


Laboratório de Engenharia de Software

## Teste Funcional 2

Arndt von Staa  
Departamento de Informática  
PUC-Rio  
Março 2015

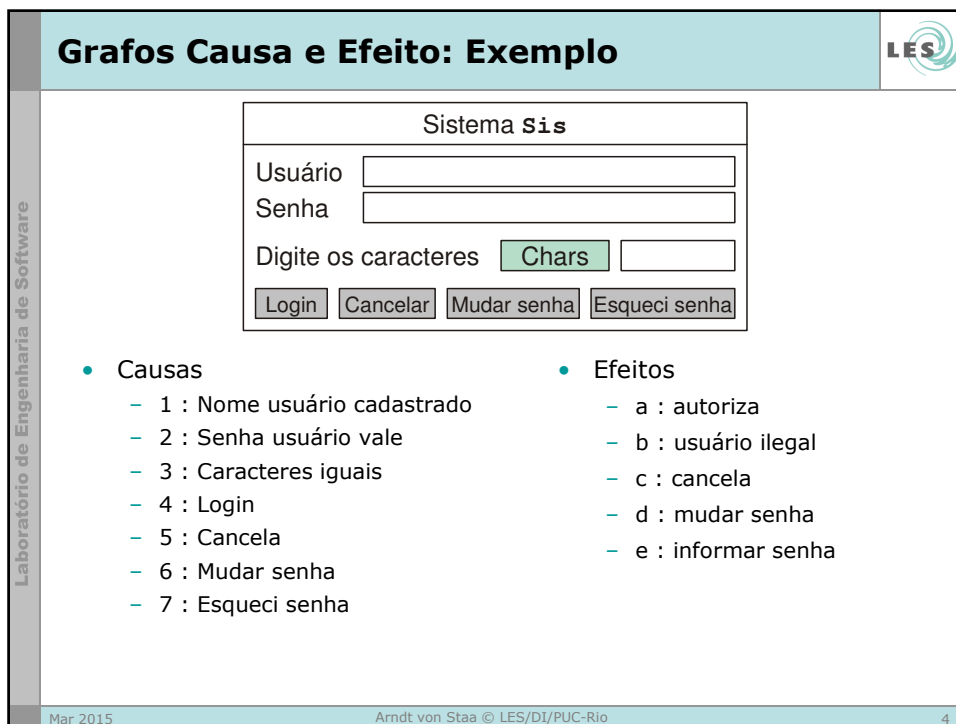
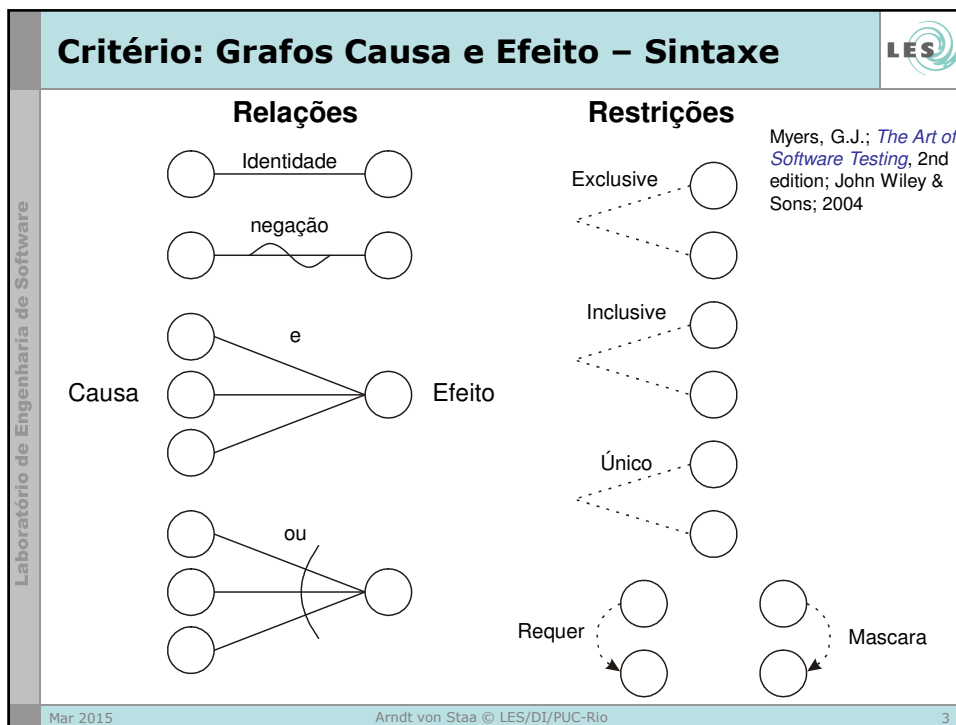
## Especificação

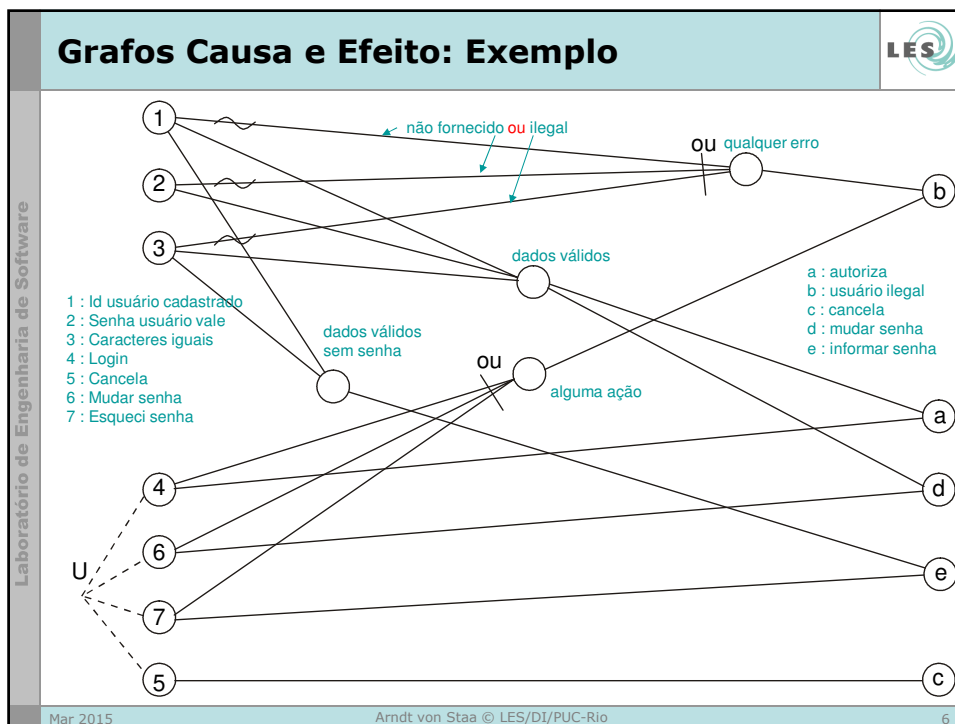
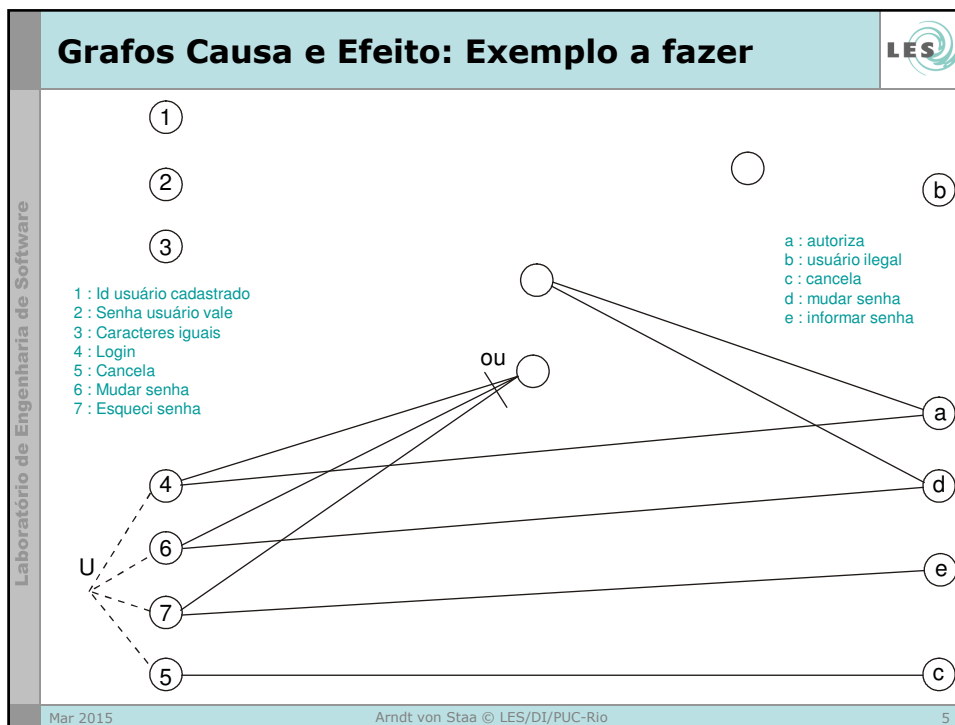


Laboratório de Engenharia de Software

- Objetivo desse módulo
  - apresentar as técnicas de teste funcional: grafos causa e efeito, classes de equivalência, gramáticas regulares, listas de controle
- Justificativa
  - testes funcionais são testes caixa fechada utilizados para verificar a existência de inadequações com relação às especificações de alto nível
- Material de leitura
  - Myers, G.J.; *The Art of Software Testing*, 2nd edition; Hoboken, New Jersey: John Wiley & Sons; 2004

Mar 2015Arndt von Staa © LES/DI/PUC-Rio2





Laboratório de Engenharia de Software

## Grafos Causa e Efeito: Tabela decisão

	1	2	3	4	5	6	7	8	9	10	11	12	13
Usuário	-	-	-	-	s	s	s	s	s	n	n	n	-
Senha	-	-	-	-	s	s	-	n	n	-	-	-	-
Captcha	-	n	n	n	s	s	s	s	s	s	s	s	-
Cancela	s	n	n	n	n	n	n	n	n	n	n	n	n
Login	-	s	n	n	s	n	n	s	n	s	n	n	n
Muda	-	-	s	n	-	s	n	-	s	-	s	n	n
Esqueci	-	-	-	s	-	-	s	-	-	-	-	s	n
Autoriza					x								
Illegal		x	x	x				x	x	x	x	x	
Cancela	x												
Muda						x							
Informa							x						
Impossível													x
	64	16	8	4	4	2	2	4	2	8	4	2	8

=128

↑  
Porque esta coluna?

Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

7

Laboratório de Engenharia de Software

## Critério: partição em classes de equivalência

- Este critério parte do pressuposto que os programas são **desenvolvidos de forma uniforme**
- Pressupostos (hipóteses assumidas como verdadeiras. Crenças?)
  - cada decisão é tratada por um único fragmento de código
    - não existe duplicação de fragmentos de código
  - se um valor, que satisfaz uma determinada condição, leva a uma falha, outros valores satisfazendo essa mesma condição também detectarão essa mesma falha

Myers, G.J.; *The Art of Software Testing*, 2nd edition; Hoboken, New Jersey: John Wiley & Sons; 2004

Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

8

## Partição em classes de equivalência



- Identifique todas as condições dos dados de entrada descritas na especificação
- Crie uma tabela em que cada linha é uma condição e as colunas indicam
  - valores válidos
  - valores não válidos

Condição	Vale	Não vale

Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

9

## Partição em classes de equivalência



- Verifique se existem condições compostas ligadas por operadores lógicos, ex. *and* e *or*
  - decomponha a condição composta em um conjunto de condições elementares possivelmente mutuamente exclusivas
    - $1 \leq t \ \&\& \ t \leq 32$ , resulta nas condições elementares
      - $1 < t$
      - $1 == t$
      - $t > 1$
      - $t < 32$
      - $t == 32$
      - $t > 32$

Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

10

## Partição em classes de equivalência



- Verifique se existem casos de teste ambíguos
  - é ambíguo quando o resultado **não permite discernir** entre a ocorrência ou não de uma ou mais condições
    - ocorreu resultado x mas não sei o que o causou
  - decomponha cada caso de teste ambíguo em diversos outros casos de teste
    - ou reformule o conjunto

## Partição em classes de equivalência



- Verifique se existe algum caso de teste avaliando **condições mascaradas**
  - uma condição *A* mascara outra condição *B* quando a condição *A* torna impossível determinar
    - se a condição *B* ocorreu ou não
      - ocorreu *A* mas não sei dizer se *B* ocorreu ou não
    - ou se a condição *B* depende de *A*
      - ocorreu *A* então também ocorreu *B*
  - decomponha este caso de teste em diversos outros casos de teste
    - ou reformule o conjunto

## Partição em classes de equivalência



- Crie um conjunto de casos de teste valorado
  - ajuste os casos de teste ao critério de valoração, se necessário criando mais casos de teste
  - no conjunto de casos de teste cada caso exercita pelo menos uma condição não exercitada nos demais casos de teste do conjunto

Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

13

## Classes de equivalência: exemplo



Sistema Sis

Usuário

Senha

Digite os caracteres 

Chars

Login

Cancelar

Mudar senha

Esqueci senha

Tabela para o caso Login

Condição	Vale	Não vale
Usuário	Cadastrado 1	Não cadastrado 2
Senha	Corresponde 3	Não corresponde 4
Caracteres	Iguais 5	Não iguais 6

pode ser incrementado com: não fornecido


Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

14

Laboratório de Engenharia de Software

## Classes de equivalência: exemplo



- Usuário legal: a: 1, 3, 5
- Usuário ilegal:
  - b: 2, 5 id errada torna impossível determinar se reconhece senha errada
    - id errada, caracteres válidos
  - c: 1, 4, 5 id errada e/ou senha errada torna impossível determinar se reconhece caracteres não válidos
    - id válida, senha errada, caracteres válidos
  - d: 1, 3, 6
    - id válida, senha válida, caracteres não válidos


Condição	Vale	Não vale
Usuário	Cadastrado <span style="color: red;">1</span>	Não cadastrado <span style="color: red;">2</span>
Senha	Corresponde <span style="color: red;">3</span>	Não corresponde <span style="color: red;">4</span>
Caracteres	Iguais <span style="color: red;">5</span>	Não iguais <span style="color: red;">6</span>

valores errados mascaram mutuamente um ao outro, portanto cada condição de erro precisa ser testada individualmente

Mar 2015
Arndt von Staa © LES/DI/PUC-Rio
15

Laboratório de Engenharia de Software

## Classes de equivalência: exemplo



Como valorar?

- idUsuario
  - cadastrado
  - não cadastrado
    - valor que não existe,
    - prefixo de valor que existe
    - extensão de valor que existe
- senha
  - idem
- caracteres
  - idem
  - mas esses são gerados a cada vez
    - como automatizar o fornecimento?

Mar 2015
Arndt von Staa © LES/DI/PUC-Rio
16



## Gramáticas regulares



- Um conjunto de elementos é uma gramática regular
  - $\langle k \rangle ::= \{ a, b, c \}$
- A concatenação de gramáticas regulares é uma gramática regular
  - $\langle k \rangle ::= \langle g \rangle \langle h \rangle \langle i \rangle$
- A seleção envolvendo gramáticas regulares é uma gramática regular
  - $\langle k \rangle ::= ( \langle g \rangle \mid \langle h \rangle \mid \langle i \rangle )$
- A repetição envolvendo gramáticas regulares é uma gramática regular
  - $\langle k \rangle ::= \overbrace{n_1 - n_2 [ \langle g \rangle ]}^{\text{eu prefiro esta notação}} \text{ ou } \langle k \rangle ::= \langle g \rangle^* ; \langle k \rangle ::= \langle g \rangle^+ ;$
- A recursão à direita é uma gramática regular
  - $\langle k \rangle ::= ( \langle g \rangle \mid \langle h \rangle \langle k \rangle )$

Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

17

## Gramáticas regulares



- Cada *widget* é um conjunto de elementos
- A natureza do conjunto depende da classe do *widget*
  - campos de dados podem valer ou não
  - coletâneas de botões podem assumir exatamente uma das seleções
  - *radio buttons* podem assumir exatamente um dos valores
  - *check boxes* podem assumir zero ou mais dos valores
  - barras de rolagem podem assumir uma das ações
  - ...

Adaptado do teste de caminhos baseado em expressões regulares [Staa, 2000]

Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

18

## Gramáticas regulares



- Cada campo de dados: um elemento
  - valores de campos podem ser legais ou ilegais
- Cada seletor mutuamente exclusivo: uma seleção

Sistema **Sis**

Usuário

Senha

Digite os caracteres

Chars

Login
Cancelar
Mudar senha
Esqueci senha

Gramática

Usuário Senha Captcha ( Login | Cancelar | Mudar | Esqueci )

Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

19

## Gramáticas regulares



- Geração dos dados
 

Usuário Senha Captcha ( Login | Cancelar | Mudar | Esqueci )

Usuário legal	Senha legal	Captcha legal	Login	→ autoriza
Usuário legal	Senha legal	Captcha legal	Cancelar	→ cancela
Usuário legal	Senha legal	Captcha legal	Mudar	→ muda
Usuário legal	Senha legal	Captcha legal	Esqueci	→ esqueci
Usuário <b>ilegal</b>	Senha legal	Captcha legal	Login	→ erro
Usuário <b>ilegal</b>	Senha legal	Captcha legal	Cancelar	→ cancela
Usuário <b>ilegal</b>	Senha legal	Captcha legal	Mudar	→ erro
Usuário <b>ilegal</b>	Senha legal	Captcha legal	Esqueci	→ erro
Usuário legal	Senha <b>ilegal</b>	Captcha legal	Login	→ erro
Usuário legal	Senha <b>ilegal</b>	Captcha legal	Cancelar	→ cancela
Usuário legal	Senha <b>ilegal</b>	Captcha legal	Mudar	→ erro
Usuário legal	Senha <b>ilegal</b>	Captcha legal	Esqueci	→ <del>esqueci</del>
Usuário legal	Senha legal	Captcha <b>ilegal</b>	Login	→ erro
Usuário legal	Senha legal	Captcha <b>ilegal</b>	Cancelar	→ cancela
Usuário legal	Senha legal	Captcha <b>ilegal</b>	Mudar	→ erro
Usuário legal	Senha legal	Captcha <b>ilegal</b>	Esqueci	→ erro
...				


Mar 2015

Arndt von Staa © LES/DI/PUC-Rio

20

Laboratório de Engenharia de Software

## Critério: Lista de quesitos



- Lista de condições de especiais, ex.
  - falta de energia
  - falta de memória
  - falta de espaço em disco
  - erro de leitura
  - erro de transmissão de dados
  - ...
- Como simular essas condições?
  - objetos de imitação (*mock objects*)
  - módulos de simulação, módulos dublê
 

} tratado em aula futura


    - simulam o funcionamento e as falhas de funcionamento
    - realizam operações como se fossem os objetos de produção

Hunt, A.; Thomas, D.; *Pragmatic Unit Test: in Java with JUnit*; Raleigh, North Carolina: The Pragmatic Bookshelf; 2004

Mar 2015
Arndt von Staa © LES/DI/PUC-Rio
21

Laboratório de Engenharia de Software

## Quesitos para aplicações Web




É um teste baseado em listas de quesitos [Splaine & Jaskiel , 2001]

- Verifique a corretude dos instaladores
  - todas as variáveis de ambientes e *registry* estão corretamente inicializadas → melhor: evite o uso do registry
  - sempre examinar para cada variável de ambiente ou condição do ambiente se contém valor válido (ex. ODBC)
- Verifique se as mensagens de erro informam corretamente o problema observado
  - devem ser evitadas mensagens do gênero
    - Ox81234 – procure o gerente da aplicação
    - idMaquina errado
    - mensagem: CPF não possui Lattes, ou candidato não possui formação necessária
      - que erros podem ter ocorrido?
      - emitida pela Plataforma Carlos Chagas do CNPq

Mar 2015
Arndt von Staa © LES/DI/PUC-Rio
22

Laboratório de Engenharia de Software

## Quesitos para aplicações Web




- Verifique se o sistema operacional cliente (versão e *service packs*) é consistente com a aplicação
- Verifique se a versão do *browser* instalada é consistente com a aplicação
- Verifique se os *plugins* do *browser* requeridos estão instalados em versão suportada
  - JavaScript, Java Applets, Flash, Lua, ...
- ...

Mar 2015
Arndt von Staa © LES/DI/PUC-Rio
23

Laboratório de Engenharia de Software

## Quesitos para aplicações Web




- Verifique a sensibilidade a parâmetros de uso do *browser*
- Verifique o comportamento com vários *browsers*
- Verifique o comportamento com várias versões de um *browser*
- Verifique o comportamento com vários tipos de periféricos
- Verifique se a aplicação observa se todos os servidores estão operando
- Verifique se a aplicação observa se todos os serviços requeridos estão operando

Vários *browsers*, várias versões de um *browser* – como testar isso?

- uso de “máquinas virtuais” – simuladores de *browsers* e de suas versões
- para windows existe a ferramenta vmware que permite criar uma variedade de máquinas virtuais cada uma com o seu sistema operacional e versões de software, todas rodando em uma mesma máquina

Mar 2015
Arndt von Staa © LES/DI/PUC-Rio
24

**Quesitos para aplicações Web**



Laboratório de Engenharia de Software

- Verifique os privilégios de acesso
- Verifique completeza e corretude da versão de componentes, ex. DLL, *dynamic objects*, tabelas XML
- Verifique correto registro de componentes (COM, Java)
  - *plugins* usados
- Verifique a corretude do DNS
- Verifique a adequação da proteção estabelecida pelo firewall
- Verifique problemas causados por linhas de transmissão lentas
  - *time out*

Mar 2015
Arndt von Staa © LES/DI/PUC-Rio
25

**Quesitos para aplicações Web**



Laboratório de Engenharia de Software

- Verifique ...
  - a lista do livro [Splaine e Jaskiel, 2001] se estende por centenas de páginas

Mar 2015
Arndt von Staa © LES/DI/PUC-Rio
26

## Referências bibliográficas



- Delamaro, M.E.; Maldonado, J.C.; Jino, M.; *Introdução ao Teste de Software*; Rio de Janeiro, RJ: Elsevier / Campus; 2007
- Myers, G.J.; *The Art of Software Testing*, 2nd edition; Hoboken, New Jersey: John Wiley & Sons; 2004
- Nguyen, H.Q.; *Testing Applications on the Web: Test Planning for Internet-Based Systems*; New York: John Wiley & Sons; 2001
- Nguyen, H.Q.; "Testing Web-based Applications"; *Software Testing and Quality Engineering*; New York: John Wiley & Sons; 2000; pages 23-29
- Splaine, S.; Jaskiel, S.P.; *The Web Testing Handbook*; Orange Park, FA: STQE Publishing; 2001



FIM