6. Prova de Conceito

Este capítulo fornece uma avaliação da aplicação prática da ferramenta construída, através da especificação e geração de testes para um software construído por terceiros, coletado da Internet.

6.1.Seleção do software

Para selecionar o software para a avaliação, procuramos um projeto de código aberto, com implementação em Java e uso do *framework* Swing, fácil de colocar em funcionamento (sem muitas dependências e configurações complexas), que, preferencialmente, fizesse acesso a banco de dados e, principalmente, que tivesse seus requisitos especificados por uma descrição textual de casos de uso.

Através de pesquisa na Internet, encontramos um software simples, chamado de "Petshop Admin",[47] criado por alunos do curso de Bacharelado em Sistemas da Informação[48] da Universidade Federal Rural de Pernambuco.[49] O software teve sua especificação realizada pelo professor da disciplina de Laboratório de Programação, Giordano Cabral. A implementação foi realizada pelos alunos da turma, no segundo semestre de 2011.

O Petshop Admin, que visa gerenciar uma loja de animais de estimação, possui especificação de requisitos por descrição textual de casos de uso, implementação em linguagem Java com uso de Swing, projeto de código construído com Netbeans[50] e armazenamento dos dados em banco de dados MySQL.[51]

6.2.Adaptação do software

Foi preciso definir os nomes (propriedade `name`) dos *widgets* do software, para

[47] Disponível em: https://code.google.com/p/petshop-ufrpe-bsi/
[48] Página oficial do curso em: http://www.bsi.ufrpe.br/
[49] Página oficial da universidade em: http://www.ufrpe.br/
[50] http://netbeans.org/
[51] http://www.mysql.com/

(pelo menos) os casos de uso selecionados para a avaliação da ferramenta. A definição desta propriedade é necessária para a identificação dos *widgets* pelo *framework* de testes, durante a execução dos testes. Para a escolha dos nomes, foi utilizada a informação dos rótulos dos campos da interface com o usuário, além da DTCU.

Preferiu-se portar o projeto do Netbeans para o Eclipse e transformá-lo num projeto Maven, para facilitar o gerenciamento das dependências. Então, por motivo da necessidade de execução dos testes, foi adicionada a dependência dos *frameworks* TestNG e FEST (com extensão para Swing).

Para a execução dos testes, foi preciso realizar uma correção no método `main`, para que execute o programa dentro da *Event-Dispatch Thread* (EDT),[52] que é a *linha de execução* (*thread*) responsável pelo tratamento de eventos de interface com o usuário, da biblioteca Swing. Uma vez que a maioria dos objetos Swing não é *thread-safe*,[53] a execução fora da EDT poderia ocasionar riscos de interferência entre *threads* ou erros de inconsistência de memória. Assim, o código original que estava no método `main` foi movido para novo método `start`, e o conteúdo do método `main` ficou como o mostrado na Listagem 13:

```
public static void main(String[] args) {
	EventQueue.invokeLater( new Runnable() {
		public void run() {
			try {
				Main main = new Main();
				main.start();
			} catch ( Exception e ) {
				e.printStackTrace();
			}
		}
	} );
}
```

Listagem 13 - Correção do método Main no SST

6.3.Seleção dos casos de uso

Os casos de uso do software Petshop Admin são: *Efetuar Login*, *Cadastrar Cliente*, *Alterar Cliente*, *Pesquisar Cliente*, *Cadastrar Animal*, *Remover Animal*,

[52] http://docs.oracle.com/javase/tutorial/uiswing/concurrency/dispatch.html
[53] Propriedade em que uma *thread* somente manipula dados compartilhados de forma que garanta sua execução segura (isto é, livre de condições de corrida) em relação a outras *threads*.

Cadastrar Produto, *Alterar Produto*, *Pesquisar Produto*, *Realizar Serviço*, *Alterar Serviço*, *Pesquisar Serviço*, *Vender Serviço* e *Vender Produto*.

Foram selecionados, para a avaliação da ferramenta, os casos de uso *Efetuar Login*, *Cadastrar Cliente*, *Cadastrar Animal* e *Remover Animal*, pois estes, apesar de simples, permitem exercitar:

- Dependências entre casos de uso definidas por pré-condições;
- Dependências entre casos de uso definidas por chamadas a casos de uso;
- Fluxos alterativos recorrentes (*loop*);
- Regras de negócio que dependem de dados armazenados (banco de dados);
- Regras de negócio que usam lista finita de valores.

A especificação dos casos de uso selecionados é discutida na seção a seguir.

6.4.Especificação dos requisitos

Os requisitos funcionais do software Petshop Admin são originalmente descritos sob a forma de casos de uso, num documento de texto constante no projeto. Foi detectada, porém, a *incompletude dos requisitos* descritos, como a falta de alguns fluxos alternativos e regras de negócio. Uma definição com mais detalhes se torna necessária, principalmente, porque os testes funcionais precisam ser bem descritos para estabelecer a exata expectativa em relação à interação entre o ator (usuário) e o sistema.

Independente disto, uma especificação incompleta dá margem para que a equipe de desenvolvedores do software complete a especificação conforme sua intuição (e criatividade), que muitas vezes diverge da intenção original do projetista, que tenta mapear o problema real. E isto, sem dúvidas, acabou ocorrendo no software avaliado.

Assim, duas opções se apresentaram em relação à especificação dos requisitos do SST:

a) Completar a especificação com os detalhes que se acreditam serem os

mais apropriados;

b) Investigar a implementação do SST, colhendo dela os detalhes que faltam na especificação.

Cada uma destas opções impacta diretamente no número potencial de falhas a serem observadas pela ferramenta. Na primeira opção, existe o risco do SST não ter sido implementado conforme a especificação gerada, ocasionando um **grande número de falhas**. Na segunda opção, como a especificação segue a implementação, um **pequeno número de falhas** deve ser observado.

Para não criar uma especificação muito divergente da implementação, o que fatalmente ocasionaria um grande número de falhas, optou-se pela segunda opção. Adicionalmente, para verificar a eficácia da ferramenta, foram geradas:

1. Uma versão **da especificação** com pequenas divergências em relação ao SST;
2. Duas versões modificadas **do SST**, com emprego de mutantes.

Para gerar as versões com empregos de mutantes, levou-se em consideração, assim como no trabalho de Gutiérrez *et al*. (2008), o modelo de falhas de caso de uso introduzidas por Binder (2000), exibidos na Tabela 5, que define operadores mutantes para casos de uso. Considera-se o uso de operadores mutantes para casos de uso mais apropriado para testes funcionais (do que os operadores "clássicos"), uma vez que seu objetivo não é testar o código do SST em si, mas gerar mudanças no comportamento do SST que possam ser observadas por testes funcionais.

Cada versão mutante do SST utilizou apenas um operador mutante.

6.5.Considerações sobre as especificações

Na especificação da descrição textual dos casos de uso, exceto para *Efetuar Login*, não foram descritos os *fluxos alternativos de cancelamento* da execução do caso de uso. A utilização deste fluxo praticamente dobra o número de cenários do caso de uso e, consequentemente, o tempo de execução dos testes, uma vez que ele geralmente ocorre ao término do caso de uso, permitindo que outros fluxos passem por ele, gerando cenários. O fluxo foi mantido apenas em *Efetuar Login*, pelo fato do número potencial de cenários deste caso de uso ser pequeno, não

impactando significativamente no tempo geral do teste.

Um caso de uso adicional, chamado de *Acessar Sistema*, foi criado para representar o acesso à tela principal, provendo acesso aos demais casos de uso do sistema, o que é necessário para a execução dos testes funcionais.

Na definição das regras de negócio dos casos de uso, foi utilizado um banco de dados "`petshoptest`", que contém dados a serem utilizados para o teste do sistema. Esse banco de dados de teste possui os dados compatíveis com os esperados pelo SST.

Para a descrição textual dos casos de uso selecionados são utilizados os seguintes campos: *Identificação (ID), Nome, Descrição*, *Acessível diretamente*, *Pré-condições*, *Fluxo de Ativação, Fluxo Principal, Fluxo(s) Alterativo(s), Pós-condições* e *Regras de Negócio*. A especificação original dos casos de uso, como encontrada no PetShop Admin, é descrita no Anexo A.

6.6.Considerações sobre os testes

Para a geração e a execução dos testes, foi utilizado um notebook (Dell Inspiron 15SE) com um processador com cinco núcleos[54] (Intel Core i7), 8GB de RAM e disco rígido de 5.200 RPM. O sistema operacional utilizado foi o Windows 7. Entretanto, a execução se deu em uma **máquina virtual**, criada no Windows 7 com o software Virtual Box,[55] contendo o sistema operacional Windows XP SP3, apenas 1 processador/núcleo e 1GB de RAM. Esta execução foi preferida para que os testes funcionais não tomassem o controle do mouse e do teclado, no Windows 7, enquanto o teste era executado, além de simular sua execução num ambiente com processador e memória mais limitados.

6.7.Especificação baseada no SST

Nesta especificação, os detalhes não presentes na especificação original são colhidos da implementação do SST. As regras de negócio modeladas nesta especificação são somente aquelas cujo sistema implementado leva em conta, ou

[54] Apesar disto, os algoritmos da ferramenta, na versão atual, não foram projetados para rodar em paralelo.
[55] O Oracle Virtual Box é *opensource* e está disponível em: https://www.virtualbox.org

seja, outras regras de negócio potencialmente interessantes não foram acrescentadas.

6.7.1.Caso de Uso Efetuar Login

A Tabela 14, a seguir, apresenta a descrição textual do caso de uso Efetuar Login.

Tabela 14 - Descrição textual do caso de uso Efetuar Login

ID/Nome	UC1/Efetuar Login	
Descrição	Concede acesso ao sistema	
Acessível diretamente	Não	
Pré-condições	Haver um usuário cadastrado	
Fluxo de Ativação	Não há	
Fluxo Principal [BF]	Ator	Sistema
P1		Exibe a tela de Login [RN1]
P2	Informa nome de usuário [RN2] e a senha [RN3]	
P3	Clica em Entrar [RN4] [AF1]	
P4		Verifica o usuário e a senha
P5		Fecha a tela de Login
Pós-condição do Fluxo Principal [BF-POS1]	Login efetuado	
Fluxo Alternativo 1 - [AF1] – "Cancelar"	Inicia em BF-P3	
P1	Clica em Cancelar [RN5]	
P2		Fecha a tela de Login
Regras de Negócio	Especificação detalhada	
RN1	"Tela de Login" é um "frame" de nome interno "JanelaLogin";	
RN2	• "nome de usuário" é uma "caixa de texto" de nome interno "usuario" e tipo "String"; • Seu fornecimento é obrigatório e sua falta faz exibir a mensagem "`Preencha o campo de usuário`"; • Seu valor vem da consulta "`SELECT login FROM usuario`" no banco de dados de teste;	
RN3	• "senha" é uma "caixa de texto" de nome interno "senha" e tipo "String"; • Seu fornecimento é obrigatório e sua falta faz exibir a mensagem "`Preencha o campo de senha`"; • Seu valor vem da consulta "`SELECT senha FROM usuario WHERE login = ?`", no banco de dados de teste, sendo "`?`" o valor fornecido para o nome de usuário;	
RN4	"Entrar" é um "botão" de nome interno "entrar";	
RN5	"Cancelar" é um "botão" de nome interno "cancelar";	

6.7.2.Caso de Uso Acessar Sistema

A Tabela 15, a seguir, apresenta a descrição textual do caso de uso Acessar Sistema.

Tabela 15 - Descrição textual do caso de uso Acessar Sistema

ID/Nome	UC2/Acessar Sistema	
Descrição	Possibilita o acesso aos demais casos de uso do sistema	
Acessível diretamente	Sim	
Pré-condições	Login efetuado [UC1-BF-POS1]	
Fluxo de Ativação	Não há	
Fluxo Principal [BF]	Ator	Sistema
P1		Exibe a Tela Principal [RN1]
Pós-condições do Fluxo Principal [BF-POS1]	Sistema acessado	
Fluxos Alternativos	Não há	
Regras de Negócio	Especificação detalhada	
RN1	"Tela Principal" é um "frame" de nome interno "Janela-Principal";	

6.7.3.Caso de Uso Cadastrar Cliente

A Tabela 16, a seguir, apresenta a descrição textual do caso de uso Cadastrar Cliente.

Tabela 16 - Descrição textual do caso de uso Cadastrar Cliente

ID/Nome	UC3/Cadastrar Cliente	
Descrição	Realiza o cadastro de um cliente do Petshop	
Acessível diretamente	Não	
Pré-condições	Login efetuado [UC1-BF-POS1]	
Fluxo de Ativação	Ator	Sistema
P1	Clica em Clientes [RN1]	
P2	Clica em Novo [RN2]	
Fluxo Principal [BF]	Ator	Sistema
P1		Exibe a Tela de Cliente [RN3]
P2	Informa o nome [RN4],	
P3	Seleciona o sexo [RN5]	
P4	Informa rua [RN6], número [RN7], complemento [RN8], bairro [RN9], cidade [RN10],	
P5	Seleciona a UF [RN11]	
P6	Informa o CEP [RN12], RG [RN13], CPF [RN14], e-mail [RN15], telefone [RN16], celular [RN17]	
P7	Clica em Adicionar [RN18]	

P8		Chama o caso de uso Adicional Animal (UC4)
P9	Informa informações adicionais [RN19]	
P10	Clica em Cadastrar [RN20]	
P11		Verifica nome, sexo, rua, complemento, bairro, cidade, UF, CEP, RG, CPF, e-mail, telefone, celular, informações adicionais.
P12		Fecha a Tela de Cliente
Pós-condição do Fluxo Principal [BF-POS1]	Cliente cadastrado	
Fluxo Alternativo 1 - [AF1] – "Remover Animal"	Inicia em BF-P9, retorna para BF-P9	
P1		Chama o caso de uso Remover Animal (UC5)
Regras de Negócio	Especificação detalhada	
RN1	"Cliente" é um "botão" de nome interno "clientes";	
RN2	"Novo" é um "botão" de nome interno "novo";	
RN3	"Tela de Cliente" é uma "janela de diálogo" de nome interno "JanelaCliente";	
RN4	• "nome" é uma "caixa de texto" de nome interno "nome" e tipo "String"; • Seu fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Possui comprimento mínimo 1, não exibindo mensagem caso menor; • Possui comprimento máximo 80, não exibindo mensagem caso maior;	
RN5	• "sexo" é uma "caixa de seleção" de nome interno "sexo" e tipo "String"; • É fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Admite apenas um dos seguintes valores: "`MASCULINO`", "`FEMININO`", não exibindo mensagem caso diferente;	
RN6	• "rua" é uma "caixa de texto" de nome interno "rua" e tipo "String"; • Seu fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Possui comprimento mínimo 1, não exibindo mensagem caso menor; • Possui comprimento máximo 70, não exibindo mensagem caso maior;	
RN7	• "número" é uma "caixa de texto" de nome interno "numero" e tipo "String"; • Possui comprimento máximo 5, não exibindo mensagem caso maior;	

RN8	• "complemento" é uma "caixa de texto" de nome interno "complemento" e tipo "String"; • Possui comprimento máximo 30, não exibindo mensagem caso maior;
RN9	• "bairro" é uma "caixa de texto" de nome interno "bairro" e tipo "String"; • Possui comprimento máximo 30, não exibindo mensagem caso maior;
RN10	• "cidade" é uma "caixa de texto" de nome interno "cidade" e tipo "String"; • Seu fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Possui comprimento mínimo 1, não exibindo mensagem caso menor; • Possui comprimento máximo 30, não exibindo mensagem caso maior;
RN11	• "UF" é uma "caixa de seleção" de nome interno "uf" e tipo "String"; • Seu fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Seu valor vem da consulta "`SELECT uf FROM estado`", no banco de dados de teste,
RN12	• "CEP" é uma "caixa de texto" de nome interno "cep" e tipo "String"; • Seu formato segue a expressão regular "`[0-9]{8}`", não exibindo mensagem caso diferente;
RN13	• "RG" é uma "caixa de texto" de nome interno "rg" e tipo "String"; • Seu fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Possui comprimento mínimo 1, não exibindo mensagem caso menor; • Possui comprimento máximo 15, não exibindo mensagem caso maior;
RN14	• "CPF" é uma "caixa de texto" de nome interno "cpf" e tipo "String"; • Seu fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Seu formato segue a expressão regular "`[0-9]{11}`", não exibindo mensagem caso diferente;
RN15	• "e-mail" é uma "caixa de texto" de nome interno "email" e tipo "String"; • Possui comprimento máximo 70, não exibindo mensagem caso maior;

RN16	• "telefone" é uma "caixa de texto" de nome interno "telefone" e tipo "String"; • Seu fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Seu formato segue a expressão regular "`([0-9]{8}|[0-9]{10})`", não exibindo mensagem caso diferente;
RN17	• "celular" é uma "caixa de texto" de nome interno "celular" e tipo "String"; • Seu fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Seu formato segue a expressão regular "`([0-9]{8}|[0-9]{10})`", não exibindo mensagem caso diferente;
RN18	"Adicionar" é um "botão" de nome interno "adicionar";
RN19	• "informações adicionais" é uma "caixa de texto" de nome interno "info" e tipo "String"; • Possui comprimento máximo 400, não exibindo mensagem caso maior;
RN20	"Cadastrar" é um "botão" de nome interno "cadastrar";

6.7.4.Caso de Uso Adicionar Animal

A Tabela 17, a seguir, apresenta a descrição textual do caso de uso Adicionar Animal.

Tabela 17 - Descrição textual do caso de uso Adicionar Animal

ID/Nome	UC4/Adicionar Animal	
Descrição	Permite cadastrar um animal de um cliente	
Acessível diretamente	Sim	
Pré-condições	Login efetuado [UC1-BF-POS1]	
Fluxo de Ativação	Não há	
Fluxo Principal [BF]	Ator	Sistema
P1		Exibe a Tela de Animal [RN1]
P2	Informa o nome [RN2]	
P3	Seleciona o sexo [RN3]	
P4	Informa a data de nascimento [RN4]	
P5	Seleciona a espécie [RN5]	
P6	Informa a raça [RN6] e informações adicionais [RN7]	
P7	Clica em Adicionar [RN8]	
P8		Verifica o nome e o nascimento
P9		Fecha a Tela de Animal
Pós-condição do Fluxo Principal [BF-POS1]	Animal adicionado	
Regras de Negócio	Especificação detalhada	

RN1	"Tela de Animal" é um "janela de diálogo" de nome interno "JanelaAnimal";
RN2	• "nome" é uma "caixa de texto" de nome interno "nome" e tipo "String"; • Seu fornecimento é obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os\nseguintes campos obrigatórios:\n\n%s`"; • Possui comprimento mínimo 1, não exibindo mensagem caso menor; • Possui comprimento máximo 50, não exibindo mensagem caso maior;
RN3	• "sexo" é uma "caixa de seleção" de nome interno "sexo" e tipo "String"; • É fornecimento obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os \nseguintes campos obrigatórios:\n\n%s`"; • Admite apenas um dos seguintes valores: "`MACHO`", "`FÊMEA`", não exibindo mensagem caso diferente;
RN4	• "data de nascimento" é uma "caixa de texto" de nome interno "nascimento" e tipo "Date";
RN5	• "espécie" é uma "caixa de seleção" de nome interno "especie" e tipo "String"; • Admite apenas um dos seguintes valores: "`CÃO`", "`GATO`", "`AVE`", "`ROEDOR`", "`OUTRO`", não exibindo mensagem caso diferente;
RN6	• "raça" é uma "caixa de seleção" de nome interno "raca" e tipo "String"; • Seu fornecimento é obrigatório e sua falta faz exibir a mensagem "`Você esqueceu de preencher os\nseguintes campos obrigatórios:\n\n%s`"; • Possui comprimento mínimo 2, não exibindo mensagem caso menor; • Possui comprimento máximo 50, não exibindo mensagem caso maior;
RN7	• "informações adicionais" é uma "caixa de texto" de nome interno "info" e tipo "String"; • Possui comprimento máximo 400, não exibindo mensagem caso maior;
RN8	"Adicionar" é um "botão" de nome interno "adicionar";

6.7.5.Caso de Uso Remover Animal

A Tabela 18, a seguir, apresenta a descrição textual do caso de uso Remover Animal.

Tabela 18 - Descrição textual do caso de uso Remover Animal

ID/Nome	UC5/Remover Animal
Descrição	Possibilita remover o anima de um cliente

Acessível diretamente	Sim	
Pré-condições	Animal adicionado [UC4-BF-POS1]	
Fluxo de Ativação	Ator	Sistema
P1	Seleciona *o primeiro* animal [RN1]	
P2	Clica em Remover [RN2]	
Fluxo Principal [BF]	Ator	Sistema
P1		Remove o animal selecionado
Pós-condições do Fluxo Principal [BF-POS1]	Animal removido	
Regras de Negócio	Especificação detalhada	
RN1	• "animal" é uma "caixa de seleção" de nome interno "animal" e tipo "String";	
RN2	• "Remover" é um "botão" de nome interno "remover";	

6.7.6.Geração dos testes

Para a especificação baseada no SST, descrita na seção 6.7, a ferramenta gerou 204 CTSVO num arquivo JSON com 200.428 linhas e 5,96 MB. Esta geração, considerando a execução dos algoritmos de geração dos CTSVO e a gravação do arquivo em disco, durou 7 segundos.

Um trecho do arquivo contendo os CTSVO em JSON pode ser visto na Listagem 14, a seguir.

```
{
  "name" : "My suite",
  "softwareName" : "PetShop Admin",
  "creation" : "2013-03-22T16:47:45.144-03:00",
  "connections" : [ ],
  "testCases" : [ {
    "name" : "CadastrarCliente_BF_AF1_BF_Test",
    "useCaseName" : "Cadastrar Cliente",
    "scenarioName" : "BF,AF1,BF",
    "includeFiles" : [
       "petshop.gui.JanelaPrincipal",
       "petshop.gui.JanelaAnimal",
       "petshop.gui.JanelaLogin",
       "petshop.gui.JanelaCliente" ],
    "scripts" : [ ],
    "testMethods" : [ {
      "name" : "all_with_minimum_values",
      "strategyKind" : "all_with_minimum_values",
      "scenarioName" : "BF,AF1,BF",
      "expectedSuccess" : true,
      "steps" : [ {
        "@class" : "semantic.SemanticActionStep",
        "id" : 1,
        "useCaseId" : 1,
        "flowId" : 0,
        "stepId" : 0,
        "actionName" : "show",
        "elements" : [ {
          "type" : "frame",
          "internalName" : "JanelaLogin",
          "name" : "Janela de Login",
          "value" : null,
          "valueConsideredValid" : null,
          "valueOption" : null
        } ]
      }, {
...
```

Listagem 14 - Trecho de arquivo JSON com CTSVO

A extensão da ferramenta gerou 3 arquivos de teste Java, num total de 568 KB, em 621 milissegundos, considerando a leitura dos CTSVO, sua transformação em código Java e a gravação em disco. Dados sobre estes arquivos podem ser visualizados na Tabela 19, abaixo:

Tabela 19 - Detalhes dos arquivos de código gerados

Caso de Uso alvo	Cenário	Arquivo gerado	
Cadastrar Cliente	BF, AF1, BF	Arquivo:	CadastrarCliente_BF_AF1_BF_Test.java
		Linhas:	3.767
		Métodos:	100
		Testes:	92
	BF	Arquivo:	CadastrarCliente_BF_Test.java
		Linhas	3.767

		Métodos:	**100**
		Testes:	**92**
Efetuar Login	BF	Arquivo:	**EfetuarLogin_BF_Test**
		Linhas:	**247**
		Métodos:	**24**
		Testes:	**20**

A Listagem 15, abaixo, apresenta um trecho do arquivo **`CadastrarCliente_BF_AF1_BF_Test.java`**, que contém código de teste gerado pela extensão da ferramenta.

```
...

/**
 * Tests the "Cadastrar Cliente" use case
 * in the following scenarios: BF,AF1,BF
 *
 * @author FunTester
 */
public class CadastrarCliente_BF_AF1_BF_Test {

      public BasicRobot robot;

      // Helper method
      public FrameFixture findJanelaLogin() {
            return WindowFinder.findFrame(
                JanelaLogin.class ).using( robot );
      }

      // Helper method
      public FrameFixture findJanelaPrincipal() {
            return WindowFinder.findFrame(
               JanelaPrincipal.class ).using( robot );
      }

      // Helper method
      public DialogFixture findJanelaAnimal() {
            return WindowFinder.findDialog(
                JanelaAnimal.class ).using( robot );
      }

      // Helper method
      public DialogFixture findJanelaCliente() {
            return WindowFinder.findDialog(
                JanelaCliente.class ).using( robot );
      }

      @BeforeClass
      public void one_time_setup() {
            FailOnThreadViolationRepaintManager.install();
      }
```

```
@BeforeMethod
public void setup() {
      robot = (BasicRobot)
          BasicRobot.robotWithNewAwtHierarchy();
      robot.settings().timeoutToBeVisible( 5 );
      // Start the application via main()
      ApplicationLauncher.application(
          "petshop.Main" ).start();
}

@AfterMethod
public void tear_down() {
      robot.cleanUp();
}

...

@Test
public void all_with_random_values() {
      FrameFixture janelaLogin1 = findJanelaLogin();
      janelaLogin1.show(); // ssid=6875
      janelaLogin1.textBox( "usuario" )
          .setText( "lucena" ); // ssid=6876
      janelaLogin1.textBox( "senha" )
          .setText( "cjpl" ); // ssid=6877
      janelaLogin1.button( "entrar" ).click(); // ssid=6878

      FrameFixture janelaPrincipal1 = findJanelaPrincipal();
      janelaPrincipal1.show(); // ssid=6880
      janelaPrincipal1.button( "clientes" )
          .click(); // ssid=6881
      janelaPrincipal1.button( "novo" )
         .click(); // ssid=6882

      DialogFixture janelaCliente1 = findJanelaCliente();
      janelaCliente1.show(); // ssid=6883
      janelaCliente1.textBox( "nome" )
         .setText( "dj9|7atSfWuaM$ZCAk(J
             :7PmA{/5B@7DyUvr mXODM|yn*06Wl$
             1aDZicCj|V5CnB{:_S1BzkLC'xk" ); // ssid=6884
      janelaCliente1.comboBox( "sexo" )
          .selectItem( "MASCULINO" ); // ssid=6885
      janelaCliente1.textBox( "rua" )
          .setText( "x?T?w]8)1/T7.r$l:y|Z
              Z*X7%/;f9%AcrOT$3xB'{_;4yzW
              >;Dx:iGqaly`4" ); // ssid=6886
      janelaCliente1.textBox( "numero" )
          .setText( "6e}" ); // ssid=6887
      janelaCliente1.textBox( "complemento" )
          .setText( "N;hqq&lo2PM/Rn" ); // ssid=6888
      janelaCliente1.textBox( "bairro" )
          .setText( "uy??W7BJYLMNRjR0qq4~T0" ); // ssid=6889
      janelaCliente1.textBox( "cidade" )
          .setText( "" ); // ssid=6890
      janelaCliente1.comboBox( "uf" )
          .selectItem( "AP" ); // ssid=6891
      janelaCliente1.textBox( "cep" )
          .setText( "88982538" ); // ssid=6892
      janelaCliente1.textBox( "rg" )
          .setText( "" ); // ssid=6893
```

```
            janelaCliente1.textBox( "cpf" )
                .setText( "28823948651" ); // ssid=6894
            janelaCliente1.textBox( "email" )
                .setText( "Z`w::2%ys=W?wB`pL]0aA
                    >bG@VpQHzw73C'S(:jB&ncM2I~`b
                    cC,dlNGp>%Z?R:*" ); // ssid=6895
            janelaCliente1.textBox( "telefone" )
               .setText( "3657944413" ); // ssid=6896
            janelaCliente1.textBox( "celular" )
               .setText( "82260536" ); // ssid=6897
            janelaCliente1.button( "adicionar" )
               .click(); // ssid=6898

            DialogFixture janelaAnimal1 = findJanelaAnimal();
            janelaAnimal1.show(); // ssid=6899
            janelaAnimal1.textBox( "nome" )
                .setText( "Q" ); // ssid=6900
            janelaAnimal1.comboBox( "sexo" )
                .selectItem( "FÊMEA" ); // ssid=6901
            janelaAnimal1.textBox( "nascimento" )
                .setText( "25/01/0001" ); // ssid=6902
            janelaAnimal1.comboBox( "especie" )
                .selectItem( "AVE" ); // ssid=6903
            janelaAnimal1.textBox( "raca" )
                .setText( "haH3y}MQixAc@g%7X#v?Ms" ); // ssid=6904
            janelaAnimal1.textBox( "info" )
                .setText( "7D74oeyM+%7q&p{-K3P?`}X-
                    D>ej4.=X,C)<G&vezqA|A'>iQjfgocw
                    @qQdMW_5}cXvv[-Q4>[*1c1t<i@L!K}
                    t4:nb]4t1(4*fN" ); // ssid=6905
            janelaAnimal1.button( "adicionar" )
                .click(); // ssid=6906

            janelaCliente1.textBox( "info" )
               .setText( "}L&^+(:R$n%hM{e6WM#l/m`Ks
                   g>dw}8o[U}Ba6wgj.oBDO$kI-dA|nx(|
                   I=i8?/%Lu>s2wU>^V}blwKF/{)t55lxO
                   d$0NA0*q@I,/YfNW8NspwB5(6rFm@3S)
                   Aa8_UT5}kRqd%6'<;Vh+y7@7}(*p_EQv
                   kx+LlA(#<T_UP:KK2#C7!M:)g95Ik2!1
                   mxpS4,#pmfv>%EH3A[.|%%thc_wO7n|b
                   J.=OY>3-4Sj27sHTh2sYY^/^uZ`Zu]}A
                   wzu@0UYj&R7^oUmm*2Yu8P4nIZKJDtfu
                   ,3;`e^$V%" ); // ssid=6908
            janelaCliente1.button( "cadastrar" )
               .click(); // ssid=6909
      }
...
```

Listagem 15 - Exemplo de código-fonte de teste gerado pela ferramenta

Cada comentário à frente das linhas de código (acima) serve para identificar o passo do teste semântico correspondente à linha de código. Essa identificação ocorre no processo de pré-análise dos resultados, indicada na seção 4.9.

6.7.7.Execução dos testes

A Tabela 20, abaixo, exibe os tempos envolvidos na execução dos testes (Java). A cada método de teste, o SST é colocado em execução e todos os casos de uso envolvidos no cenário são executados (incluindo os casos de uso que são pré-condições do *caso de uso sob teste*), simulando as operações de um usuário sobre o sistema.

Tabela 20 - Desempenho da execução dos testes

Número de testes	204
Tempo de execução	18m26s
Tempo médio por teste	5s (5424ms)
Teste mais lento	7s (7501ms)
Teste mais rápido	0s (361ms)

Para a realização da execução destes testes, o *framework* TestNG foi configurado para esperar no máximo 5 segundos por um evento (ao invés dos 30 segundos utilizados por *default*) e para utilizar o tempo de 60 milissegundos (que é o valor *default*) entre cada entrada de dados.

6.7.8.Transformação dos resultados de execução

A pré-análise e a transformação dos resultados de execução para o formato padrão da ferramenta ocorreram, juntas, em 640 milissegundos. O processo gerou o arquivo `results.json`, contendo 7.415 linhas e 635 KB. A Listagem 16, abaixo, exibe um trecho desse arquivo.

```
{
  "totalTests" : 204,
  "totalPassed" : 128,
  "totalSkipped" : 0,
  "totalFailures" : 76,
  "totalErrors" : 0,
  "totalUnknown" : 0,
  "timeInMillis" : 1106552,
  "creation" : "2013-03-21T17:44:06.443-03:00",
  "toolName" : "Funtester for TestNG and FEST Swing",
  "targetLanguage" : "Java",
  "targetGUIFrameworks" : "Swing",
  "targetTestFrameworks" : "TestNG 6.8, FEST Swing 1.2.1",
  "originalTestResultFile" :
"C:\\dev\\workspace\\funtester\\src\\sut\\petshopstore\\target\\sure
fire-reports\\testng-results.xml",
  "suites" : [ {
```

```
    "totalTests" : 204,
    "totalPassed" : 128,
    "totalSkipped" : 0,
    "totalFailures" : 76,
    "totalErrors" : 0,
    "totalUnknown" : 0,
    "timeInMillis" : 1106552,
    "name" : "Command line suite",
    "testCases" : [ {
      "className" : "petshop.funtester.EfetuarLogin_BF_Test",
      "methods" : [ {
        "name" : "setup",
        "timeInMillis" : 500,
        "status" : "PASS",
        "exceptionClass" : null,
        "exceptionMessage" : null,
        "stackTrace" : null,
        "erroneousFile" : null,
        "erroreousFileLineNumber" : null,
        "erroreousLineOfCode" : null,
        "erroneousSemanticStepId" : null,
        "forConfiguration" : true
      }, {
...
```

Listagem 16 - Trecho de arquivo JSON com o resultado da execução dos testes

6.7.9.Análise dos resultados

A totalização dos testes é dividida em cinco categorias:

i. **Testes que passaram**: São os testes que terminaram com sucesso;

ii. **Testes pulados**: São os testes que foram pulados devido a alguma dependência de outro teste que não foi executado. Um teste pulado não é considerado necessariamente uma falha.[56] Essa totalização foi mantida para tornar a ferramenta compatível com *frameworks* que suportam dependências entre testes ou grupos de teste, como TestNG.

iii. **Testes que falharam**: São testes cuja execução falhou devido à assertiva não ter sido atendida ou porque houve algum problema de execução durante o teste, como o fato de um *widget* não sido encontrado;

iv. **Testes que obtiveram erro**: São testes cujo código-fonte contém algum erro de compilação/interpretação. Espera-se que não existam testes com erro, uma vez que a geração do código-fonte esteja correta;

[56] Mais informações sobre testes pulados em: http://testng.org/doc/documentation-main.html

v. **Testes com resultado desconhecido**: São testes que não se enquadram em nenhuma das categorias anteriores e que não tiveram seu problema detectado pelo *framework* de testes ou pelo FunTester. Espera-se que não existam testes com resultado desconhecido, uma vez que a geração do código-fonte esteja correta e que os *frameworks* de teste funcionem corretamente.

A Figura 20, a seguir, exibe a tela do FunTester que apresenta os resultados da execução dos testes, classificando as totalizações conforme descrito acima. Para fornecer uma visualização do percentual de cada total, a tela fornece um gráfico de pizza (*pie chart*). O *total pulado* e o *total com resultado desconhecido* só são apresentados se forem maiores que zero (0).

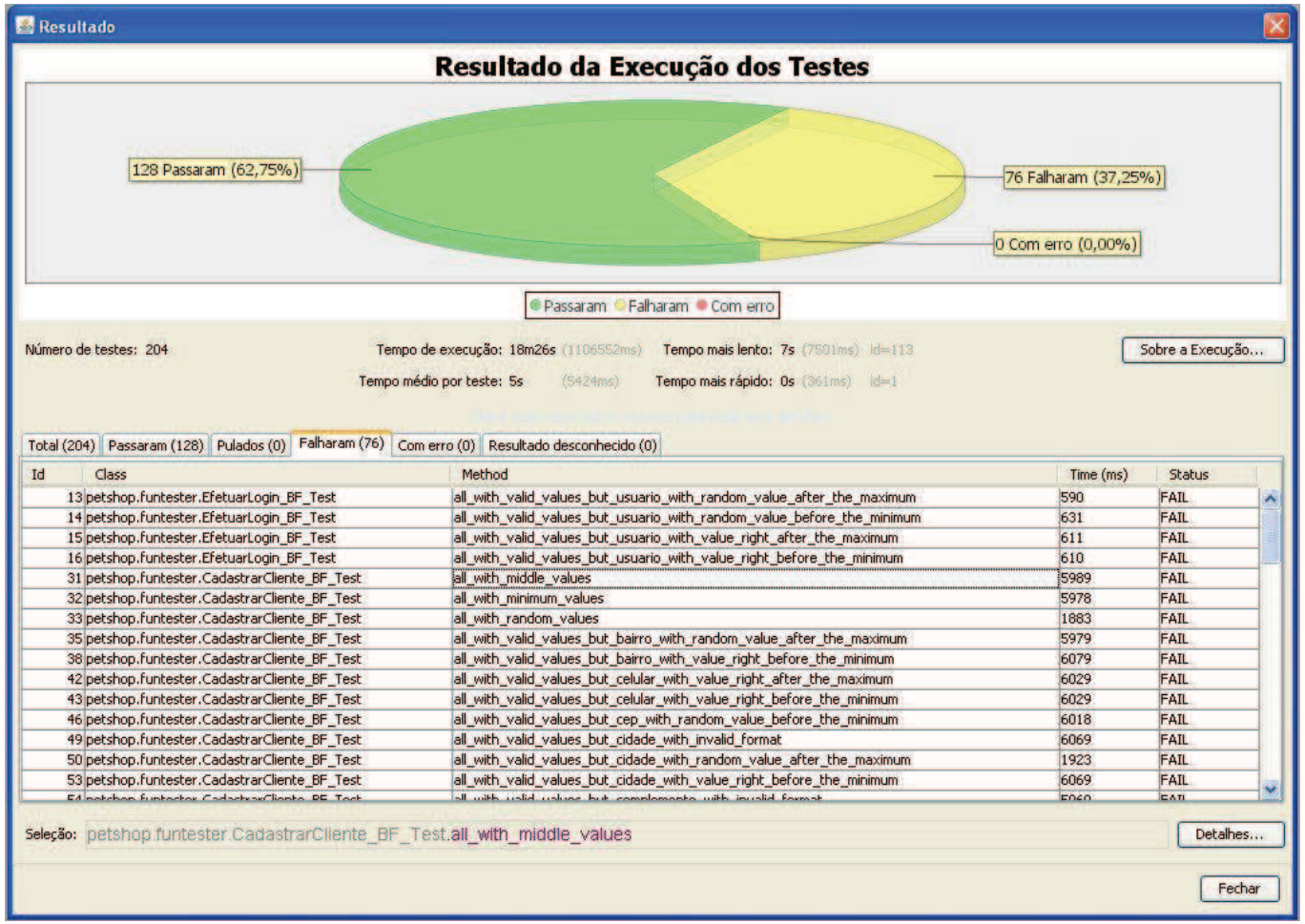

Figura 20 - Tela com os resultados da execução dos testes

Dos 204 testes executados, 76 falharam (37,25%) e nenhum obteve erro. Esse número poderia ser maior caso a especificação não tivesse levado em conta certas regras colhidas da implementação, conforme discutido na seção 6.4.

Para ilustrar o motivo da ocorrência das falhas, serão apresentados detalhes sobre duas falhas observadas nos testes. A Figura 21 exibe detalhes relacionados a um teste para o caso de uso *Efetuar Login* (seção 6.7.1), que apresenta uma falha devido à especificação ter esperado que fosse apresentada uma mensagem indicando que o usuário informado fosse inválido, mas esta mensagem não foi exibida pelo SST. É importante notar que a mensagem esperada pela especificação foi colhida do próprio código do SST, não sendo ela lexicamente diferente.

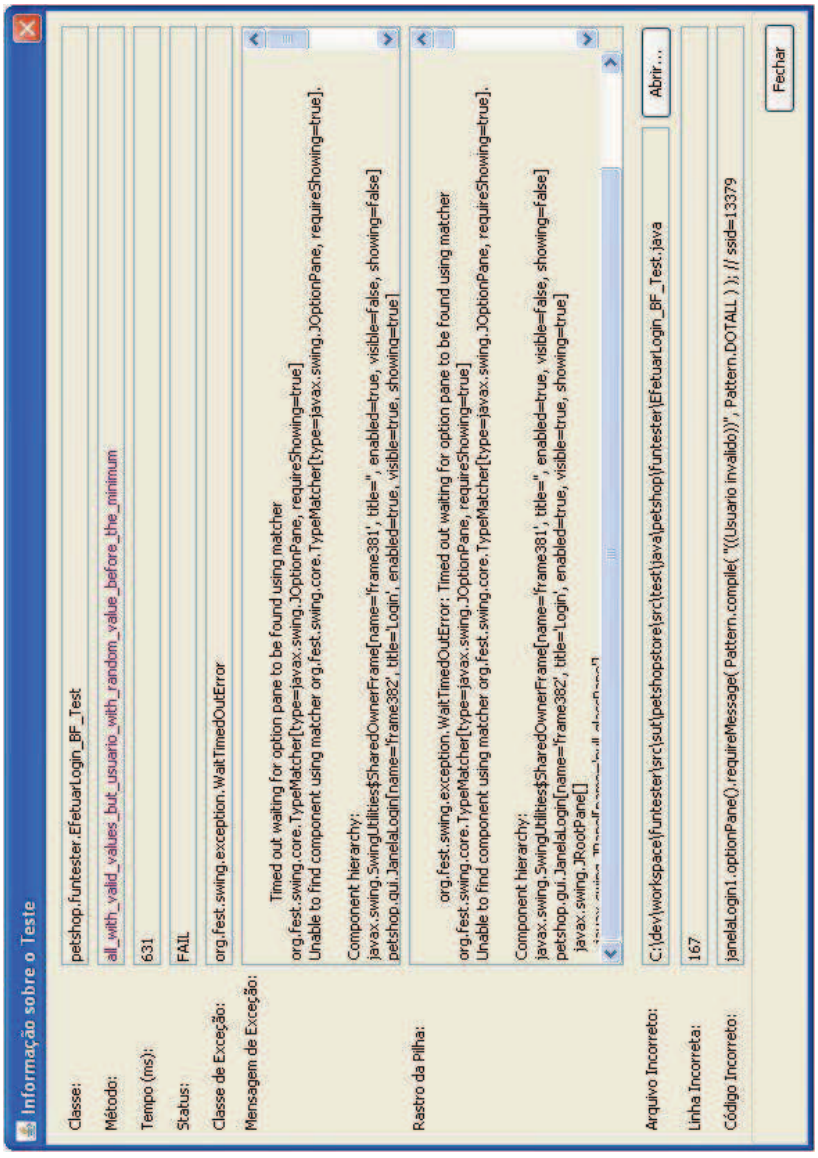

Figura 21 – Primeiro exemplo de detalhes sobre falha

A Listagem 17 apresenta o código-fonte do teste onde a falha foi observada, conforme indicação da tela anterior.

```
@Test
public void
all_with_valid_values_but_usuario_with_random_value_before_the_min
imum() {
      FrameFixture janelaLogin1 = findJanelaLogin();
      janelaLogin1.show(); // ssid=13375
      janelaLogin1.textBox( "usuario" )
          .setText( "+#7){FO1x3" ); // ssid=13376
      janelaLogin1.textBox( "senha" )
          .setText( "[mwbP4d&tK0r#HZ$f#>/k|NA8
              JA6-}!65C3-?X{/%LfgxI/Z5oqV[uI)8
              Ss_.-1vo)JNu$OfwpwIvrvRu>GbGryl}3
              6187,debc)XA" ); // ssid=13377
      janelaLogin1.button( "entrar" ).click(); // ssid=13378
      janelaLogin1.optionPane().requireMessage(
          Pattern.compile( "((Usuario invalido))",
              Pattern.DOTALL ) ); // ssid=13379

  }
```

Listagem 17 - Código-fonte do teste onde a primeira falha exemplificada foi observada

Ao se investigar a causa do problema no SST, descobriu-se que o texto usado na senha gerou uma exceção no programa (`java.sql.SQLException`), que foi apresentada *apenas na saída para o console*. Com a exceção, a mensagem esperada não foi exibida pelo SST, o que foi detectado pelo teste. Ou seja, os dados de entrada gerados com o FunTester conseguiram explorar uma falha do SST.

A Figura 22, a seguir, apresenta detalhes sobre outra falha, relacionada ao caso de uso *Cadastrar Cliente* (seção 6.7.3). Como é possível observar, essa falha tem relação com o fornecimento de um determinado valor para o campo "numero".

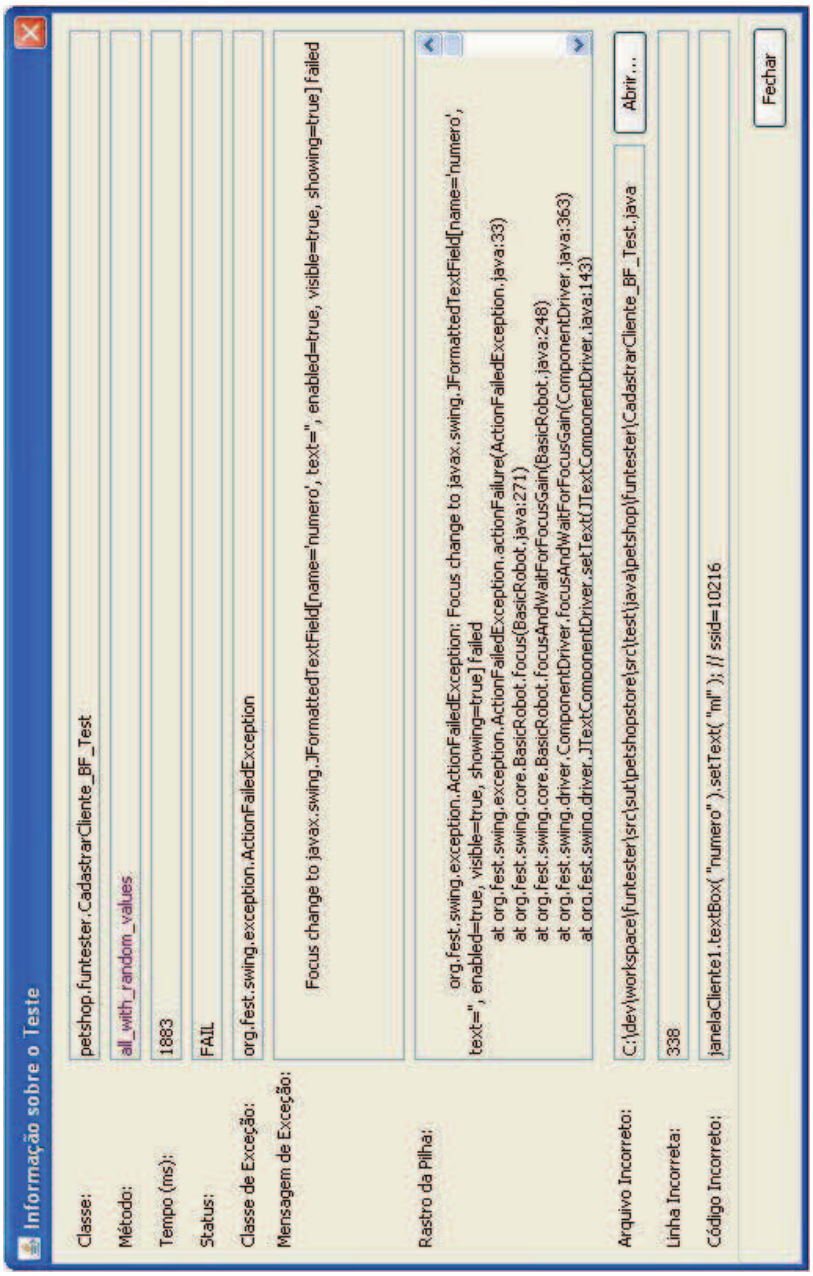

Figura 22 – Segundo exemplo de detalhes sobre falha

A Listagem 18, a seguir, apresenta o código-fonte do teste onde a falha foi observada. Pode-se notar que foi gerado um conteúdo alfanumérico para um campo "numero" e esse valor pode ter causado o erro.

```
@Test
public void all_with_random_values() {
      FrameFixture janelaLogin1 = findJanelaLogin();
      janelaLogin1.show(); // ssid=10204
      janelaLogin1.textBox( "usuario" )
          .setText( "simone" ); // ssid=10205
      janelaLogin1.textBox( "senha" )
          .setText( "sdbj" ); // ssid=10206
      janelaLogin1.button( "entrar" ).click(); // ssid=10207
```

```
        FrameFixture janelaPrincipal1 = findJanelaPrincipal();
        janelaPrincipal1.show(); // ssid=10209
        janelaPrincipal1.button( "clientes" ).click(); // ssid=10210
        janelaPrincipal1.button( "novo" ).click(); // ssid=10211

        DialogFixture janelaCliente1 = findJanelaCliente();
        janelaCliente1.show(); // ssid=10212
        janelaCliente1.textBox( "nome" )
            .setText( "l*_uk2'9O/uUyhB&]CFc'KJ
                fE'G5YN" ); // ssid=10213
        janelaCliente1.comboBox( "sexo" )
            .selectItem( "MASCULINO" ); // ssid=10214
        janelaCliente1.textBox( "rua" )
            .setText( "{`K:*_" ); // ssid=10215
        janelaCliente1.textBox( "numero" )
            .setText( "ml" ); // ssid=10216
        janelaCliente1.textBox( "complemento" )
            .setText( " " ); // ssid=10217
        janelaCliente1.textBox( "bairro" )
            .setText( "kd}X7D(#F+NvISG<+H-" ); // ssid=10218
        janelaCliente1.textBox( "cidade" )
            .setText( "N" ); // ssid=10219
        janelaCliente1.comboBox( "uf" )
            .selectItem( "GO" ); // ssid=10220
        janelaCliente1.textBox( "cep" )
            .setText( "89314051" ); // ssid=10221
        janelaCliente1.textBox( "rg" )
            .setText( "" ); // ssid=10222
        janelaCliente1.textBox( "cpf" )
            .setText( "84622893501" ); // ssid=10223
        janelaCliente1.textBox( "email" )
            .setText( "xOKX!Id" ); // ssid=10224
        janelaCliente1.textBox( "telefone" )
            .setText( "7411760055" ); // ssid=10225
        janelaCliente1.textBox( "celular" )
            .setText( "3614157825" ); // ssid=10226
        janelaCliente1.button( "adicionar" )
            .click(); // ssid=10227

        DialogFixture janelaAnimal1 = findJanelaAnimal();
        janelaAnimal1.show(); // ssid=10228
        janelaAnimal1.textBox( "nome" )
            .setText( "#" ); // ssid=10229
        janelaAnimal1.comboBox( "sexo" )
            .selectItem( "MACHO" ); // ssid=10230
        janelaAnimal1.textBox( "nascimento" )
            .setText( "25/01/0001" ); // ssid=10231
        janelaAnimal1.comboBox( "especie" )
            .selectItem( "ROEDOR" ); // ssid=10232
        janelaAnimal1.textBox( "raca" )
            .setText( "$=@dV==w+|*u*/%]5ItZBEK
                RN(8gb?E=t UMDhtCLS=8>E" ); // ssid=10233
        janelaAnimal1.textBox( "info" ).setText(
            "o!/}PF1+zpad8;fG}JS49We2+@H*2)WJ^zb
            lsZ_%,Ek>=8bzR6{PEt8n{4@Z0s'6_f7?CTx[
            0A$/jaXOG%,BsNYY(]68:oU?x^zZu;Z=tQ3fY
            mW0M<qW#(?]I%NI=4ASPK%M+|o>s=xgAf`7by
            lA5]OqVV24$XyQj0U%I-zc1:pFD!]Q>:sx#{2
            DKQ|Z13hNUv0s6:<DlR(g;^ITGgOuqVm|+X5Q
            ZIYAuVEK'VjxY|)+<V8] o5VyVfz'D?-tN,3" ); // ssid=10234
```

```
        janelaAnimal1.button( "adicionar" ).click(); // ssid=10235

        janelaCliente1.textBox( "info" ).setText(
            ",nN(m%jAuvLR++PIU@BY]i5)IEQo7sc=`XW:
            :TmT*/S3OgnhOR>3<&D+8.E3484^Vg[q*jC%R
            +K{4LT!!`!<9?H!,mLW%b9cU^m+yeDs}14aX?
            4z<Sd.B;7x%(TDxvO-Eo&<=yWb[+QjGd8//_Uu
            |b}G~ko=UiN5^T0 !!t/.8D:)3_sAz6@28G!mD
            h={;CM_].V@_*T!MWAO/{)FAY5AF" ); // ssid=10237
        janelaCliente1.button( "cadastrar" ).click(); // ssid=10238

    }
```

Listagem 18 - Código-fonte do teste onde a segunda falha exemplificada foi observada

Ao se investigar a causa do problema no SST, descobriu-se que o valor usado para o campo gerou uma exceção no programa (`java.lang.NumberFormatException`), que foi apresentada *apenas na saída para o console*. Esta exceção fez com que o comportamento do SST ficasse diferente do esperado pelo teste, fazendo este acusar a falha. Ou seja, o SST na verdade esperava um valor numérico para o campo, mas não verificou o formato do valor fornecido, o que gerou a exceção. Novamente foi constatada uma falha no SST devido a um valor gerado pela ferramenta.

6.7.9.1.Falhas por cenário

A Tabela 21 apresenta a contabilização das falhas por cenário. Ela será útil na comparação com as próximas versões da especificação e do SST.

Tabela 21 - Falhas por cenário

Cenário	Falhas
Efetuar Login – BF	4
Cadastrar Cliente – BF	35
Cadastrar Cliente – BF, AF1, BF	37
Total	**76**

Como os casos de uso *Adicionar Animal* e *Remover Animal* estão inclusos nos cenários de execução do caso de uso *Cadastrar Cliente*, as falhas destes estão embutidas nessa contagem.

6.8.Versão levemente divergente da especificação

Como mencionado na seção 6.4, uma versão da especificação levemente divergente da descrita na seção 6.7 foi construída, para verificar se esta diferença seria observada pelos testes gerados pela ferramenta.

A seguir são apresentadas apenas as divergências em relação à outra especificação, para facilitar sua compreensão.

6.8.1.Caso de Uso Efetuar Login

As alterações na especificação do caso de uso foram:

1. Alteração da Regra de Negócio 2 (RN2), para que o comprimento máximo do "nome de usuário" seja de 20 caracteres e seja exibida a mensagem "Por favor, informe um nome de usuário com até 20 caracteres.", caso seja informado um valor de comprimento maior;
2. Alteração da Regra de Negócio 3 (RN3), para que o comprimento máximo da "senha" seja de 20 caracteres e seja exibida a mensagem "Por favor, informe uma senha com até 20 caracteres.", caso seja informado um valor de comprimento maior.

6.8.2.Caso de Uso Acessar Sistema

Nenhuma mudança de especificação foi realizada para este caso de uso.

6.8.3.Caso de Uso Cadastrar Cliente

As alterações na especificação do caso de uso foram:

1. Alteração da Regra de Negócio 4 (RN4), para que o comprimento máximo do "nome" seja de 50 caracteres e seja exibida a mensagem "Por favor, informe um nome com até 50 caracteres.", caso seja informado um valor de comprimento maior;
2. Alteração da Regra de Negócio 15 (RN15), para que o e-mail possua o

formato baseado na seguinte expressão regular "[A-Za-z0-9_-]+@([A-Za-z0-9_]+\.)+[A-Za-z]{2,4}", exibindo a mensagem "Por favor, informe um e-mail válido.", caso seu formato esteja correto.

6.8.4.Caso de Uso Adicionar Animal

As alterações na especificação do caso de uso foram:

1. Alteração da Regra de Negócio 2 (RN2), para que o comprimento mínimo do "nome" seja de 2 caracteres e seja exibida a mensagem "Por favor, informe um nome com pelo menos 2 caracteres.", caso seja informado um valor de comprimento menor;

6.8.5.Caso de Uso Remover Animal

Nenhuma mudança de especificação foi realizada para este caso de uso.

6.8.6.Geração dos testes

Para a especificação levemente divergente, a ferramenta gerou 204 CTSVO num arquivo JSON com 200.428 linhas e 6,06 MB. Esta geração, considerando a execução dos algoritmos de geração dos CTSVO e a gravação do arquivo em disco, durou 8 segundos.

6.8.7.Execução dos testes

A Tabela 22 apresenta os tempos envolvidos na execução dos testes.

Tabela 22 - Tempos envolvidos na execução dos testes para a versão com especificação levemente divergente

Número de testes	204
Tempo de execução	18m29s
Tempo médio por teste	5s (5439ms)
Teste mais lento	6s (6631ms)
Teste mais rápido	0s (340ms)

Os mesmos parâmetros de execução foram utilizados, em relação à especificação original.

6.8.8.Transformação dos resultados de execução

A pré-análise e a transformação dos resultados de execução para o formato padrão da ferramenta ocorreram, juntas, em 791 milissegundos. O processo gerou o arquivo `results.json`, contendo 7.415 linhas e 684 KB.

6.8.9.Análise dos resultados

Dos 204 testes executados, 85 falharam (41,67%) e nenhum obteve erro. Com as 5 regras de negócio adicionais, não atendidas pelo SST, foram detectadas 9 falhas adicionais.

6.8.9.1.Falhas por cenário

Para efeitos comparativos, a Tabela 23 apresenta as falhas encontradas por cenário:

Tabela 23 - Falhas por cenário na versão levemente divergente

Cenário	Falhas
Efetuar Login – BF	10
Cadastrar Cliente – BF	40
Cadastrar Cliente – BF, AF1, BF	35
Total	**85**

Os casos de uso *Adicionar Animal* e *Remover Animal* estão inclusos em *Cadastrar Cliente*, por serem executados a partir dele.

6.9.Versões do SST com mutantes

Conforme mencionado na seção 6.4, foram criadas duas versões do SST com uso de mutantes, para avaliar se a ferramenta é capaz de observá-los.

6.9.1.Primeiro SST mutante

Nesta primeira versão mutante do SST, foi utilizado o operador "**Substituição de regras de validação ou da admissão de um dado como incorreto**", apresentado na Tabela 5 (seção 3.2). Para implementá-lo, foram invertidos os operadores condicionais que realizavam a validação dos valores. A Tabela 24, a seguir, apresenta o número de mutações por caso de uso.

Tabela 24 - Mutações realizadas por caso de uso, para o primeiro SST mutante

Caso de uso	Mutações realizadas
Efetuar Login	2
Cadastrar Cliente com Remover Animal	14
Adicionar Animal	6
Acessar Sistema	0
Total	**22**

Para a realização dos testes, **um caso de uso foi modificado por vez**, para não fazer com que a execução dos casos de uso *relacionados* ao caso de uso alvo do teste falhasse, impedindo ou influenciando sua execução.

6.9.1.1.Execução com Adicionar Animal mutante

A execução dos 204 testes sobre o caso de uso *Adicionar Animal* mutante resultou em 168 falhas (82,35%) e nenhum erro. A Tabela 25, a seguir, apresenta a distribuição das falhas por cenário.

Tabela 25 - Distribuição das falhas por cenário usando Adicionar Animal com o primeiro mutante

Cenário	Falhas
Efetuar Login – BF	4
Cadastrar Cliente – BF	82
Cadastrar Cliente – BF, AF1, BF	82
Total	**168**

A Tabela 26, a seguir, apresenta os tempos envolvidos na execução dos testes.

Tabela 26 - Tempos de execução para os testes usando Adicionar Animal com o primeiro mutante

Número de testes	204
Tempo de execução	18m37s
Tempo médio por teste	5s (5475ms)
Teste mais lento	7s (7340ms)
Teste mais rápido	0s (210ms)

Pode-se observar que os *tempos de execução quase não mudaram*, mas o número total de falhas cresceu de 76 (da versão original) para 168, uma diferença de 102 testes (121%). O cenário "*Efetuar Login – BF*" que não possui relação com Adicionar Animal, *não* foi afetado pela mudança (conforme esperado). Para "*Cadastrar Cliente – BF*", o número de falhas aumentou de 35 para 82. E para "*Cadastrar Cliente – BF, AF1, BF*", de 37 para 82. Ou seja, com apenas 6 mutantes introduzidos no caso de uso *Cadastrar Cliente*, obteve-se uma *média* de 46 novas falhas detectadas.

6.9.1.2.Execução com Cadastrar Cliente mutante

A execução dos 204 testes sobre o caso de uso *Cadastrar Cliente* mutante resultou em 77 falhas (37,75%) e nenhum erro. A Tabela 27, a seguir, apresenta a distribuição das falhas por cenário.

Tabela 27 - Distribuição das falhas por cenário usando Cadastrar Cliente com o primeiro mutante

Cenário	Falhas
Efetuar Login – BF	4
Cadastrar Cliente – BF	36
Cadastrar Cliente – BF, AF1, BF	37
Total	**77**

A Tabela 28, a seguir, apresenta os tempos envolvidos na execução dos testes.

Tabela 28 - Tempos de execução para os testes usando Cadastrar Cliente com o primeiro mutante

Número de testes	204
Tempo de execução	18m31s
Tempo médio por teste	5s (5448ms)
Teste mais lento	6s (6808ms)
Teste mais rápido	0s (230ms)

Pode-se observar que tanto os tempos de execução quanto o número de falhas permaneceram estáveis. O cenário "*Efetuar Login – BF*" que não possui relação com Adicionar Animal, *não* foi afetado pela mudança (conforme esperado). Para "*Cadastrar Cliente – BF*", o número de falhas aumentou de 35 para 36. E para "*Cadastrar Cliente – BF, AF1, BF*", mantiveram-se as 37 falhas.

6.9.1.3.Execução com Efetuar Login mutante

A execução dos 204 testes sobre o caso de uso *Efetuar Login* mutante resultou em 192 falhas (94,12%) e nenhum erro. A Tabela 29 apresenta a distribuição das falhas por cenário.

Tabela 29 - Distribuição das falhas por cenário usando Efetuar Login com o primeiro mutante

Cenário	Falhas
Efetuar Login – BF	8
Cadastrar Cliente – BF	92
Cadastrar Cliente – BF, AF1, BF	92
Total	**192**

A Tabela 30, a seguir, apresenta os tempos envolvidos na execução dos testes.

Tabela 30 - Tempos de execução para os testes usando Efetuar Login com o primeiro mutante

Número de testes	204
Tempo de execução	18m58s
Tempo médio por teste	5s (5582ms)
Teste mais lento	5s (5939ms)
Teste mais rápido	0s (350ms)

Pode-se observar que os tempos de execução se mantiveram estáveis e o número de falhas passou de 76 para 192 (aumento de 152%). Entretanto, todos os (12) testes que tiveram sucesso pertencem ao caso de uso *Efetuar Login*, o que indica que a modificação no SST gerou uma falha no cenário de execução, afetando diretamente os casos de uso dependentes dele, fato este já previsto. É interessante observar, todavia, que o número de falhas no caso de uso *Efetuar Login* dobrou (de 4 para 8) com as duas mutações introduzidas.

6.9.1.4.Resultado consolidado

A Tabela 31 apresenta os resultados de execução consolidados, para o primeiro mutante.

Tabela 31 - Resultados consolidados para o primeiro mutante

Cenário	Falhas no SST Original	Falhas		
		Adicionar Animal	Cadastrar Cliente	Efetuar Login
Efetuar Login – BF	**4**	*4*	*4*	*8*
Cadastrar Cliente – BF	**35**	*82*	*36*	92
Cadastrar Cliente – BF, AF1, BF	**37**	*82*	*37*	92
Total	**76**	**168**	**77**	**192**

Para o cenário "*Efetuar Login – BF*", pode ser considerado apenas o resultado da versão de mutante de *Efetuar Login*, uma vez que ele não sofre influência dos demais casos de uso. Tem-se, então, que com a introdução de 2 mutantes, a

ferramenta observou 4 falhas adicionais.

Para o cenário "*Cadastrar Cliente – BF*", pode ser desconsiderado o resultado da versão mutante de *Efetuar Login*, uma vez que foi constatada que as falhas geradas neste impediram a execução do cenário. A adição de 14 mutantes em *Cadastrar Cliente* gerou apenas 1 falha adicional no cenário, enquanto a adição de 6 mutantes em *Adicionar Animal* gerou 47 falhas adicionais.

Para o cenário "*Cadastrar Cliente – BF, AF1, BF*", também é desconsiderada a versão mutante de *Efetuar Login*, de forma que as 14 mutações em *Cadastrar Cliente* não influenciaram o número de falhas, enquanto as 6 mutações em *Adicionar Animal* geraram 45 falhas adicionais.

É possível concluir, então, que os testes gerados pela ferramenta foram capazes de observar as falhas em decorrência das mutações realizadas.

6.9.2.Segundo SST mutante

Nesta segunda versão mutante do SST, foi utilizado o operador "**Informação incorreta ou incompleta mostrada pelo sistema**", apresentado na Tabela 5 (seção 3.2). Para implementá-lo, foram realizadas modificações nas mensagens exibidas pelo sistema, para o usuário final. A Tabela 32 apresenta o número de mutações por caso de uso.

Tabela 32 - Mutações realizadas por caso de uso, para o segundo SST mutante

Caso de uso	Mutações realizadas
Efetuar Login	4
Cadastrar Cliente com Remover Animal	4
Adicionar Animal	2
Acessar Sistema	0
Total	**10**

Novamente, para a realização dos testes, **um caso de uso foi modificado por vez**, para não fazer com que a execução dos casos de uso *relacionados* ao caso de

uso alvo do teste falhasse, impedindo ou influenciando sua execução.

6.9.2.1.Execução com Adicionar Animal mutante

A execução dos 204 testes sobre o caso de uso *Adicionar Animal* mutante resultou em 92 falhas (45,10%) e nenhum erro. A Tabela 33 apresenta a distribuição das falhas por cenário.

Tabela 33 - Distribuição das falhas por cenário usando Adicionar Animal com o segundo mutante

Cenário	Falhas
Efetuar Login – BF	4
Cadastrar Cliente – BF	42
Cadastrar Cliente – BF, AF1, BF	46
Total	**92**

A Tabela 34, a seguir, apresenta os tempos envolvidos na execução dos testes.

Tabela 34 - Tempo de execução para os testes usando Adicionar Animal com o segundo mutante

Número de testes	204
Tempo de execução	18m28s
Tempo médio por teste	5s (5433ms)
Teste mais lento	6s (6819ms)
Teste mais rápido	0s (330ms)

Pode-se observar que os tempos de execução quase não mudaram, mas o número total de falhas cresceu de 76 (da versão original) para 92, uma diferença de 16 testes (21%). O cenário "*Efetuar Login – BF*" que não possui relação com Adicionar Animal, *não* foi afetado pela mudança (conforme esperado). Para "*Cadastrar Cliente – BF*", o número de falhas aumentou de 35 para 42. E para "*Cadastrar Cliente – BF, AF1, BF*", de 37 para 46. Ou seja, com apenas 2 mutantes introduzidos no caso de uso *Cadastrar Cliente*, obteve-se uma *média* de 8 novas falhas detectadas.

6.9.2.2.Execução com Cadastrar Cliente mutante

A execução dos 204 testes sobre o caso de uso *Cadastrar Cliente* mutante resultou em 79 falhas (38,73%) e nenhum erro. A Tabela 35 apresenta a distribuição das falhas por cenário.

Tabela 35 - Distribuição das falhas por cenário usando Cadastrar Cliente com o segundo mutante

Cenário	Falhas
Efetuar Login – BF	4
Cadastrar Cliente – BF	36
Cadastrar Cliente – BF, AF1, BF	39
Total	**79**

A Tabela 36, a seguir, apresenta os tempos envolvidos na execução dos testes.

Tabela 36 - Tempos de execução para os testes usando Cadastrar Cliente com o segundo mutante

Número de testes	204
Tempo de execução	18m27s
Tempo médio por teste	5s (5427ms)
Teste mais lento	6s (6628ms)
Teste mais rápido	0s (340ms)

Pode-se observar que tanto os tempos de execução quanto o número de falhas permaneceram estáveis. O cenário "*Efetuar Login – BF*" que não possui relação com Adicionar Animal, *não* foi afetado pela mudança (conforme esperado). Para "*Cadastrar Cliente – BF*", o número de falhas aumentou de 35 para 36. E para "*Cadastrar Cliente – BF, AF1, BF*", aumentou de 37 para 39.

6.9.2.3.Execução com Efetuar Login mutante

A execução dos 204 testes sobre o caso de uso *Efetuar Login* mutante resultou

em 82 falhas (40,20%) e nenhum erro. A Tabela 37 apresenta a distribuição das falhas por cenário.

Tabela 37 - Distribuição das falhas por cenário usando Efetuar Login com o segundo mutante

Cenário	Falhas
Efetuar Login – BF	10
Cadastrar Cliente – BF	33
Cadastrar Cliente – BF, AF1, BF	39
Total	**82**

A Tabela 38, a seguir, apresenta os tempos envolvidos na execução dos testes.

Tabela 38 - Tempos de execução para os testes usando Efetuar Login com o segundo mutante

Número de testes	204
Tempo de execução	18m27s
Tempo médio por teste	5s (5429ms)
Teste mais lento	6s (6737ms)
Teste mais rápido	0 (360ms)

Pode-se observar, novamente, que os tempos de execução se mantiveram estáveis. Quanto ao número de falhas, passou de 76 para 82. O número de falhas no cenário "*Efetuar Login – BF*" passou de 4 para 10 com as 4 mutações introduzidas. Curiosamente, as falhas de "*Cadastrar Cliente – BF*" diminuíram em 2 unidades, fato este que pode ter sido obtido por algum teste anterior deste cenário ter falhado no processo de *login*, que estava diferente da especificação, mas que, com o mutante, funcionou como esperado. Para o cenário "*Cadastrar Cliente – BF, AF1, BF*", 2 falhas foram adicionadas.

6.9.2.4.Resultado consolidado

A Tabela 39, a seguir, apresenta os resultados de execução anteriormente informados de forma consolidada.

Tabela 39 - Resultado consolidado de execução para o segundo mutante

Cenário	Falhas no SST Original	Falhas		
		Adicionar Animal	Cadastrar Cliente	Efetuar Login
Efetuar Login – BF	**4**	*4*	*4*	*10*
Cadastrar Cliente – BF	**35**	*42*	*36*	*33*
Cadastrar Cliente – BF, AF1, BF	**37**	*46*	*39*	*39*
Total	**76**	**92**	**79**	**82**

Para o cenário "*Efetuar Login – BF*", pode-se considerar apenas o resultado da versão de mutante de *Efetuar Login*, uma vez que ele não sofre influência dos demais casos de uso. Tem-se, então, que com a introdução de 4 mutantes, a ferramenta observou 6 falhas adicionais.

Para o cenário "*Cadastrar Cliente – BF*", as 2 mutações de Adicionar Animal e as 4 mutações de Cadastrar Cliente, provocaram 6 falhas adicionais, com uma média de 2 falhas introduzidas.

Para o cenário "*Cadastrar Cliente – BF, AF1, BF*", as mesmas mutações descritas anteriormente, provocaram 13 falhas adicionais, com uma média de 4 falhas introduzidas.

É possível concluir que os testes gerados pela ferramenta foram capazes de observar as falhas em decorrência das mutações realizadas.

6.9.3.Análise dos resultados das versões mutantes

Apesar de não ter sido realizada uma análise mais detalhada sobre quais mutantes especificamente foram mortos, além da detecção de eventuais mutantes equivalentes, o número de falhas geradas com a introdução dos mutantes permite intuir que a ferramenta consegue detectar eficazmente diferenças de implementação em relação à especificação, que é seu objetivo principal.

Futuramente pretende-se realizar um estudo mais detalhado, possivelmente com uso de alguma instrumentação que permita reduzir o (elevado) esforço de análise dos mutantes, para que a pontuação (*score*) resultante possa ser obtida,

concedendo um maior grau de certeza sobre os resultados.