura e casos de teste são apropriadamente definidos;
- O critério de cobertura deve focar nos *hiperlinks* (implementam a navegação na aplicação), bem como nos componentes internos da aplicação(*code statements).*

Testes de caixa cinza

- É uma mistura dos testes de caixa preta e caixa branca;
- Considera tanto o comportamento da aplicação do ponto de vista de um usuário (como os testes de caixa preta), bem como a estrutura interna da aplicação(como os testes de caixa branca);
- Existe um conhecimento limitado do funcionamento interno da aplicação. Os *testers* que realizarão estes testes têm acesso a documentos de design detalhados com informação além do documento de requisitos (Arquitetura do sistema, definições formais…).

Enquadramento das estratégias na arquitetura web



Bibliografia

- **Documentos**
  - Testing Web Applications, Adithya N.
  - Bypass Testing of Web Applications, Information and Software Engineering,George Mason University
  - A Case Study of Automatically Creating Test Suites from Web Application Field Data, University of Delaware
  - Teste de Aplicações Web, Manuel Costa
  - WebSiteTesting, Edward Miller

- **www**
  - http://www.onjava.com/pub/a/onjava/2003/05/07/blackboxwebtest.html
  - http://www.softwareqatest.com/qatweb1.html
  - http://www.logigear.com/training/course_catalog/course.asp?courseId=4
  - http://www.securityfocus.com/infocus/1704
  - http://www.pantos.org/atw/testing.html
  - http://www.bls.gov/ore/htm_papers/st960150.htm