




Laboratório de Engenharia de Software

Teste Funcional 2

Arndt von Staa
Departamento de Informática
PUC-Rio
Março 2017

Especificação



Laboratório de Engenharia de Software

- Objetivo desse módulo
 - apresentar as técnicas de teste funcional: classes de equivalência, gramáticas regulares, e 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 2017Arndt von Staa © LES/DI/PUC-Rio2

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?*)
 - se um caso de teste que satisfaz um determinado conjunto de condições leva a uma falha, outras escolhas de valores satisfazendo esse mesmo conjunto detectarão essa mesma falha
 - processamentos não determinísticos violam essa restrição
 - não existe duplicação de fragmentos de código
 - condições de um caso de teste
 - caso de teste abstrato
 - mais critério de valoração selecionado

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

Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

3

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 2017

Arndt von Staa © LES/DI/PUC-Rio

4

Laboratório de Engenharia de Software

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 2017
Arndt von Staa © LES/DI/PUC-Rio
5

Laboratório de Engenharia de Software

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
 - é comum para casos de teste de verificadores de dados
 - se os dados contêm dois ou mais erros, qual erro foi observado?
 - decomponha cada caso de teste ambíguo em diversos outros casos de teste
 - ou reformule o conjunto

Mar 2017
Arndt von Staa © LES/DI/PUC-Rio
6

Laboratório de Engenharia de Software

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 ocorrerá *B*
 - decomponha o caso de teste em diversos outros casos de teste
 - ou reformule o conjunto


Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

7

Laboratório de Engenharia de Software

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

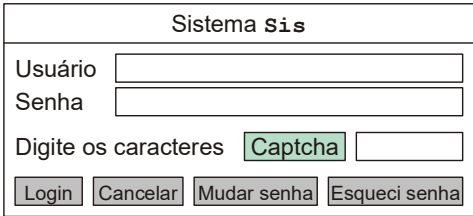
Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

8

Laboratório de Engenharia de Software

Classes de equivalência: exemplo






Tabela para o caso Login


| Condição | Vale | Não vale |
|----------|---------------|-------------------|
| Usuário | Cadastrado 1 | Não cadastrado 2 |
| Senha | Corresponde 3 | Não corresponde 4 |
| Captcha | Igual 5 | Não igual 6 |

pode ser incrementado com: *não fornecido*

Mar 2017
Arndt von Staa © LES/DI/PUC-Rio
9

Laboratório de Engenharia de Software

Classes de equivalência: exemplo



- Usuário correto: a: 1, 3, 5
- Usuário incorreto:
 - b: 2, 3, 5
 - id errada, senha a que corresponderia ao usuário correto, captcha válido
 - c: 1, 4, 5
 - id válida, senha errada, captcha válido
 - d: 1, 3, 6
 - id válida, senha válida, captcha não válido

id errada torna impossível determinar se reconhece senha errada

| Condição | Vale | Não vale |
|----------|---------------|-------------------|
| Usuário | Cadastrado 1 | Não cadastrado 2 |
| Senha | Corresponde 3 | Não corresponde 4 |
| Captcha | Igual 5 | Não igual 6 |

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

Mar 2017
Arndt von Staa © LES/DI/PUC-Rio
10

Laboratório de Engenharia de Software

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 2017

Arndt von Staa © LES/DI/PUC-Rio

11

Laboratório de Engenharia de Software

Classes de equivalência: exemplo

Como valorar?

- idUsuario
 - cadastrado
 - não cadastrado
 - valor que não existe,
 - prefixo de valor que existe – tem n chars usar n-1, início e final
 - extensão de valor que existe – tem n chars usar n+1, início e final
- senha
 - idem
- captcha
 - idem
 - mas esses são gerados a cada vez
 - como automatizar o fornecimento?


Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

12

Laboratório de Engenharia de Software

Critério lista de quesitos



- Esse critério baseia-se em um documento contendo uma lista dos quesitos que a organização considera relevantes
- Quesitos são adicionados e eliminados à medida que a organização vai identificando o que é relevante
- Alguns quesitos são sempre relevantes:
 - critério de valoração
- Outros dependem do tipo de software que é desenvolvido


Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

13

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
 - consumo de energia (ex. celulares)
 - . . .


Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

14

Laboratório de Engenharia de Software

Critério: Lista de quesitos



- 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 2017

Arndt von Staa © LES/DI/PUC-Rio

15

Laboratório de Engenharia de Software

Quesitos para aplicações Web



- Verifique a corretude dos instaladores
 - todas as variáveis de ambientes e *registry* estão corretamente inicializadas
 - para cada variável de ambiente ou condição do ambiente sempre examinar 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? No caso, CPF incorreto
 - emitida pela Plataforma Carlos Chagas do CNPq

Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

16


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

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 2017
Arndt von Staa © LES/DI/PUC-Rio
17

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 2017
Arndt von Staa © LES/DI/PUC-Rio
18

Laboratório de Engenharia de Software

Quesitos para aplicações Web



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


Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

19

Laboratório de Engenharia de Software

Gramáticas



- Gramáticas são usualmente usadas para verificar se sentenças são válidas
 - sentenças são fragmentos de texto
 - português: análise sintática
 - linguagens de programação
- No entanto, podem ser usadas também para
 - verificar se sequências de ações são válidas
 - gerar sentenças
 - gerar sequências de ações
- A geração pode ser automatizada
 - isso permite criar suítes de teste extensas a partir de uma gramática

Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

20

Gramáticas regulares



- Um conjunto de um ou mais 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
 - $\langle k \rangle ::= n_1 - n_2 [\langle g \rangle]$ ou $\langle k \rangle ::= \langle g \rangle^* ; \langle k \rangle ::= \langle g \rangle^+ ;$
eu prefiro esta notação
- A recursão **à direita** é uma gramática regular
 - $\langle k \rangle ::= (\langle g \rangle \mid \langle g \rangle \langle k \rangle) \rightarrow \langle k \rangle ::= \langle g \rangle^+$

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
 - ...

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 Captcha

Gramática

(Usuário correto | Usuário incorreto) (Senha correta | Senha incorreta)
 (Captcha correto | Captcha incorreto) (Login | Cancelar | Mudar | Esqueci)

Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

23

Gramáticas regulares



Gramática: (Usuário correto | Usuário incorreto) (Senha correta | Senha incorreta)
 (Captcha correto | Captcha incorreto) (Login | Cancelar | Mudar | Esqueci)

| | | | | |
|--------------------------|------------------------|--------------------------|----------|------------------|
| Usuário correto | Senha correto | Captcha correto | Login | → autoriza |
| Usuário correto | Senha correto | Captcha correto | Cancelar | → cancela |
| Usuário correto | Senha correto | Captcha correto | Mudar | → muda |
| Usuário correto | Senha correto | Captcha correto | Esqueci | → esqueci |
| Usuário incorreto | Senha correto | Captcha correto | Login | → erro |
| Usuário incorreto | Senha correto | Captcha correto | Cancelar | → cancela |
| Usuário incorreto | Senha correto | Captcha correto | Mudar | → erro |
| Usuário incorreto | Senha correto | Captcha correto | Esqueci | → erro |
| Usuário correto | Senha incorreto | Captcha correto | Login | → erro |
| Usuário correto | Senha incorreto | Captcha correto | Cancelar | → cancela |
| Usuário correto | Senha incorreto | Captcha correto | Mudar | → erro |
| Usuário correto | Senha incorreto | Captcha correto | Esqueci | → esqueci |
| Usuário correto | Senha correto | Captcha incorreto | Login | → erro |
| Usuário correto | Senha correto | Captcha incorreto | Cancelar | → cancela |
| Usuário correto | Senha correto | Captcha incorreto | Mudar | → erro |
| Usuário correto | Senha correto | Captcha incorreto | Esqueci | → erro |

...

Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

24

Laboratório de Engenharia de Software

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


Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

25

Laboratório de Engenharia de Software

FIM



Mar 2017

Arndt von Staa © LES/DI/PUC-Rio

26