

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Geração automática de casos de teste a partir da utilização de SaaS

Pedro Miguel Vieira da Silva



Mestrado Integrado em Engenharia Informática e Computação

Orientador: Ana Paiva

Co-orientador: André Restivo

23 de Julho de 2018

Geração automática de casos de teste a partir da utilização de SaaS

Pedro Miguel Vieira da Silva

Mestrado Integrado em Engenharia Informática e Computação

Abstract

Nowadays many web applications play a very important role in our society and in the business world. Much of those companies earn large part of their revenues due to their web applications that provide support services that must be maintained and improved over time. Most of these services operate on a large scale and are in constant change due to the environment in which they operate and due to the rapid technological evolution that is always trying to find new ways to improve our lives.

Due to these constant changes, it is difficult to estimate the impact of these changes and to maintain the software requirements documents updated. To estimate this impact, it is highly necessary to have requirements traceability information and a good test methodology so that we can analyze the evolution and consistency of our software. This information is not always present in the companies files sometimes because of documents disorder or because of not so well safekeeping information which makes the process of updating requirements very difficult.

On the other hand, those changes also produce a negative impact on the testing process, since regular alterations to the developed software lead to changes in the developed test cases which makes it harder to produce consistent and reliable test cases.

Regression testing is an essential part of the quality process and ensures that code changes do not hurt the existing functionality. Regression testing is a type of software testing that ensures that previously developed and tested software still performs the same way after it is changed or interfaced with other software. Effective regression testing can save a company's time and money. It should become a routine procedure while developing an application. Regression testing includes executing an increasing set of tests along with covering existing functionality until the product is done. Continuous regression testing help teams build software that behaves as intended and remains stable.

The main goal of this work is to extract a set of regression tests from web usage to automate the process of maintaining and validating software requirements. It is also intended to study the reliability of the test cases obtained in order to verify the quality of the results.

During the experiment three cases studies were conducted in three different systems, being the ERASMUS and Multimedia sections of the IPVC Portal and also the Newspaper of the City of Tomar.

Firstly, it was concluded that it was possible to extract test cases from existing navigation data with relative ease. However, reliability and consistency were not expected, due in large part, to the poor quality of the navigation data provided.

Resumo

Hoje em dia, imensas aplicações *web* desempenham um papel crucial na sociedade e no mundo dos negócios. Muitas empresas obtêm grande parte das suas receitas graças às suas aplicações *web* que oferecem serviços de suporte que são constantemente mantidos e melhorados ao longo do tempo. A maioria destes serviços opera em grande escala e está em constante mudança, devido ao ambiente em que se inserem e à rápida evolução tecnológica.

Com estas mudanças é difícil estimar o impacto que estas têm e manter os documentos de requisitos de *software* atualizados. Para estimar esse impacto é altamente necessário obter informações de rastreabilidade de requisitos e aplicar uma boa metodologia de teste, de forma a ser possível analisar a evolução e a consistência do *software*. Esta informação nem sempre está presente nos arquivos das empresas, muitas vezes, devido a desorganização de documentos ou devido a informações de segurança tão boas que tornam muito difícil o processo de atualização de requisitos.

Por outro lado, estas mudanças têm também um impacto bastante negativo no processo de teste de *software*. Alterações constantes ao sistema levam a que seja necessário proceder, também, a mudanças nos testes desenvolvidos, dificultando assim a tarefa de produção de testes com a consistência e fiabilidade desejadas.

Os testes de regressão possuem um papel importante no processo de garantia de fiabilidade e consistência de um sistema. Um teste de regressão pode ser definido como um tipo de teste que assegura que o *software*, previamente desenvolvido e testado, continua a executar devidamente, após terem sido efetuadas novas alterações ao sistema. A execução deste tipo de testes tem como objetivo verificar se as alterações efetuadas produzem efeitos negativos em alguma das componentes do sistema desenvolvido.

O principal objetivo deste trabalho de investigação é conseguir extrair uma bateria de testes de regressão a partir dos dados navegação recolhidos, de forma a automatizar o processo de manutenção e validação de requisitos em sistemas de *software*.

Pretende-se também estudar a fiabilidade dos casos de teste obtidos afim de verificar a qualidade dos resultados.

Durante a realização desta experiência foram conduzidos três casos de estudo a três sistemas diferentes, sendo eles as secções de ERASMUS e Multimédia do Portal do IPVC e também o *Newspaper* da Cidade de Tomar.

Em primeiro lugar, concluiu-se que foi possível extrair casos de teste através dos dados de navegação existentes com relativa facilidade. No entanto, a fiabilidade e consistência dos mesmos não foi a esperada, devido em muito, à fraca qualidade dos dados de navegação fornecidos.

Agradecimentos

Em primeiro lugar, gostaria de dedicar este trabalho ao meu avô, que infelizmente já não se encontra entre nós. De seguida, aos meus pais e à minha avó, por todo trabalho e tempo que investiram em mim, pelos sacrifícios que fizeram para que eu pudesse ter uma educação deste nível e por todo o apoio que sempre me deram ao longo do tempo.

De seguida, queria agradecer à Professora Ana Paiva e ao Professor André Restivo por todo o apoio prestado ao longo deste trabalho de Dissertação.

Posteriormente, uma grande palavra de apreço a todos os meus amigos, por todos os momentos memoráveis que passamos juntos e por todas estas grandes amizades que levo comigo para a vida e guardo bem junto ao coração. Sem vocês não teria tido a mesma piada que teve obrigado por tudo malta.

Por último, um grande obrigado à Cristiana, por ter estado lá para mim sempre que foi preciso. Sem ti, de certeza que não teria feito um trabalho deste nível e um grande obrigado por todo apoio prestado neste trabalho e por tudo o que és para mim.

Pedro Silva

“If there is no struggle, there is no progress.”

Frederick Douglass

Conteúdo

1	Introdução	1
1.1	Contexto/Enquadramento	1
1.2	Motivação e Objetivos	3
1.2.1	Estrutura do Documento	5
2	Revisão Bibliográfica	7
2.1	Introdução	7
2.2	Engenharia de Requisitos	7
2.2.1	Tipo de Requisitos	8
2.2.2	Etapas do Processo de Engenharia de Requisitos	9
2.2.3	Rastreabilidade de Requisitos	11
2.3	<i>Web Analytics</i>	15
2.3.1	Ferramentas	15
2.4	Sistemas de Recomendação	17
2.4.1	Ferramentas Existentes	18
2.5	Testes de <i>Software</i>	21
2.5.1	<i>Model-Based Testing</i>	21
2.5.2	Testes de Regressão	23
2.5.3	<i>Codeception</i>	25
3	Proposta de Solução	29
3.1	Definição do Problema	29
3.2	Abordagem Proposta	30
3.2.1	Tecnologias	32
3.2.2	Casos de Estudo	32
4	Implementação	33
4.1	Recolha dos <i>logs</i> do OWA	33
4.1.1	Estrutura da Base de Dados	33
4.2	Seleção e Ordenação dos Dados	34
4.3	Geração do <i>Script</i> de Testes	35
4.3.1	Execução do <i>Script</i> de Testes	40
5	Resultados	43
5.1	Problema Estudado	43
5.1.1	Especificação Técnica	44
5.1.2	Metodologia de Teste	44
5.2	Casos de Estudo	47

CONTEÚDO

5.2.1	Instituto Politécnico de Viana do Castelo	47
5.2.2	Multimédia IPVC	48
5.2.3	<i>Newspaper</i> Cidade de Tomar	49
5.3	Conclusões	50
6	Discussão e Conclusão	51
6.1	Discussão	51
6.1.1	Q1 - É possível extrair casos de teste a partir de informação de dados de navegação <i>web</i> ?	51
6.1.2	Q2 - Será a informação recolhida pelo OWA suficiente para produzir casos de teste consistentes?	51
6.2	Conclusão e Trabalho Futuro	52
	Referências	53
A	<i>Scripts</i> de Teste	57
A.1	Caso de Estudo I - Portal de Erasmus do IPVC	57
A.2	Caso de Estudo II - Portal de Multimédia do IPVC	65
A.3	Caso de Estudo III - <i>Newspaper</i> da Cidade de Tomar	73

Lista de Figuras

2.1	Fases do processo de Engenharia de Requisitos	9
2.2	Arquitetura da Ferramenta <i>REQAnalytics</i>	21
2.3	Sequência de passos de <i>Module-Based Testing</i>	22
2.4	Arquitetura da Ferramenta <i>Codeception</i>	28
2.5	Exemplo de Relatório de Execução	28
3.1	Fases da Implementação	31
4.1	Estrutura da Base de Dados	34
4.2	Interface de Testes	35
4.3	Exemplo de informação sobre Inputs	36
4.4	Exemplo de informação sem Inputs	37
4.5	Exemplo de formulário de introdução de informação	39
4.6	Exemplo de formulário só com verificação final	40
4.7	Ambiente de Teste	41
5.1	Extrato das Ocorrências de cada id de utilizador no <i>site</i> de Multimédia	45
5.2	Exemplo de uma amostra válida	46
5.3	Exemplo de uma amostra inválida	46

LISTA DE FIGURAS

Lista de Tabelas

5.1	Pesquisa de sessões válidas	47
5.2	Pesquisa de sessões válidas	48
5.3	Pesquisa de sessões válidas	49
5.4	Comparação de Resultados	50

LISTA DE TABELAS

Abreviaturas e Símbolos

BDD	Behavior-Driven Development
DSL	Domain Specific Language
ER	Engenharia de Requisitos
FDA	Food Drug Administration
KPI	Key Performance Indicator
MTTR	Mean Time To Repair
OWA	Open Web Analytics
TDD	Test-Driven Development

Capítulo 1

Introdução

Normalmente, obter requisitos consistentes e fiáveis de *stakeholders* é considerada uma tarefa árdua, dispendiosa e muito suscetível a erros [KGH10]. É, portanto, necessário validar requisitos de *software* o mais cedo possível no processo de desenvolvimento de *software*, de maneira a detetar e prevenir erros na especificação de requisitos [KNG⁺14]. Esta validação permite produzir *software* com uma garantia de qualidade elevada, o que permite assim uma posição de destaque perante a concorrência [GP16b].

1.1 Contexto/Enquadramento

Hoje em dia, no quotidiano em que nos inserimos, a Qualidade do Software é um dos tópicos de maior preocupação e que cada vez mais atenção merece da nossa parte. Vivemos num mundo em constante expansão tecnológica e cada vez mais dependente da Internet. Diariamente surgem novas oportunidades tecnológicas que nos permitem melhorar a nossa qualidade de vida e nos ajudam a expandir os nossos projetos de negócio.

Particularmente, neste último aspeto citado, os *websites* desempenham um papel essencial, visto que, atualmente, são bastante utilizados como principais ferramentas de suporte a vários projetos de negócio, essencialmente, através de serviços e aplicações web [Som04]. Este tipo de *software* tem de se encontrar sempre disponível para ser utilizado e ser capaz de cumprir com todos os requisitos que a equipa de desenvolvimento acordou com o cliente e, além disso, tem de ser um *software* capaz de se adaptar a futuras mudanças e novas exigências que alterações no ambiente em que o projeto de negócio está inserido possam proporcionar. No entanto, variadas vezes, encontramos projetos de *software* que não cumprem com os requisitos a que se propuseram e que, por isso, entram num constante ciclo de sucessivos fracassos que acabam por conduzir ao fim desse mesmo projeto de negócio. As principais razões para o fim antecipado de vários projetos de *software* são:

- Numa primeira fase, uma má elicitação de requisitos que, por sua vez, conduz a uma má compreensão das funcionalidades que o cliente pretende que o *software* execute;

Introdução

- Falhas na comunicação entre elementos da equipa que não reportam quando ocorre alguma falha nos requisitos aos seus superiores e restantes elementos de equipa, o que, por sua vez, conduz a uma ineficiente gestão dos requisitos do projeto.

Estas falhas na gestão de requisitos têm um impacto bastante negativo no desenvolvimento de *software*, pois causam falhas no sistema que implicam fortes investimentos, tanto financeiros, como tecnológicos, os quais são utilizados para contornar este tipo de problemas que, se não forem solucionados, geram um sistema de *software* incapaz de responder as necessidades dos seus *stakeholders*.

Um requisito pode ser definido como a descrição dos serviços que um sistema de *software* possui e as restrições segundo as quais o mesmo terá de operar [Ban11]. Estes mesmos requisitos podem ser definidos por um grande rol de papéis que podem variar, desde descrições abstratas de funcionalidades do sistema, até especificações detalhadas de funções matemáticas específicas.

Os requisitos podem ser classificados como funcionais ou não funcionais. Os primeiros definem os serviços que o sistema deve prestar, como este deve reagir a certos *inputs* exteriores e como se deve comportar em certos casos de uso específicos. Por outro lado, um requisito não funcional define as restrições de interoperabilidade do sistema, tais como, restrições temporais ou espaciais, restrições no processo de desenvolvimento de *software*, entre outras [CoE]. O processo que permite a recolha de requisitos e os implementa ao longo de todas as fases do ciclo de vida do *software* denomina-se Engenharia de Requisitos (ER) [McF].

Esta área de investigação está intrinsecamente ligada aos objetivos e Qualidade do Software produzido, tentando sempre dar resposta às perguntas *Como irá o sistema funcionar?* e *Que qualidades terão de ter as propriedades do sistema para produzir os resultados pretendidos?*. É importante também referir que um dos objetivos deste processo é interagir e conhecer mais aprofundadamente as intenções e objetivos dos seus *stakeholders* para o projeto. Um *stakeholder* pode ser definido como "*um indivíduo, grupo ou organização que pode afetar, ser afetado por uma decisão, atividade ou resultado de um projeto em que está inserido*"[SFG99].

Normalmente, a equipa de desenvolvimento do projeto molda o desenvolvimento do *software* de acordo com os requisitos do cliente. Isto significa que a maior parte das etapas da Engenharia de Requisitos decorrem durante os atos de comunicação ente clientes e equipa. Este processo de Engenharia de Requisitos é considerado um processo iterativo que decorre em várias fases, estando cada uma das fases intrinsecamente ligada à fase que a precede.

A ER foca-se bastante no processo de análise de requisitos do sistema e, numa primeira fase, tem como foco principal as pessoas e corporações que fazem parte do âmbito do projeto [Som07]. No entanto, e tal como já foi referido anteriormente, a Engenharia de Requisitos expõe processos de design, prototipagem e validação bastante iterativos, apesar de que o design não é visto como um dos focos principais deste processo [Som07].

Durante o ciclo de vida do software/sistema tecnológico de serviços, a ER permite definir, reconhecer, modelar, ligar, documentar e manter os requisitos de *software*, de forma a reconhecer possíveis problemas que o possam afetar e apoiar na sua resolução, sendo assim importante acompanhar e monitorizar em tempo real as mudanças dos requisitos. À medida que o tempo passa,

estes tornam-se mais complexos, detalhados e concretos. A documentação de design criada inicialmente pode ser deteriorada e inutilizada, já que esta fica desatualizada e não reflete corretamente o estado funcional do sistema. Estas mudanças devem ser tratadas de forma a atualizar o estado do sistema de acordo com a fase de desenvolvimento do projeto e localização de outros artefactos de desenvolvimento. Ocorrem, porque os sistemas de serviços tecnológicos não existem sem um contexto que é determinado pelas necessidades do Ser Humano. Reconhecendo que os sistemas de serviços tecnológicos têm que acompanhar a evolução cíclica, impulsionada pelo meio ambiente dos requisitos de sistemas sujeitos a pressões adaptativas, é necessário usar modelos práticos de requisitos que possam ser aplicados durante o ciclo de vida do *software* [CHCH10].

Durante o ciclo de vida de desenvolvimento de *software*, novos requisitos emergem e os existentes sofrem variadas alterações. Foi observado que na maioria dos projetos de *software* mais de metade dos requisitos de sistema sofrem modificações ainda antes de o sistema ser lançado para execução [Gro95].

Estes dados indicam que tantas alterações nos requisitos podem causar graves problemas durante o desenvolvimento do sistema e os mesmos podem ter um grande impacto na qualidade do *software* desenvolvido. Esta gestão de requisitos permite manter estabilidade e acordo entre *stakeholders*, através de uma análise cuidadosa aos impactos que estas alterações terão na estabilidade do sistema.

Porém, durante o ciclo de vida de desenvolvimento de *software*, torna-se muito difícil determinar quais os pedidos de alterações a requisitos que devem receber principal atenção e, portanto, devido a todos os fatores supra numerados, muitas vezes a relação existente entre a rastreabilidade de requisitos e a sua implementação perde-se [Gro95].

De forma a combater este problema, foram criados vários métodos de análise e coleção de dados durante as várias fases do ciclo de vida de um projeto de *software*. Estes métodos são denominados de *Web Analytics* e têm como principal objetivo providenciar dados concretos aos utilizadores sobre a qualidade do seu *software*.

1.2 Motivação e Objetivos

Atualmente, mudanças tardias e especificações desatualizadas de requisitos têm um impacto bastante negativo no processo de desenvolvimento de *software* que, em alguns casos, pode levar ao colapsar do sistema [Gro95].

Quanto mais tarde no processo de desenvolvimento de *software* se procede à alteração ou inserção de requisitos, mais esforço, tempo e dinheiro será requerido à equipa de desenvolvimento para impedir que estas alterações tenham impacto negativo no projeto.

É, por isso, importante num contexto de desenvolvimento de sistema de *software*, proceder a uma análise constante dos requisitos durante o ciclo de vida de um sistema de *software*, visto que esta análise pode fornecer dados extremamente relevantes para a gestão e manutenção de requisitos, como por exemplo, sugerir prioritização ou novas mudanças no sistema de *software*. Neste seguimento, caso a análise indicar que um dos requisitos é mais utilizado pelos utilizadores

do produto, pode sugerir-se uma alteração na prioridade do mesmo, tornando-a mais elevada do que outro requisito menos utilizado.

Atualmente, existem vários métodos de análises de requisitos, no entanto, de momento as *Web Analytics Tools* são consideradas o método de análise de requisitos mais eficaz [KSK12]. Este tipo de ferramentas são capazes de recolher diversa informação sobre a utilização de um determinado *website*. No entanto, estas ferramentas, na sua maioria, apenas geram relatórios com estatísticas de tráfego de um determinado *website* e algumas métricas que, na sua maioria, são utilizadas apenas para os conteúdos com mais interessa da parte dos utilizadores [VMKR13]. Assim, neste momento, é possível afirmar que este tipo de ferramentas tem como foco principal a análise e divulgação de métricas de negócio, sendo os comerciantes os principais interessados neste tipo de dados.

É, portanto, essencial que este tipo de ferramentas possuam funcionalidades que permitam a possibilidade de realizar uma análise mais correta e concreta aos requisitos de um sistema, através de sugestão de novos *workflows* de trabalho, identificação e remoção de funcionalidades pouco ou nada utilizadas ou através da divulgação de relatórios mais detalhados e específicos sobre a *performance* das funcionalidades de um sistema.

Todos estes aspetos têm um impacto negativo na Qualidade do *Software* produzido. Esta última constitui um aspeto bastante tido em conta durante as fases de desenvolvimento de *software*, pois é algo que pode ser utilizado como ponto de distinção e de superação quando comparado com outros sistemas inseridos no mesmo ambiente e que competem entre eles pela liderança do mercado.

Posto isto, é possível concluir que esta área de análise e recomendação de requisitos também denominada de *web usage mining* tem um potencial bastante elevado, tanto para análise de preferências de utilizadores, como para ajudar no melhoramento das funcionalidades de um *website* e que, apesar de já muita pesquisa ter sido efetuada nesta área, ainda existem alguns desafios que precisam de ser trabalhados, tais como: falta de foco no melhoramento das especificações dos requisitos de *software* e falta de recomendação de melhoramentos a funcionalidades do sistema por parte da ferramenta e falhas na explicação detalhada de suporte e utilização deste tipo de ferramentas, que conduzem a suspeições por parte de *stakeholders* quanto aos verdadeiros benefícios da utilização das mesmas [CB].

No entanto, é importante referir que o processo de testes de *software* também possui um grande impacto na gestão e validação de requisitos [Ber07]. O processo de testes de *software* tem como objetivo principal garantir que todo o código desenvolvido cumpre com todas as funcionalidades propostas na especificação de requisitos [Mye04]. Este processo permite, assim, detetar falhas nos sistemas desenvolvidos e também erros passíveis de serem corrigidos atempadamente, melhorando a qualidade e consistência dos sistemas desenvolvidos.

Dentro dos vários tipos de teste existentes, os testes de regressão desempenham um papel primordial na validação de requisitos [YH07]. Este tipo de testes é executado após uma qualquer alteração, que tenha sido efetuada no *software* desenvolvido, com o objetivo de verificar se foi mantida a consistência do sistema e se o mesmo continua a operar como suposto [YH07]. Contudo,

existem atualmente várias dificuldades na criação de testes de regressão [YH07] , principalmente em termos de:

- **Complexidade.** Com o adicionar de cada vez mais funcionalidades extra ao sistema, o mesmo torna-se cada vez mais complexo, levando a que sejam necessários cada vez mais testes de regressão.
- **Complexidade Temporal.** A complexidade temporal é elevada, uma vez que estes tipos de teste envolvem a execução de uma grande quantidade de testes.
- **Valor de Negócio.** Uma vez que se torna difícil para um programador explicar a importância deste tipo de testes para pessoas que não possuem experiência técnica na área.

Torna-se, assim, importante proceder à automação deste tipo de teste, de forma a economizar e melhorar o tempo investido no processo de testes e também a qualidade dos mesmos.

O principal objetivo deste trabalho de investigação é conseguir extrair uma bateria de testes de regressão a partir dos dados navegação recolhidos, de forma a automatizar o processo de manutenção e validação de requisitos em sistemas de *software*.

Pretende-se também estudar a fiabilidade dos casos de teste obtidos afim de verificar a qualidade dos resultados obtidos em cada um dos casos de teste. A recolha de informação será efetuada por uma ferramenta denominada *Open Web Analytics* capaz de capturar dados de navegação associados a sessões de utilizadores em *websites*.

1.2.1 Estrutura do Documento

Para além da introdução, este relatório possui mais cinco capítulos.

No capítulo 2, está presente toda a revisão bibliográfica relacionada com a área de investigação deste projeto de investigação, bem como algum do trabalho relacionado e que serviu de base para este projeto. No capítulo 3 é apresentado o problema que serviu de motivação a este trabalho, bem como a abordagem a utilizar para o solucionar. São também apresentadas as ferramentas que servirão de base para a implementação da abordagem proposta.

No capítulo 4 são apresentados todos os detalhes relacionados com a implementação da abordagem proposta e no capítulo 5 é apresentada a metodologia de teste a utilizar, bem como os resultados da abordagem proposta em três casos de estudo diferentes.

Por último, no capítulo 6 são discutidos os resultados obtidos com a aplicação da abordagem proposta nos três casos de estudos diferentes e são apresentadas as conclusões finais deste trabalho, bem como trabalho futuro a realizar nesta área.

Introdução

Capítulo 2

Revisão Bibliográfica

Neste capítulo será feita uma revisão literária sobre o estado da arte na área da gestão e manutenção de requisitos de *software*.

Em primeiro lugar, na secção 2.2 são abordados todos os aspetos relacionados com a área de Engenharia de Requisitos. Dentro destes aspetos são inicialmente apresentados todos os tipos de requisitos existentes, havendo de seguida uma breve explicação de cada um deles. Posteriormente, é feita uma explicação detalhada de todo o processo de ER, bem como das fases que o englobam. Por fim, são discutidos, nesta secção, aspetos relacionados com a rastreabilidade de requisitos e a importância que este fator tem na produção de *software* com qualidade.

De seguida, nas secções 2.3 e 2.4 são apresentadas várias ferramentas relacionadas com análise de dados de navegação *web* e recomendação de alterações à especificação de sistemas de *software*.

Finalmente, em 2.5 são apresentadas várias metodologias de testes que têm como objetivo a automação e sistematização do processo de testes de *software*.

2.1 Introdução

Num mercado tão competitivo como é o da tecnologia, a qualidade do *software* produzido pode muito bem fazer a diferença entre um produto de excelência reconhecido por todos os *stakeholders* [MA15] e entre um produto menos bom, que, apesar de cumprir com todas as funcionalidades propostas, não possui uma *performance* tão elevada. Um dos fatores que influencia a qualidade do *software* produzido é a boa ou má gestão dos requisitos do sistema. O processo que permite a recolha de requisitos e os implementa ao longo de todas as fases do ciclo de vida do *software* denomina-se Engenharia de Requisitos (ER) [Som07].

2.2 Engenharia de Requisitos

Segundo o IEEE [CB] um requisito pode ser definido como:

- Uma condição ou capacidade necessária pelo utilizador de forma a solucionar um determinado problema ou atingir um determinado objetivo;
- Uma condição ou capacidade que um determinado sistema, ou uma das suas componentes, deve possuir de forma a conseguir satisfazer um contrato ou especificação impostas por um determinado documento;
- Uma representação documentada de uma condição ou capacidade definida nos pontos acima referidos.

Um requisito é, de facto, uma propriedade bastante valiosa de um sistema de *software*, visto que são eles a ponte entre as ideias do cliente para o funcionamento do sistema e a implementação palpável dessas mesmas ideias. Apesar disso, de modo a fazer uma leitura mais exata da especificação e objetivo de um determinado requisito, o mesmo precisa de ser devidamente quantificado, relevante, verificável, rastreável e detalhado.

2.2.1 Tipo de Requisitos

Um requisito de *software* pode ser definido como requisito funcional ou requisito não funcional.

Um requisito funcional tem como foco principal a especificação das funcionalidades que o sistema deverá fornecer aos seus utilizadores, por exemplo, no caso de um pacote de leite um dos seus requisitos funcionais principais poderá ser: "*A capacidade de conter liquido sem que o mesmo escorra para fora do recipiente*" [DDgP]. Em alguns casos, segundo Sommerville [Som04] um requisito funcional também pode exemplificar que tipos de ações é que um sistema não deve tomar, sendo que este tipo de requisitos está muito dependente do tipo de *software* a ser produzido e do tipo de abordagem tomada pela equipa de desenvolvimento aquando da escrita da documentação necessária para a especificação deste tipo de requisitos.

Por outro lado, um requisito não funcional pode ser descrito como um tipo de requisito que especifica o tipo de critérios que serão usados para julgar o desempenho de um sistema como um todo, em vez de apenas algum tipo de função específica [Som07]. Estes requisitos podem também ser apelidados de "*atributos de qualidade*" [Som07].

Os requisitos não funcionais podem ser divididos em 2 categorias de maior importância, sendo elas:

- Qualidades de execução, como por exemplo, segurança e usabilidade do sistema;
- Capacidade de evolução, como por exemplo, escalabilidade, manutenção, teste e extensão;

Em relação às qualidades de execução, é possível afirmar que um requisito de segurança tem como objetivo principal prevenir o sistema contra possíveis ataques, tanto físicos, como tecnológicos e externos do meio ambiente e um requisito de usabilidade tem como objetivo principal tornar a interação entre o utilizador e o sistema mais simples e agradável para o mesmo, por exemplo,

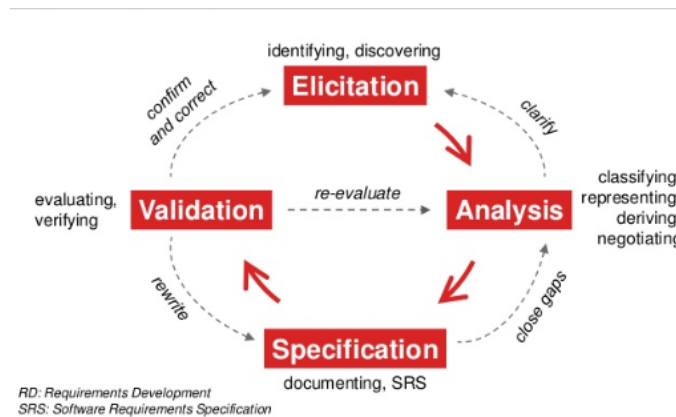


Figura 2.1: Fases do processo de Engenharia de Requisitos

através do uso de uma interface gráfica adaptada ao segmento de mercado que se pretende atingir com o produto.

Quanto aos requisitos de escalabilidade, os mesmos têm como foco principal, facilitar o possível futuro crescimento do projeto, tanto em termos de funcionalidades, como em termos de performance aos olhos dos utilizadores; já um requisito de manutenção foca-se mais no aspeto de reação do sistema a possíveis falhas que ele possa ter, por exemplo, através do *Mean Time To Repair - MTTR* que indica o tempo médio de correção de uma falha no sistema após a mesma ser descoberta. Por fim, um requisito de teste preocupa-se com o grau de suporte que um determinado sistema de *software* suporta um determinado tipo de teste num certo contexto, enquanto um requisito de extensão se preocupa com possíveis funcionalidades que possam ser adicionadas no futuro ao sistema de *software* em estudo [Don].

2.2.2 Etapas do Processo de Engenharia de Requisitos

As principais fases deste processo de Engenharia de Requisitos são: elicitação, análise, especificação, validação e manutenção de requisitos. No entanto, estas atividades podem ser agrupadas em duas fases principais: desenvolvimento de requisitos, que inclui as fases de elicitação, análise, especificação e validação e uma segunda fase, denominada gestão de requisitos, que engloba a manutenção dos mesmos como é possível observar na figura 2.2.

A elicitação de requisitos tem como objetivo principal o de identificar as fontes principais de onde irão ser extraídos os requisitos do sistema e, após esta fase de identificação tem como principal objetivo extrair requisitos do sistema dessas mesmas fontes [BS97a]. A elicitação de requisitos é uma fase crucial do processo de desenvolvimento de requisitos, a qual requer da parte da equipa de analistas bastante foco e conhecimento do domínio, padrões e restrições do sistema. Durante este processo de elicitação de requisitos decorrem as seguintes atividades: identificação de atores, cenários e casos de uso. Definem-se também as várias relações e dependências entre estes casos de uso e procede-se à refinação dos mesmos. Por fim, são também identificados os requisitos não funcionais do sistema e os objetos intervenientes do mesmo. Para realizar as atividades acima

mencionadas são utilizadas várias técnicas de elicitação de requisitos, como: entrevistas, *brainstormings*, inquéritos, prototipagem e revisões de documentos, que têm como objetivo principal proporcionar à equipa de desenvolvimento um melhor conhecimento do domínio e condições do sistema pretendidas pelo cliente.

A análise de requisitos é a fase na qual os requisitos anteriormente recolhidos são analisados em termos de custo e peso para o sistema [DM03]. Durante esta fase várias dependências e conexões entre os diferentes requisitos são identificadas. Esta fase é considerada o passo inicial que ajuda na compreensão de como o sistema se irá integrar com os processos de negócio. Durante a análise de requisitos, o âmbito do projeto e limitações em termos de *software* são definidos.

Durante a fase supra-mencionada também decorre uma sub-etapa denominada negociação de requisitos. A negociação de requisitos é um processo onde são identificados todos os *stakeholders* do sistema. O objetivo desta sub-etapa passa por resolver possíveis conflitos que possam existir entre diferentes requisitos. Durante este processo todos os requisitos existentes são discutidos com cada um dos *stakeholders* e no final das reuniões são criados vários critérios de aceitação para requisito, sendo que cada um dos requisitos existentes é priorizado. Esta fase tem um impacto e influência bastante importantes para o processo de desenvolvimento de requisito, pois permite evitar problemas de âmbito, custo e tempo, visto que, com a análise e priorização de requisitos, são definidos planos de trabalhos mais realistas que permitem atenuar o descontrolo no desenrolar do desenvolvimento do projeto. Para além dos problemas resolvidos, esta fase de desenvolvimento de requisitos tem também como objetivo evitar possíveis conflitos existentes entre *stakeholders*, entender melhor os requisitos do sistema existentes e as necessidades do cliente para o projeto[CoE].

A especificação de requisitos é a fase na qual todos os requisitos acordados e analisados com o cliente são documentados. Durante esta fase são produzidos vários documentos com o objetivo de dar uma especificação mais detalhada de cada requisito, de forma a que a compreensão dos mesmos seja facilitada. Estes documentos produzidos são cruciais para o desenvolvimento do sistema, pois facilitam uma perceção mais aprofundada de cada funcionalidade em específico e das relações e dependências que cada requisito possui em relação a outros. Uma boa documentação deve conter: uma explicação detalhada de serviços e funcionalidades que o sistema deve realizar; detalhes das restrições sobre as quais o sistema vai operar; explicação e definição de outros sistemas externos ao nosso, com os quais vai ocorrer interação; uma explicação detalhada sobre o domínio do problema e, por fim, o conhecimento das restrições do processo de desenvolvimento que a equipa de *software* utilizará [BS97b].

A fase de verificação e validação é responsável por confirmar que todo o *software* desenvolvido está de acordo com todos os requisitos documentados e que cumpre com as exigências dos clientes. Este processo de verificação e validação decorre durante as várias fases de desenvolvimento do sistema e uma análise rigorosa e detalhada é realizada de forma a verificar se o sistema cumpre com todos os requisitos especificados. Durante esta fase do processo de desenvolvimento de *software* várias técnicas são utilizadas para assegurar que todos os objetivos são cumpridos. Entre estas técnicas, é possível destacar: inspeções de *software*, revisões de código, auditorias e verificações de requisitos. O uso de todas estas técnicas de revisão tem como objetivo a descoberta e estudo de

possíveis falhas de *software* que possam ocorrer. Estas técnicas ajudam na prevenção de potenciais *bugs de software*, que podem levar ao colapsar de todo o sistema.[Som04]

A validação de *software* pode ser definida como o processo de análise ao produto que vem sendo desenvolvido, de forma a verificar se o mesmo cumpre com todas as expectativas dos clientes e se cumpre com todas as especificações documentadas.[McF]

Este processo de validação é realizado, tal como o de verificação, durante o processo de desenvolvimento de *software* ou no fim do mesmo. Estas duas fases (verificação e validação) estão intrinsicamente ligadas, no entanto a Validação ocorre sempre depois de se efetuar o processo de verificação. Nesta fase de validação são redigidos vários planos e casos de teste para verificar se o sistema cumpre com todos os seus requisitos. Durante a validação são efetuados vários tipos de testes, tais como testes de validação, integração, aceitação e funcionais. Os testes de validação e funcionais focam-se mais no teste individual de cada funcionalidade do sistema, de forma a verificar se a mesma está de acordo com todos os aspetos documentados. Posteriormente, os testes de integração testam um sistema como um todo, testando as várias interações entre funcionalidades e componentes do sistema. Por fim, os testes de Aceitação focam-se mais na perspetiva que os *stakeholders* têm do produto no seu estado atual e, se o mesmo é cativante e cumpre com todos os requisitos aos olhos destes.

A fase de gestão e manutenção de requisitos tem como objetivo minimizar o impacto que mudanças tardias nos requisitos possam ter nas sucessivas fases do ciclo de *software*.

A gestão de requisitos é o processo no qual mudanças nas especificações de requisitos são documentadas e controladas. Este processo envolve várias etapas, como: estabelecimento de um plano de gestão de requisitos, nova elicitação de requisitos, criação de casos de uso e de especificações adicionais, desenho do sistema e criação de casos de testes derivados dos casos de uso e das especificações suplementares [CHCH09].

2.2.3 Rastreabilidade de Requisitos

Segundo o IEEE[BS97b] rastreabilidade pode ser definida como:

- O grau pelo qual é possível estabelecer uma relação entre dois ou mais produtos do processo de desenvolvimento de *software*, com especial atenção para produtos que possuam relação predecessor-sucessor ou relação mestre-subordinado entre eles;
- A identificação e documentação de caminhos, conexões e dependências entre funcionalidades do sistema em questão;
- O grau entre o qual cada funcionalidade do processo de desenvolvimento de *software* estabelece a sua importância para o produto final;
- Todas as associações perceptíveis entre duas ou mais entidades lógicas, como por exemplo, requisitos, elementos do sistema, verificações e tarefas.

Na área de Engenharia de Requisitos, a rastreabilidade tem como principal objetivo perceber e especificar mais aprofundadamente requisitos de nível mais elevado, como por exemplo: objetivos, ambições e necessidades são transformados em requisitos de nível mais baixo (funcionais e não funcionais). Esta área está, portanto, intrinsecamente ligada à relação entre e satisfação entre as várias camadas de informação (artefactos). Apesar disso, a rastreabilidade também pode documentar várias relações entre diferentes tipos de artefactos de *software*, como por exemplo, requisitos, testes, *design*, modelos e componentes de *software*. Assim, é bastante comum fazer-se uso da rastreabilidade, de forma a demonstrar ou verificar que um requisito é verificado por um determinado artefacto de teste.

Rastreabilidade assume um papel de extrema importância quando do desenvolvimento de sistemas críticos e que, por conseguinte, possuem um número elevado de regras e métodos de segurança a seguir e a ser aplicados. Nestes tipos de métodos um dos requisitos mais requisitados passa por garantir que ocorre uma verificação entre os vários requisitos de segurança e performance existente, sendo a forma mais eficiente de realizar essa verificação através do uso de rastreabilidade [Ban11].

Geralmente, são utilizados dois tipos de verificação de rastreabilidade, sendo eles:

- Rastreabilidade Pré-Requisitos: usando rastreabilidade de requerimentos, uma funcionalidade já implementada pode ser relacionada com a pessoa ou grupo que inicialmente a requisitou durante a elicitação de requisitos. Esta ação pode ser realizada durante o processo de desenvolvimento de *software*, de forma a priorizar o requisito, determinando quão o mesmo é importante para um cliente específico. No entanto, este tipo de rastreabilidade pode também ser aplicada após o *deployment* dos casos de estudo, que indicam que uma certa funcionalidade já não é tão utilizada, de forma a perceber porque é que a mesma foi requisitada em primeiro lugar;
- Rastreabilidade Pós-Requisitos: este tipo de método permite rastrear as relações existentes entre os requisitos e todos os artefactos associados a cada um desses mesmos requisitos, como por exemplo, modelos, análise de resultados, casos de uso e de teste, implementação e, por fim, verificação. Artefactos associados a fases tardias do processo de desenvolvimento de *software* devem também ser rastreados até aos seus requisitos iniciais. Este processo é tradicionalmente feito através de uma matriz de rastreabilidade de requisitos.

No âmbito da Engenharia de Requisitos, o uso de rastreabilidade é deveras importante, principalmente para, tal como já foi acima referido, interligar cada requisito aos seus artefactos, sendo que estas conexões trazem vários tipos de benefícios, tais como:

- Análise de Impacto de mudanças - se um requisito sofre alterações, é necessário também rever as ligações entre esse mesmo requisito e os artefactos dependentes desse requisito. Graças ao uso de rastreabilidade, essas ligações e os próprios artefactos podem muito mais facilmente ser analisados e alterados, caso seja necessário, sendo assim reduzida a probabilidade de surgirem artefactos que não foram sequer analisados;

Revisão Bibliográfica

- Análise de Cobertura - a rastreabilidade garante também que, para além dos seus artefactos, o próprio requisito não deixa de ser analisado detalhadamente, principalmente no caso de sistemas críticos, onde é de extrema importância que se verifique se todos os requisitos estão implementados e funcionam devidamente;
- Análise do Estado do projeto - a análise da informação adquirida aquando da aplicação da rastreabilidade dos requisitos é também usualmente utilizada para verificar o estado em que o projeto se encontra de momento. Requisitos que não possam ser rastreados ou com uma matriz de rastreabilidade incompleta indicam que mais trabalho adicional é ainda necessário, de forma a finalizar o projeto em questão. Estas falhas na matriz de rastreabilidade indicam mais detalhadamente quais os requisitos e artefactos que necessitam de ser mais trabalhados;
- Reutilização de Componentes do Projeto - graças à rastreabilidade, é também possível estruturar os requisitos e os seus respetivos artefactos em diferentes *packages*. Estes mesmos *packages* podem, posteriormente, ser utilizados em vários produtos diferentes;
- Fortalecimento das relações entre requisitos e artefactos;
- Otimização de Testes - através da ligação entre requisitos, código fonte e casos de teste e seus resultados, torna-se mais acessível identificar as partes do código fonte afetadas pelo falhanço de algum caso de teste. É também possível eliminar testes de redundância com o uso de rastreabilidade.

Variados estudos documentaram a eficiência, mas também as dificuldades de recolher informação de rastreabilidade, sendo que se pode concluir segundo estes que:

- O uso de informação de rastreabilidade acelera e otimiza o processo de desenvolvimento de *software*. Um estudo com 71 programadores sujeitos a análise, que desenvolveram alterações a código fonte primeiro, com e depois sem suporte de informação de rastreabilidade, demonstrou bastantes benefícios do uso de rastreabilidade, sendo que esses mesmos programadores, com uso de rastreabilidade, completavam as tarefas propostas 24x mais rápido e com mais do dobro de eficiência de correção;
- Informação de rastreabilidade mais completa ajuda a evitar defeitos de *software* - Após um análise efetuada ao desenvolvimento do *software* de 24 projetos *open-source* de tamanho médio, foi possível concluir que existe uma relação inversamente proporcional entre a quantidade de informação de rastreabilidade encontrada sobre o projeto e a taxa de defeito encontrada no código fonte desenvolvido, sendo que quanto mais informação de rastreabilidade era encontrada e disponibilizada aos programadores, menor era a taxa de defeitos no código encontrada;
- Alcançar rastreabilidade de confiança máxima é extremamente complicado - Uma análise efetuada ao mercado de teste de *software* usado em dispositivos médicos na *US Fodd and*

Drug Administration (FDA) em 2013 identificou discrepâncias significantes entre informações de rastreabilidade prescritas e arquivadas aplicadas ao caso de alguns medicamentos.

Existem também várias formas de visualizar e analisar toda a informação recolhida aquando da aplicação de métodos de rastreabilidade, sendo as mais comuns: matrizes de rastreabilidade que podem ser definidas como representações tabelares que mapeiam os vários artefactos e requisitos colunas com os seus correspondentes artefactos e requisitos em linhas. Caso uma célula que mapeia dois requisitos esteja preenchida com um visto, pode concluir-se que existe uma relação de dependência entre os mesmos. A vantagem deste tipo de representação de informação prende-se com o facto de que todas as ligações entre artefactos são passíveis de analisar de uma só vez, sendo que o uso de filtros pode também ajudar na filtragem de requisitos que se pretendem rastrear. Em projetos de pequena e média dimensão esta representação é bastante útil, no entanto em projetos de dimensão mais elevada, com centenas de requisitos, a visualização deste tipo de informação torna-se mais complicada de analisar.

Outro tipo de representação de informação de rastreabilidade utilizada na engenharia de requisitos são os grafos de rastreabilidade. Neste tipo de representação, cada artefacto é representado por um nó e os nós ligam-se entre si através de arestas, sendo que esta conexão entre dois nós apenas acontece caso ocorra uma relação de rastreabilidade entre estes. Este tipo de grafos é bastante indicado para representar tarefas de desenvolvimento de *software*, permitindo obter uma melhor perspetiva nas conexões existentes entre tarefas. São caracterizados por possuírem um grande índice de compreensão da informação que disponibilizam por parte da equipa de desenvolvimento de *software*.

Por fim, existem ainda mais dois grandes tipos de representação de informação de rastreabilidade: listas e *hyperlinks* [Poh10].

Nas listas, as ligações de rastreabilidade são representadas em apenas uma entrada, que pode conter informação relacionada com o artefacto de origem que o atual provém, ou informação sobre o próximo com quem o artefacto atual se conecta. Este tipo de representação é especialmente utilizada quando operações em massa para diferentes artefactos devem ser executadas, sendo que os mecanismos de filtragem e ordenação permitem gerir, da melhor maneira possível, a forma como a informação é apresentada à equipa de desenvolvimento. No entanto, e comparado com todas as outras representações já demonstradas, esta é a menos aconselhada para ser aplicada em projetos de desenvolvimento de *software*.

Já os *hyperlinks* fazem a ligação entre artefactos rastreáveis entre si e permitem à equipa de desenvolvimento deslocar-se do artefacto fonte para qualquer um dos artefactos conectados a esse mesmo artefacto fonte [Poh10]. Este tipo de representação é bastante útil, caso seja necessária informação demasiado detalhada sobre um artefacto, pois permite melhor navegação entre requisitos do sistema, no entanto, visto que os requisitos não são visualizados compactamente, torna-se complicado fazer uso deste tipo de informação em projetos de grande dimensão.

2.3 Web Analytics

Uma das formas mais comuns de análise de tráfego e de dados de usabilidade de Internet é a análise estatística [GP16b].

Neste tipo de análise, a informação é agrupada em diferentes grupos pré-determinados por diferentes métricas, tais como: visualizações de páginas, domínios, sessões e visitas. Estudar o comportamento dos utilizadores de um determinado *website* pode conduzir a uma otimização da experiência do utilizador ao longo do ciclo de desenvolvimento desse mesmo *website*. Para medir o impacto das diferentes ações que acontecem durante a utilização de um *website* as *web analytics tools* fazem uso de diferentes métricas.

Métricas são diferentes tipos de informação disponível de utilização de *websites* e que são utilizadas como um meio de analisar o tráfego de um determinado *website*. Podem também servir de apoio no melhoramento desse mesmo *website*, de forma a que este consiga atingir os seus objetivos. Estas métricas podem ser divididas em quatro categorias diferentes, sendo elas: usabilidade de um *website*, referências, análise ao conteúdo de um *website* e garantia de qualidade.

Web analytics lida com diversos métodos para recolha, análise e medição de informação. Este processo envolve várias fases a serem aplicadas, sendo que, inicialmente, é preciso começar por definir os objetivos para cada métrica. Posteriormente, é necessário recolher toda a informação de tráfego e utilização de um determinado *website* de todas as fontes de recolha disponíveis que sejam consideradas relevantes. Por fim, após recolha de toda a informação disponível, cada pedaço de informação é agrupado na sua métrica respetiva e, após esta fase, cada métrica procede à sua análise independente, sendo depois todas as mudanças necessárias implementadas.

2.3.1 Ferramentas

?? Atualmente, existem bastante ferramentas para colecionar e analisar informação sobre tráfego de um determinado *website*. Cada uma delas possui diferentes capacidades e funcionalidades, podendo diferir entre si quanto ao tipo de informação que cada uma coleciona. No entanto, e apesar de existirem várias ferramentas, o foco principal da maioria delas consiste na análise estatística do tráfego de um determinado *website*, não fornecendo como pretendido os necessários relatórios de análise detalhada a cada funcionalidade [BS97a].

Ao recolher várias métricas de análise de um *website*, como número de visitas e visitantes e duração da visita, pode desenvolver-se indicadores de desempenho chave (*KPIs* - *Key Performance Indicators*) - um modelo analítico versátil que mede diversas métricas entre si para definir as tendências dos visitantes. Os *KPIs* fazem uso desses números dinâmicos para obter uma mais detalhada imagem do comportamento de um utilizador no *website*. Esta informação permite que as empresas alinhem os seus objetivos pessoais com os seus objetivos de negócios, de forma a identificar áreas a melhorar, promover partes impulsionadoras do *website*, testar as novas funcionalidades do *website* e, finalmente, aumentar a receita [CHCH10].

O tipo de dados recolhidos também está interligado com as métricas e *KPIs* que cada um analisa e reporta. Existe um grande número de métricas e *KPIs* presentes em cada ferramenta, no entanto os mais presentes costumam ser:

- Índice de Conversão: KPI responsável por medir a percentagem total de utilizadores que ao visitarem o *website* executam uma determinada ação;
- Página de saída: A última página que os utilizadores visitaram antes de abandonar o *website*;
- Página de Navegação: As páginas pelas quais os utilizadores acederam ao *website*;
- Novo utilizador: Um utilizador que está a aceder ao *website* pela primeira vez;
- Percentagem de novos utilizadores: KPI que mede o rácio entre novos e regulares utilizadores no *website*;
- Utilizador repetente: Um utilizador que já visitou este mesmo *website* e está agora a revisita-lo;
- Utilizador único: Um utilizador específico que acede ao *website*.

Apesar de atualmente existir uma pesquisa contínua no âmbito das *Web Analytics*, ainda existem algumas barreiras que os investigadores têm de superar. A análise e a pesquisa realizadas até ao momento mostram que alguns dos desafios atuais deste ramo são:

- Existem, atualmente, várias ferramentas de análise a *websites* com um elevado volume de dados, no entanto não fornecem recomendações para a melhoria do *website* com base nos dados recolhidos.
- Os *stakeholders* são, muitas vezes, céticos à introdução de uma nova forma de suporte automatizado de ferramentas [CHCH09]. Portanto, é necessário apresentar recomendações de elevado nível para que seja possível responder a todas as dúvidas dos mesmos.
- Falta de foco no melhoramento da especificação dos requisitos de *software*.
- As ferramentas de análise a *websites* atualmente existentes possuem uma série de pontos fracos. Por exemplo, não analisam o serviço com base em diferentes tipos (funções) dos utilizadores, não analisam os caminhos de navegação típicos (que podem ser úteis para definir ou melhorar os fluxos de trabalho); não produzem relatórios baseados nos requisitos e, portanto, num idioma mais próximo do negócio.

A área de *Web Mining* possui um grande potencial para fornecer as necessárias recomendações na gestão de requisitos aos utilizadores com base nas suas preferências [DCH10]. A *Web usage Mining* é, também, um campo de estudo com imenso potencial para a recomendação de requisitos, de forma a ajudar a equipa de desenvolvimento de *software* a melhorar o seu produto.

Atualmente, uma análise direcionada à melhoria dos requisitos não acontece, o que leva a que esses mesmos dados acabem por ser desconsiderados para o processo de desenvolvimento do produto e de melhoria da qualidade de um *website*. As ferramentas de *Web Analytics* existentes [CoE] não disponibilizam funcionalidades que permitam realizar uma análise mais assertiva para sugerir melhorias no *website*, como por exemplo, sugerir novos fluxos de trabalho, identificar e remover recursos não utilizados ou apresentar relatórios mais legíveis.

2.4 Sistemas de Recomendação

Em Engenharia de *Software*, um Sistema de Recomendação pode ser definido como: "*Uma aplicação de software que providencia à equipa de desenvolvimento de software informação valiosa capaz de ajudar a equipa numa determinada tomada de decisão de uma determinada tarefa*"[MT09].

Um sistema de recomendação, na maioria dos casos, tem como objetivo ajudar a equipa de desenvolvimento a escolher quais as melhores decisões a tomar durante as etapas do processo de desenvolvimento de *software*, podendo essas mesmas decisões variar, desde reutilização de código, a escrita de relatórios detalhados de divulgação de erros [GP18a].

Em sistemas de recomendação, a utilidade de um determinado item é, normalmente, representada por um *rating* que indica o quanto um determinado utilizador gostou de usufruir de um determinado item. Dependendo da aplicação utilizada, esta informação pode estar disponibilizada e este item pode ser especificado pelo utilizador, o que, normalmente, acontece em casos de *ratings* definidos pelos utilizadores ou, então, o *rating* desse mesmo item pode ser computado pela aplicação, fazendo, neste caso, a aplicação uso de funções que tentam maximizar o lucro e a eficiência do sistema em estudo[PB07]. Atualmente, este tipo de sistemas tem sido bastante utilizado em *websites* com preponderância comercial, como por exemplo *Amazon e ebay*.

Os Sistemas de Recomendação são normalmente classificados em três categorias distintas:

- **Recomendações baseadas em conteúdo:** este tipo de sistema sugere ao utilizador itens baseados nas preferências anteriores desse utilizador, aquando das suas visitas passadas ao *website* em análise. As recomendações são efetuadas através de comparações entre o perfil do utilizador em análise com uma série de itens pertencentes às mesmas áreas de preferência desse utilizador. Após esta análise estar concluída, são recomendados ao utilizador os itens com maior taxa de proximidade às suas preferências. A melhor abordagem de implementação deste tipo de recomendações passa por calcular o grau de semelhança entre as preferências do utilizador e cada item em análise individualmente [SS10].
- **Recomendações colaborativas:** neste tipo de abordagem são recomendados ao utilizador itens que uma grande maioria de outros utilizadores gostaram no passado. A implementação deste tipo de recomendação é feita tendo em conta as opções de compra que outros utilizadores tiveram no passado e, tendo também em conta, o tráfego em cada item e os comentários feitos a esse mesmo item [SS10].

- **Recomendações Híbridas:** neste tipo de recomendações são exercidas várias combinações entre recomendações colaborativas e recomendações baseadas em conteúdo. Sistemas de recomendação híbridos utilizam várias técnicas de recomendação, de forma a obter uma maior *performance* e eficiência nos resultados e a reduzir a taxa de possível rejeição por parte do utilizador. Normalmente, a abordagem mais utilizada é combinar recomendações colaborativas com qualquer uma das outras técnicas, de forma a evitar problemas de duplicação de informação.

Atualmente, existe bastante trabalho de pesquisa efetuado na área dos sistemas de recomendação [GP16a]. Estes trabalhos focam-se essencialmente na fase de desenvolvimento de código, onde esse tipo de ferramentas serve de auxílio à equipa de recomendação, transmitindo-lhes sugestões de reutilização de código ou de alteração na elicitação dos requisitos do sistema [MRE12]. Apesar disso, um sistema de recomendações, de forma a executar um trabalho completo, terá de providenciar à equipa de desenvolvimento de *software* a seguinte informação [BS97a]:

- Características do utilizador, tais como, emprego, nível de experiência laboral, trabalho já realizado e aspetos da sua vida social.
- O tipo de tarefa a executar, como por exemplo: adição de novas funcionalidades, otimização ou procura de erros.
- Detalhes específicos da tarefa a realizar: editar, analisar dependências ou visualizar código.
- As ações passadas executadas por esse utilizador, em relação ao sistema em estudo, tais como, artefactos visualizados e artefactos recomendados.

2.4.1 Ferramentas Existentes

Dentro da área de Sistemas de Recomendações em Engenharia de *Software* existem, atualmente, muitas ferramentas capazes de realizar uma eficaz recomendação de alterações a *software* a efetuar por parte da equipa de desenvolvimento. A maior parte destas ferramentas têm incidência nas fases de elicitação e análise de requisitos, existindo até agora pouco trabalho realizado no aspeto de validação de requisitos de *software*.

De entre as ferramentas existentes as mais utilizadas são: *ReqView*, *Orcanos*, *Process Street* e *Cradle*.

ReqView é uma ferramenta de gestão de requisitos, no qual é possível adicionar requisitos de um sistema de uma forma bem estruturada, o que, por conseguinte, permite executar uma eficaz rastreabilidade de requisitos e de design do produto e também uma boa gestão de testes e de possíveis riscos que afetem o sistema. A aplicação permite uma boa priorização de requisitos e fornece relatórios de testes e de estado atual do projeto.

Orcanos é uma ferramenta de gestão de requisitos que tem como grandes vantagens a integração com *cloud* e repositórios de código e também com *softwares* de controlo de qualidade de

código. Esta ferramenta permite aos seus utilizadores gerir requisitos e casos de testes numa interface bastante prática e possui índices e métodos de gestão de qualidade de *software*, no entanto tem como atual ponto fraco, o facto de não gerar uma boa documentação para os seus utilizadores.

Proccess Street é uma ferramenta de requisitos com um grande foco na produção de documentação viável para facilitar a gestão de requisitos por parte de uma equipa de desenvolvimento de *software*. Esta ferramenta, de todas as acima referidas, é a que possui melhor documentação como resultado do seu processo de análise e recomendação de requisitos, sendo que tem algumas falhas ao nível de interface, pois possui uma interface pouco intuitiva de se utilizar e, até ao momento, não permite priorização de requisitos como uma das suas funcionalidades.

Por fim, o *Cradle* é considerada a ferramenta mais completa de se utilizar, aquando da gestão de requisitos de um complexo sistema de *software* [SS10]. Esta ferramenta providencia ao seu utilizador um estado da arte de utilização de ferramentas de gestão de requisitos e integração completa com outro tipo de ferramentas que servem de âncora ao processo de gestão de requisitos, tais como: modelação, gestão de casos de testes e gestão de configuração. Esta ferramenta é capaz de executar todas as tarefas que um sistema de recomendação precisa de executar, como: rastreabilidade e priorização de requisitos, produção de documentação, atributos de utilizador e produção de relatórios de análise a requisitos. No entanto, o único problema desta ferramenta, prende-se com o facto de, devido à sua complexidade, ser necessário algum tempo de habituação, que o tal estado da arte acima referido tenta compensar.

2.4.1.1 *REQAnalytics*

A *REQAnalytics* é um sistema de recomendação que, fazendo uso da informação de utilização de um determinado *website* e do mapeamento entre os requisitos funcionais e os elementos do sistema, permite fazer sugestões para a especificação dos requisitos de *software*. Apesar de possuir muitas funcionalidades comuns à maior parte das ferramentas de mapeamento de Requisitos, a mesma não se pode definir como tal, devido ao facto do objetivo principal da ferramenta *REQAnalytics* ser complementar e servir de suporte à tarefa de manutenção de requisitos através da atribuição de várias recomendações de modificação das propriedades de cada requisito do sistema. As principais funcionalidades desta ferramenta são:

- **Integração com uma ferramenta de análise de tráfego *web*.** A ferramenta é capaz de receber e relacionar a informação proveniente dos dados de tráfego *web* com os requisitos funcionais do sistema. Com esta análise da informação recebida pela ferramenta, é possível fazer sugestões de alteração aos requisitos dos sistemas, pois esta permite analisar a performance individual de cada requisito no sistema [GP18b].
- **Importação de Requisitos.** A ferramenta permite também a leitura de requisitos funcionais do sistema a partir da especificação de requisitos de *software* presente num ficheiro XML devidamente estruturado [GP18b].

- **Mapeamento.** Esta funcionalidade está presente na ferramenta e permite exercer um mapeamento entre requisitos funcionais do sistema com páginas *web* e elementos do *website*. O mapeamento associa cada requisito funcional ao seu elemento de *website* correspondente [GP16b].
- **Listagem dos Requisitos Funcionais.** Com esta funcionalidade, é possível obter uma listagem de todos os requisitos funcionais do sistema, bem como toda a informação de mapeamento, mostrando, para cada requisito, todos os elementos e páginas *web* associadas [GP16a].
- **Dashboard Informativo.** Com este *dashboard*, é possível visualizar a seguinte informação: requisitos já mapeados, número de mapeamentos estabelecidos, número de recomendações já geradas pela ferramenta e percentagem de requisitos prioritários existentes em cada sistema [GP18b].
- **Grafo de Dependências Diretas de Requisitos.** Esta funcionalidade permite observar o grafo das dependências entre requisitos funcionais detetadas pela ferramenta. Cada nó do grafo representa um requisito funcional do sistema e cada aresta representa uma dependência direta entre um ou mais requisitos do sistema [GP16a].
- **Sequência de Traços de Execução mais Comuns.** Com a sequência o utilizador pode visualizar quais são as sequências de ações mais escolhidas por cada utilizador do sistema. Porém, esta sequência, em vez de ser definida por páginas *web* mais utilizadas, é-o pelos requisitos funcionais mais executados ao longo da referida sequência de traços de execução.
- **Matriz de Rastreabilidade.** Nesta matriz estão presentes todas as correlações existentes entre cada requisito funcional e elementos e páginas *web* associados [GP16b].
- **Relatório de Análise Estatística aos Requisitos.** Neste relatório pode observar-se os requisitos mais utilizados e outro tipo de métricas, como requisitos de entrada e de saída [GP16b].
- **Relatório de sugestões de alterações à especificação de requisitos do sistema.** Neste relatório estão presentes as principais sugestões de alterações à especificação de requisitos de *software* do sistema. As principais sugestões podem ser: criação de novos requisitos de *software*, alteração da prioridade de determinados requisitos, eliminação de determinados requisitos, divisão de um requisito em dois ou mais requisitos de sistema e adição ou eliminação de dependências entre requisitos de *software* do sistema [GP16a].

Esta ferramenta foi construída de forma a ser utilizada a partir de um navegador *web*. Em 2.2, é possível visualizar um diagrama onde está presente a arquitetura da ferramenta. Esta foi desenvolvida em PHP e usa uma base de dados *MySQL* como suporte para armazenamento de dados. Tal como já foi referido anteriormente, o objetivo principal passa por gerar recomendações de alteração à especificação de requisitos de *software* de um sistema, através da análise dos dados

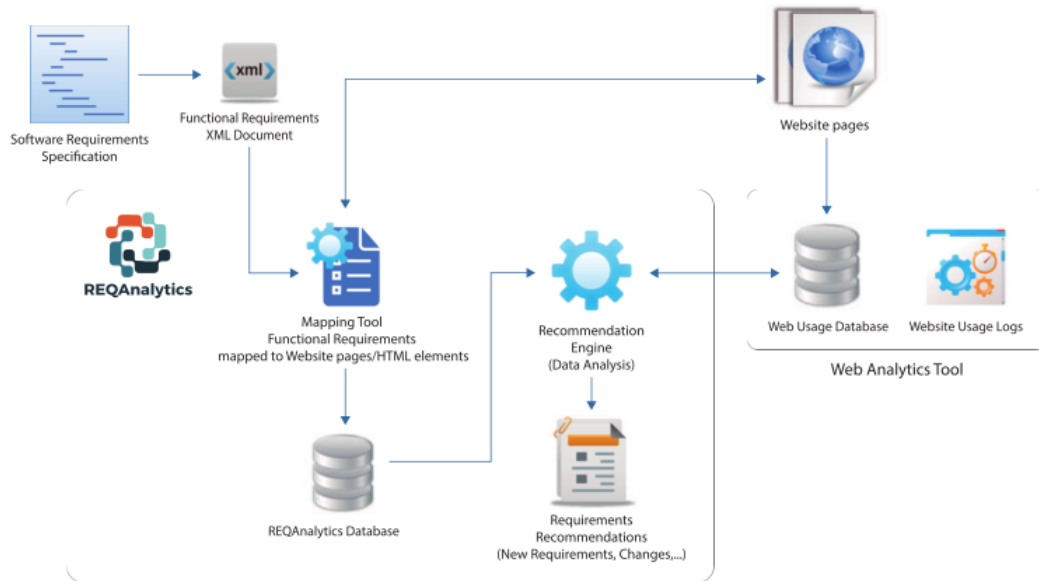


Figura 2.2: Arquitetura da Ferramenta *REQAnalytics*

de navegação *web* desse mesmo sistema. De forma a gerar estas recomendações, foi desenvolvida uma ferramenta de mapeamento integrada na *REQAnalytics*, que permite executar o mapeamento entre os requisitos funcionais do sistema e elementos *web* associados. Para recolher os dados de navegação a ferramenta faz uso de uma *Web Analytics Tool*, denominada OWA.

2.5 Testes de Software

Hoje em dia, é dada cada vez mais importância ao processo de testes aplicado ao *software* desenvolvido. Este processo permite obter uma maior consistência e credibilidade no produto desenvolvido, e não pode ser encarado como uma ação à parte, a realizar apenas no fim do processo de desenvolvimento do produto [MP14a]. No entanto, quanto maior for a dimensão e complexidade do sistema desenvolvido, mais moroso será o desenvolvimento de um processo de teste manual a aplicar [MP14a].

É por isso, cada vez mais importante, fazer uso de abordagens que consigam automatizar o processo de desenvolvimento de testes com qualidade, de forma a reduzir os custos que este processo de desenvolvimento de testes implica em todo o ciclo de criação de *software* [MP14a].

2.5.1 Model-Based Testing

Model-Based Testing é uma técnica de testes de *software*, na qual casos de teste são gerados a partir de um determinado modelo, que descreve variados aspetos do sistema sob teste [AD97]. Esta abordagem pretende verificar se a implementação executada está em conformidade com o modelo

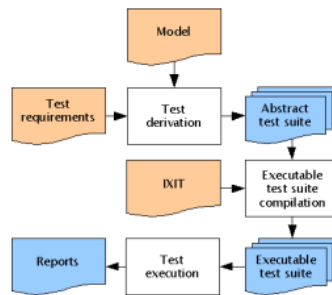


Figura 2.3: Sequência de passos de *Module-Based Testing*

de sistema previamente desenvolvido, introduzindo assim uma maior automação e criterização ao processo de testes aplicado.

De acordo com [AD97], recorrendo à utilização de programas que simulem modelos de *software*, (como por exemplo transições, ações, ...), é então possível gerar casos de teste que após execução permitem:

- Descobrir falhas no sistema;
- Definir os cenários bases de uso do sistema;
- Manter um histórico da estrutura do sistema e reduzir custos do processo de desenvolvimento de testes.

Existem várias variantes deste processo que podem ser aplicadas para gerar casos de teste a partir de um modelo, no entanto, a mais comum é a utilização de caminhos entre cenários. Um caminho pode ser definido como a sequência de ações ou eventos executados através de todo o modelo, que definem um cenário de uso do sistema em estudo [AD97]. À medida que vão sendo criados caminhos entre os vários cenários do sistema, são também gerados vários *scripts* de teste para cada um desses caminhos. Esses mesmos *scripts* são, posteriormente, aplicados ao sistema, de forma a validar a integridade e consistência do mesmo como um todo, como é possível observar na Figura 2.3.

2.5.1.1 PBGT

Pattern Based GUI Testing é uma abordagem de teste baseada na técnica de *Model-Based Testing* [CPFA10], que tem como objetivo principal a sistematização e automação do processo de testes em interfaces gráficas fazendo uso de padrões de teste em interfaces gráficas [MP14b]. Um padrão de desenho de interface gráfica pode ser definido como o conjunto de soluções à disposição das equipas de desenvolvimento de *software* para solucionar possíveis problemas na criação de interfaces gráficas [MP14b].

Este tipo de metodologia é suportado por uma ferramenta, disponível como *plug-in* do Eclipse, que está dividida em cinco componentes principais:

- **PARADIGM.** *PARADIGM* é uma *DSL (Domain Specific Language)* para desenvolver modelos de teste para interfaces gráficas baseados nos padrões de teste das mesmas [MP14a].
- **PARADIGM-RE.** Uma componente extra que tem como objetivo modelos *PARADIGM* a partir de páginas *web* sem ter de recorrer a código fonte [NPF14].
- **PARADIGM-TG.** Uma componente de geração de casos de teste automáticos, que permite criar casos de testes a partir de modelos *PARADIGM* [MP14b].
- **PARADIGM-TE.** Uma ferramenta que permite a execução de casos de teste e que produz um relatório de análise aos mesmos [MPM13].
- **PARADIGM-COV.** Uma componente extra que permite ao testador analisar cobertura dos testes gerados [PV17].
- **PARADIGM-ME.** Um ambiente de modulação que permite ao testador a criação e configuração de modelos de teste [MP13].

O processo de PBGT engloba seis fases: modulação, configuração, geração de casos de teste, execução de casos de teste, análise de resultados e atualização do modelo quando necessário. Na fase de modulação é obtido parte do modelo de teste a partir de processos de *reverse engineering* de uma aplicação já em análise. Este passo é executado pela componente *PARADIGM-RE*, que explora aplicações sob teste e tenta inferir um modelo de padrões de testes em interfaces gráficas aplicado à aplicação a testar. Após este passo estar concluído, a componente *PARADIGM-TG* gera os casos de teste necessários, a partir dos modelos criados. De seguida, esses mesmos casos de teste são executados na componente *PARADIGM-TE* e é feita uma análise ao relatório de execução gerado pela ferramenta. Posteriormente, caso seja necessário, podem ser adicionadas alterações aos modelos desenvolvidos, recorrendo ao ambiente *PARADIGM-ME*, sendo depois necessário executar novamente testes ao modelo gerado, de forma a preservar a integridade do sistema.

Tendo em conta que os casos de teste gerados a partir desta abordagem são derivados de modelos e não de código fonte, este tipo de abordagem é vista como uma forma de *black-box testing* [AOV16]. Nesta abordagem de testes baseados em modelos, se houver alterações a efetuar, altera-se o modelo e depois são gerados casos de teste automaticamente. Quando não existem modelos, a solução poderá passar por gerar testes de regressão automaticamente. No entanto, esta solução possui muitos obstáculos, que serão abordados na secção 2.5.2.

2.5.2 Testes de Regressão

Segundo [Mye04], testes de regressão podem ser definidos como um conjunto de baterias de teste, que asseguram que um sistema previamente desenvolvido e testado, quando em contacto com outros sistemas de *software* ou após alterações efetuadas no mesmo, continua a operar devidamente. Algumas destas alterações, como atualizações, melhoramentos e mudanças de configuração, podem levar ao aparecimento de novos *bugs* no sistema em questão. É, então, de extrema

importância executar uma análise aos efeitos que estas mudanças podem ter na consistência do sistema, de forma a determinar as áreas mais suscetíveis a falhas após a aplicação das referidas mudanças.

O objetivo principal desta metodologia é assegurar que mudanças, como as acima referidas, não introduzam novas falhas no sistema [Mye04] e se, estas mesmas mudanças, em partes específicas no sistema afetam outras partes do mesmo [GA15].

As técnicas mais utilizadas aquando da aplicação desta abordagem de teste são:

- **Correr todos os casos de teste novamente.** Esta técnica, apesar de dispendiosa, visto que é preciso voltar a executar todos os testes previamente desenvolvidos, assegura que não surgirão novos *bugs* no sistema, após modificação do código previamente desenvolvido, e garante a manutenção da integridade do sistema;
- **Seleção de Testes de Regressão.** Ao contrário da técnica anterior, neste caso, são selecionadas apenas alguns casos de teste para serem executados, desde que o seu custo de execução seja inferior ao da execução da totalidade de todos os casos de teste do sistema [Dug08].
- **Priorização de casos de teste.** Nesta técnica prioriza-se os casos, de modo a que testes com um índice mais elevado de prioridade sejam executados primeiro que testes com menor índice de prioridade. Os critérios de priorização mais utilizados são: priorização por versão, onde casos de testes relativos a uma determinada versão do *software* recebem um maior grau de prioridade e priorização geral, onde casos de teste sequencias recebem índices de prioridade de acordo com a ordem dessa sequência [Dug08].

Aquando do uso de metodologias ágeis para desenvolvimento de *software*, quando os ciclos de desenvolvimento de *software* são muito curtos e existem mudanças muito frequentes no sistema, o uso de testes de regressão pode envolver alguns problemas, devido ao elevado custo de execução de alguns casos de teste [YH07]. Este tipo de testes é, normalmente, executado por ferramentas de automação (ex: *Selenium*) e pode ser formalmente caracterizado como um conjunto de testes funcionais. Testes funcionais são o conjunto de testes responsáveis por testar todas as especificações do sistema, fornecendo ao programa um conjunto de *inputs* que visa testar a consistência do mesmo a ameaças externas [Mye04].

Contudo, ainda existem alguns desafios a superar nesta área. Em alguns casos, quando o sistema a testar é demasiado complexo o testador pode não ter acesso a algumas componentes do mesmo, o que torna o processo de geração de testes de regressão menos consistente [YH07]. Em casos de sistemas complexos, o processo de execução deste tipo de testes, pode também ser demasiado extenso e requerer uma grande alocação de recursos por parte da equipa de desenvolvimento [Dug08]. Em muitos casos, estas equipas de desenvolvimento têm dificuldade em alocar estes recursos, uma vez que a importância deste tipo de testes é menosprezada por elementos sem experiência técnica na área, visto que é difícil explicar a importância de testar, com regularidade, todas as componentes de um sistema [YH07].

2.5.3 Codeception

Codeception é uma ferramenta de execução de testes de regressão desenvolvida com o objetivo de tornar os testes mais práticos e intuitivos, visto que facilita a escrita e leitura dos mesmos. A ferramenta foca-se, principalmente, na aplicação de BDD. BDD é uma metodologia de desenvolvimento de *software* ágil que surgiu com o intuito de melhorar o processo de comunicação entre as equipas de negócio e de implementação. As principais práticas utilizadas nesta metodologia são:

- Grande uso de testes de automação;
- Pequena descrição de cada funcionalidade a testar em ambiente de teste;
- Implementação simplista do teste de cada funcionalidade, recorrendo a exemplos abstratos;
- Implementação de um teste diferente para cada um dos passos de teste;

Com este tipo de implementação, em que cada teste é implementado num formato semelhante a uma [user story], é possível que as equipas de desenvolvimento, manutenção e de negócio estejam no mesmo nível de conhecimento sobre de que forma as funcionalidades do sistema vão ser testadas. Com este formato de testes é, também, possível que pessoas com menos conhecimento informático consigam escrever ou editar testes de aceitação.

2.5.3.1 Ambiente de Teste da Ferramenta *Codeception*

De forma a poder criar um ambiente de teste, primeiro, é necessário gerar um ficheiro PHP onde o testador irá criar os seus próprios testes. Para isso, basta aceder ao diretório de instalação da ferramenta no seu sistema através da linha de comandos e executar uma instrução de criação de ficheiro de testes. A ferramenta permite a criação de ficheiros para testes unitários, funcionais ou de aceitação. Como parâmetros é necessário indicar o tipo de teste a criar e, seguidamente, o nome do ficheiro de testes. Em 2.1, é possível visualizar um exemplo da criação de um ficheiro de testes. Este ficheiro estará depois presente na pasta `tests` adicionada ao diretório do seu sistema, após a instalação da ferramenta.

```
1 $ php vendor/bin/codecept generate acceptance HelloWorld
```

Listing 2.1: Exemplo de Criação de Ficheiro de Testes

Após a execução da instrução anterior, obtém-se um ficheiro, presente em 2.2 já com as seguintes instruções pré-definidas:

```
1 <?php
2 $I = new AcceptanceTester($scenario);
3 $I->wantTo('perform actions and see result');
```

Listing 2.2: Ficheiro de Testes gerado

Na primeira instrução é criado um cenário abstrato onde o testador, representado na variável `$I`, poderá criar os seus próprios testes. De forma a criar esses testes, o testador tem à sua disposição os seguintes métodos:

- `wantTo()` - Este método deve ser utilizado antes da criação dos vários passos de um teste e tem como objetivo fornecer ao utilizador uma breve descrição inicial do objetivo do teste que se pretende executar;
- `see()` - Com este método, é possível executar verificações e comparações entre elementos HTML, de forma a verificar se um dado elemento está presente no passo de teste em que o ficheiro em execução está no momento;
- `fillField()` - Este método simula o preenchimento de um determinado campo de texto com a informação introduzida pelo testador. Para encontrar o campo a preencher, é possível utilizar o *Xpath*, *class* ou *id* do elemento em questão.
- `click()` - Com este método pode simular-se um *click* num determinado elemento HTML presente no sistema. O *click* pode ser executado através do texto do elemento HTML desejado ou através do seu *XPATH*, *class* ou *id*.
- `submitForm()` - Função que simula a submissão de um determinado formulário.
- `amOnPage()` - Este método permite verificar se o teste de momento se encontra na página *web* correspondente ao *url* introduzido.
- `expect()` - Este método deve ser utilizado previamente a um passo de execução de um teste e tem como objetivo fornecer à equipa de teste um comentário, no qual está presente o resultado que se pretende obter com aquele passo.

Em 2.3 pode observar-se um exemplo de um teste escrito recorrendo aos métodos previamente descritos. Tal como foi referido previamente, é possível observar que todos os passos de teste são de fácil compreensão, mesmo para utilizadores com menos capacidades de compreensão técnica. Para além disso, foram adicionados comentários extra, de forma a informar os utilizadores do objetivo de cada ação tomada.

```
1 <?php
2 $I = new AcceptanceTester($scenario);
3 $I->wantTo('Test forgotten password functionality');
4 $I->amOnPage('/forgotten')
5 $I->see('Enter email');
6 $I->fillField('email', 'incorrect@email.com');
7 $I->click('Continue');
8 $I->expect('Reset password link not sent for incorrect email');
9 $I->see('Email is incorrect, try again');
10 $I->amGoingTo('Fill correct email and get link');
11 $I->see('Enter email');
```

Revisão Bibliográfica

```
12 $I->fillField('email', 'correct@email.com');  
13 $I->click('Continue');  
14 $I->see('Please check your email for next instructions');
```

Listing 2.3: Teste Exemplo

Após a criação dos testes desejados, procede-se à execução dos mesmos. Para executar um ficheiro de teste é necessário correr na *shell* o seguinte comando, presente em 2.4:

```
1 php vendor/bin/codecept run acceptance --steps
```

Listing 2.4: Instrução para a Execução de um Teste

Após este passo, será lançado um ambiente de testes pela ferramenta que em *background* irá executar todos os passos de teste desejados. Para cada ação pode visualizar-se uma pequena descrição formal da mesma, tal como se observa no exemplo de ambiente de teste presente em 2.5.

```
1  Acceptance Tests (1) -----  
2  SigninCest: Login to website  
3  Signature: SigninCest.php:signInSuccessfully  
4  Test: tests/acceptance/SigninCest.php:signInSuccessfully  
5  Scenario --  
6    I am on page "/login"  
7    I fill field "Username" "davert"  
8    I fill field "Password" "qwerty"  
9    I click "Login"  
10   I see "Hello, davert"  
11   OK  
12  -----  
13  
14  Time: 0 seconds, Memory: 21.00Mb  
15  
16  OK (1 test, 1 assertions)
```

Listing 2.5: Exemplo de Ambiente de Teste

Neste exemplo, é também possível, observar a memória consumida pela execução do teste, bem como outras características: tempo de execução e número de testes executados com sucesso e sem sucesso.

2.5.3.2 Arquitetura da Ferramenta *Codeception*

Após a instalação da ferramenta, será adicionada ao diretório `src` do projeto em questão uma pasta denominada `tests`. Esta pasta contém mais 4 pastas no seu interior denominadas `data`, `output`, `support` e `acceptance`, o que se pode observar em 2.4.

Revisão Bibliográfica

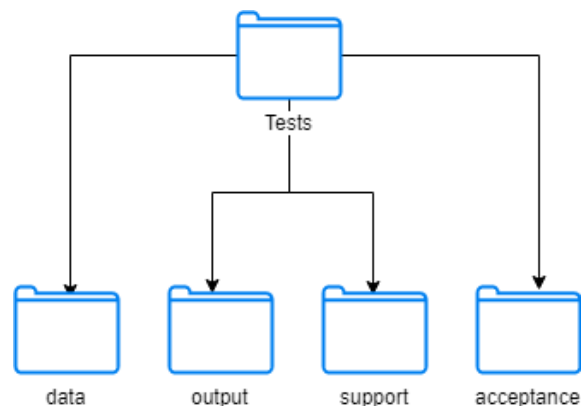


Figura 2.4: Arquitetura da Ferramenta *Codeception*

Na pasta *data* está presente um histórico de todos os testes previamente executados no referido projeto. É possível encontrar informação relativa à data em que os mesmos foram executados. Em *support* estão presentes *scripts* de código auxiliar necessários para a criação do ambiente de teste e execução dos mesmos. Na pasta *output*, é possível encontrar um relatório da execução do *script* de testes. Este relatório possui informação como: tempo de execução de cada teste, se o mesmo passou ou falhou e, caso tenha falhado, em que passo de execução é que isso se sucedeu, bem como cada passo de teste executado, como é possível de observar em 2.5. Por fim, na pasta *acceptance* encontram-se presentes todos os ficheiros de testes criados pelo testador.

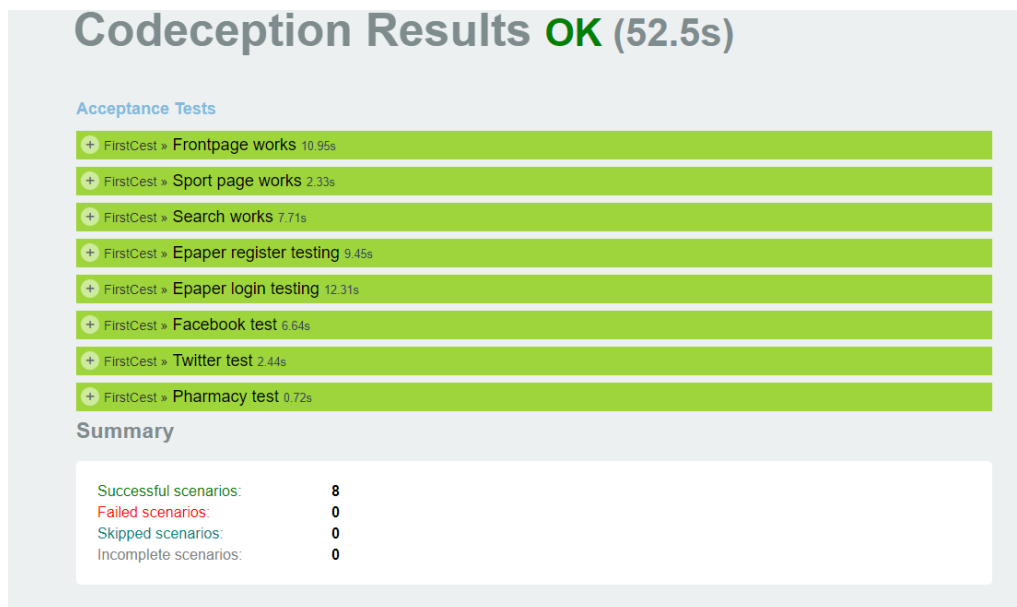


Figura 2.5: Exemplo de Relatório de Execução

Capítulo 3

Proposta de Solução

Neste capítulo, será apresentada a proposta de solução a desenvolver no âmbito desta experiência. Primeiramente, em 3.1, será apresentada uma breve contextualização do problema, que serviu de base para a realização desta experiência. Por fim, na secção 3.2 é apresentada e discutida, a abordagem a implementar de forma a solucionar o problema previamente referido em 3.1.

3.1 Definição do Problema

Tal como já foi referido, falhas na gestão ou eliciação de requisitos levam muitas vezes à rutura de projetos de *software* [Gro95]. Hoje em dia, as empresas apostam bastante na aplicação de novas tecnologias e métodos para recolher, analisar, documentar e gerir os requisitos dos seus projetos. Muitos destes requisitos são expostos em vários tipos de documentos com informação consideravelmente detalhada sobre os requisitos de cada um dos projetos.

Apesar disso, estes tipos de especificação de requisitos possuem ainda algumas limitações, tais como [BS97a]:

- Dificuldades em proceder a atualizações constantes do documento;
- Dificuldades em comunicar pequenas alterações no documento aos restantes membros da equipa de desenvolvimento;
- Dificuldades em guardar informação suplementar sobre cada requisito em específico;
- Dificuldades em criar ligações de dependência e de rastreabilidade entre requisitos e os seus artefactos correspondentes, nomeadamente: casos de uso e de testes, código, design e tarefas.

Para além disso, muitos destes projetos possuem uma fraca cobertura de testes, o que leva a que, aquando da sua execução em ambiente externo, estejam muito mais propícios a falhas do que outros projetos com uma maior cobertura de testes. Com isto, muitas empresas perdem grande parte do seu fator competitivo (qualidade do *software* desenvolvido), pois distribuem sistemas com uma menor garantia de qualidade e de performance para os seus utilizadores.

As ferramentas de gestão de requisitos conseguem atenuar estas dificuldades, visto que as primeiras possuem muitas funcionalidades de gestão de requisitos, que permitem o uso de uma rastreabilidade mais apurada e eficaz e melhor visualização das dependências entre requisitos, através de matrizes e listas, por exemplo. Algumas destas ferramentas possuem também a capacidade de análise aos casos de teste e de uso de cada requisito, produzindo ou documentação ou divulgando, através de interfaces, informação à equipa de desenvolvimento, o que permite analisar os vários artefactos de cada requisito.

Ainda assim, muitas destas ferramentas não focam um ponto essencial, que passa pela extração e análise de dados de utilização e de tráfego do *website* em estudo. Este tipo de dados permite à equipa de desenvolvimento uma análise mais eficaz do comportamento de cada funcionalidade do seu sistema, uma vez que esta extração de dados contém informação de extrema importância para a análise de um requisito, como por exemplo: o número de cliques num determinado botão, tráfego médio de pessoas numa determinada página, comentários positivos e negativos que permitem exercer vários tipos de sugestões, como a priorização de requisitos, alterações a certas funcionalidades sugeridas pelos utilizadores, eliminação de requisitos prejudiciais para o sistema ou que já não merecem tanta atenção por parte do utilizador.

Para além disso, e de acordo com o trabalho de pesquisa realizado até ao momento, foi possível denotar que nenhuma das ferramentas de gestão de requisitos que efetivamente extrai e analisa dados de navegação web, previamente referidas em 2, consegue gerar casos de teste a partir dessa mesma informação recolhida. Estes casos de teste são de extrema importância, visto que oferecem uma maior consistência ao sistema e permitem exercer várias verificações de prevenção que previnam o aparecimento ou a descoberta de novas falhas no sistema. Com a otimização de todo este processo, podemos então obter uma maior garantia de qualidade e de robustez do nosso produto, que lhe dará outro tipo de maturidade aquando da sua exposição a ameaças externas ao sistema.

O principal objetivo deste trabalho é, então, construir uma bateria de testes de regressão automaticamente, a partir da análise dos traços de execução extraídos por uma ferramenta de *web analytics*.

3.2 Abordagem Proposta

Tal como referido anteriormente, o objetivo deste trabalho é gerar casos de teste, através de informação de dados de navegação *web*. Para tal, será implementada uma abordagem dividida em 3 fases, tal como é possível observar em 3.1.

Na primeira fase serão recolhidos os dados de navegação *web* capturados pelo OWA. Estes dados serão, posteriormente, guardados numa base de dados alojada no servidor *phpMyAdmin*. Os dados a recolher serão os *clicks* efetuados por cada um dos utilizadores durante cada uma das sessões efetuadas pelos mesmos, bem como todo o tipo de informação associada a um elemento *clickado(tag,id,class e texto do mesmo),id* da sessão de cada um dos utilizadores e *id* de utilizador. Esta informação será distribuída pelas tabelas *owa_click,owa_session* e *owa_visitor* respetivamente.

Proposta de Solução

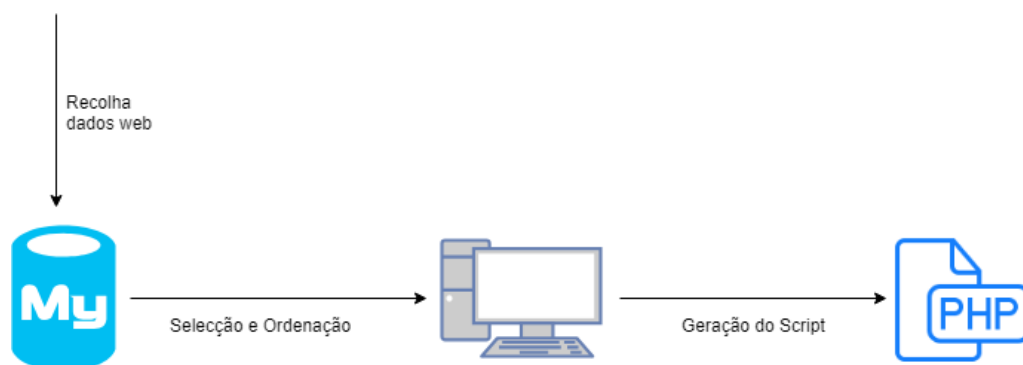


Figura 3.1: Fases da Implementação

Numa segunda fase, será criada uma extensão na ferramenta *REQAnalytics*, que permitirá ao utilizador fazer uma seleção dos dados a utilizar para a geração do seu *script* de testes. Na extensão o utilizador poderá:

1. Selecionar o *website* que pretende utilizar;
2. Selecionar a sessão de utilizador que pretende utilizar;
3. Para a sessão escolhida, visualizar quais os passos que possuem campos de texto que precisam que se introduza informação para a execução do teste, por exemplo, preencher campos de formulários como email, utilizador, password, entre outros;
4. Preencher os campos de texto com a informação que pretende;
5. Pedir a criação do ficheiro de teste;

Após a execução destes passos, será criado um *parser* que transformará a informação previamente seleccionada em instruções PHP executáveis em ambiente de teste. As regras de conversão a ser aplicadas são:

1. Um *click* num elemento HTML será convertido numa instrução `click(elemento X)` que simule esse mesmo *click*;
2. Um preenchimento de um campo de texto será simulado por `fillField(elemento X, informação)`

Para além destas conversões, serão adicionadas verificações intermédias antes e após a execução de cada passo de teste, como `see(elemento X)` para verificar se um determinado elemento existe antes de ser premido e `amOnPage(url X)`, que verifica se o teste foi redirecionado para a página correta após um determinado *click*. O ficheiro posteriormente gerado poderá ser consultado na pasta `acceptance`.

3.2.1 Tecnologias

Tanto a ferramenta *REQAnalytics*, como a *Codeception* foram desenvolvidas recorrendo a *PHP* e *Javascript*, essencialmente. *PHP* é a linguagem principal usada no desenvolvimento das ferramentas em questão, visto que todas as operações de *backend* executadas pelas mesmas foram desenvolvidas recorrendo a esta linguagem.

Para armazenar os dados foi escolhida uma base de dados do tipo *MySQL*, devido à melhor interoperabilidade existente entre *PHP* e *MySQL*. Uma vez que a *REQAnalytics* era uma ferramenta já em desenvolvimento, *PHP* foi a linguagem escolhida para o desenvolvimento da extensão e do *parser*, previamente referidos.

3.2.2 Casos de Estudo

De forma a validar a abordagem já explicada, serão analisados três *websites* diferentes: o *Newspaper* da Cidade de Tomar, o Portal do IPVC (Instituto Politécnico de Viana do Castelo) e o Portal de Multimédia da Cidade de Viana do Castelo. Os principais fatores para a escolha destes três *websites* foram: o facto da ferramenta *REQAnalytics* já conter dados relacionados com cada um dos *websites*, como por exemplo: dados de navegação, caminhos mais frequentes e requisitos funcionais de cada um dos sistemas e, também, o total acesso sem restrições a todos os dados de navegação relacionados com cada um dos casos em análise.

O principal objetivo desta experiência será verificar se realmente é possível extrair casos de teste válidos a partir da análise de dados de navegação de vários sites diferentes. Um caso de teste será considerado válido quando simular na perfeição todos os passos executados por um determinado utilizador durante uma determinada sessão.

Capítulo 4

Implementação

Neste capítulo é apresentada e discutida a abordagem utilizada para a geração dos casos de teste. O processo de geração está dividido em três partes. Na secção 4.1 é apresentado o processo usado para recolha e armazenamento dos *logs* de navegação recolhidos. Posteriormente, na secção 4.2 é exposto o método utilizado para seleção e ordenação dos dados a testar. Por fim, em 4.3 são explicadas todas as técnicas utilizadas para transformar dados de navegação num *script* de testes.

Os resultados da execução destes *scripts* de testes serão discutidos no Capítulo seguinte.

4.1 Recolha dos *logs* do OWA

O OWA(*Open Web Analytics*) é uma ferramenta *open source* de análise de dados de navegação *web*. Esta ferramenta é capaz de recolher vários tipos de informação, como:

- URL de páginas visitadas;
- Elementos HTML clicados(*ids, classes, tags, texto*) e informação a eles associada;
- Informação de sessões de utilizadores(Número da sessão,duração e *id* do utilizador);
- Sequência de páginas visitadas.

De forma a poder recolher todo este tipo de informação diversificada, o OWA usa um *script* de código *Javascript*, que precisa de ser injetado em cada página *web* que se pretende analisar. Este processo é muito semelhante ao adotado por outras ferramentas de *Web Analytics*, tais como: *Google Analytics*, *Woopra* ou *Piwik*. Toda esta informação é, depois, armazenada numa base de dados *MySQL* para posterior análise.

4.1.1 Estrutura da Base de Dados

Tal como referido anteriormente, toda a informação proveniente dos dados de navegação é armazenada numa base de dados *MySQL*. Esta está alojada numa plataforma de *software* denominada *phpMyAdmin*. A plataforma, desenvolvida em *PHP*, tem como objetivo facilitar ao utilizador

Implementação

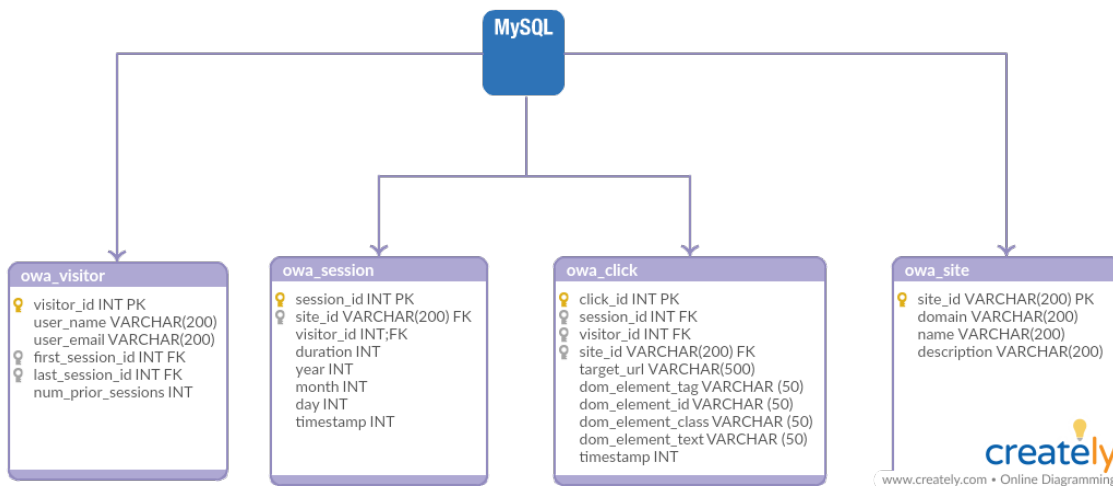


Figura 4.1: Estrutura da Base de Dados

a administração de bases de dados *MySQL*, proporcionando uma interface que permite ao administrador realizar vários tipos de operações sobre uma determinada base de dados.

Em 4.1, é possível observar a organização da Base de Dados utilizada para armazenamento de dados de navegação *web*. Na tabela *owa_click* é guardada toda a informação referente a um determinado *click* efetuado numa determinada página *web*, como por exemplo: o id do utilizador e da sessão em que esse *click* foi efetuado e o *url* do elemento em que o *click* se realizou, bem como as propriedades desse mesmo elemento (*ids, classes, tags* e texto associado). Esta tabela tem correlação com as posteriores tabelas *owa_session*, onde é armazenada toda a informação referente a uma determinada sessão, como a duração da mesma, data em que esta se iniciou e o *id* do utilizador que a iniciou; *owa_site* onde está armazenada a informação referente a um determinada *website*, como por exemplo o seu domínio, nome e uma breve descrição sobre o mesmo e *owa_visitor* onde estão os dados de um determinado utilizado, como o seu nome de utilizador, email, *id* da primeira e última sessão que efetuou e número de sessões já iniciadas.

4.2 Seleção e Ordenação dos Dados

Nesta etapa do processo de geração de casos de testes é pedido ao testador que filtre toda a informação presente na base de dados para gerar um possível caso de teste. Esta filtragem é executada de forma a permitir encontrar uma sequência de instruções reais realizadas por um determinado utilizador durante uma visita a um determinado *website*. Como é possível de observar na Figura 4.2, de forma a facilitar a execução deste processo, foi criada na ferramenta *REQA-nalytics* uma pequena interface que permite introduzir a informação necessária à construção do referido caminho.

O primeiro passo do processo passa pela escolha do *website* que servirá como ambiente de teste. São disponibilizados três ambientes de teste, sendo eles, o *newspaper* da Cidade de Tomar,

Implementação

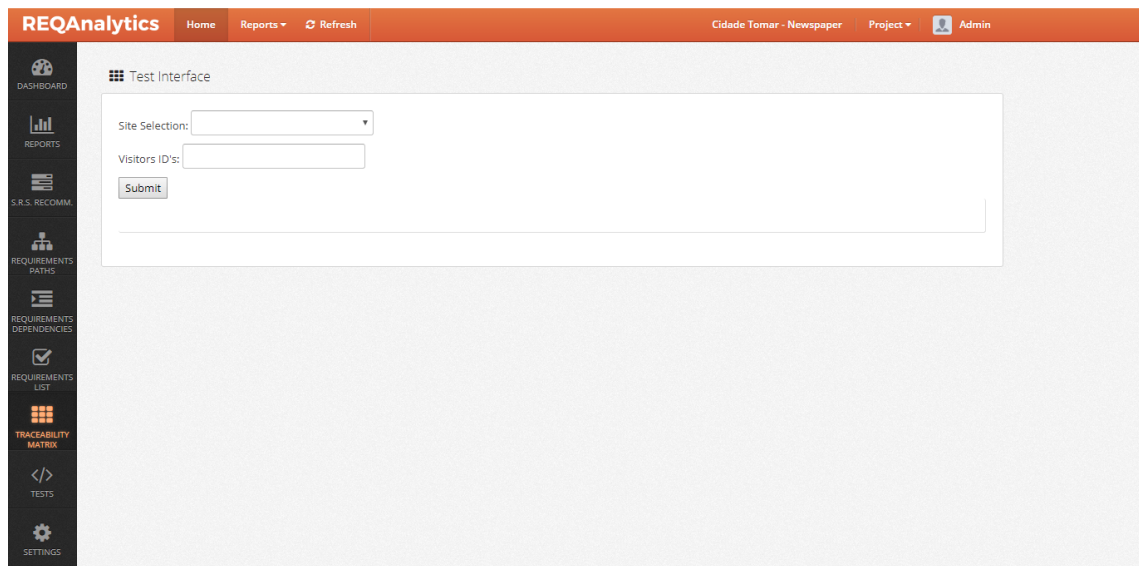


Figura 4.2: Interface de Testes

a plataforma do Instituto Politécnico de Viana do Castelo(IPVC) e, por fim, a plataforma do Centro de Multimédia do Porto. De seguida, é pedido ao testador que introduza o ID do utilizador do qual pretende simular a navegação realizada. Após a inserção destes dados, o sistema executa a seguinte *query*(4.1) à base de dados onde estão armazenados os dados de navegação.

```
1 $query = "SELECT dom_element_text,dom_element_tag,timestamp,dom_element_class,  
2 dom_element_id,target_url  
3 FROM owa_click WHERE site_id=$selected_site_id  
AND visitor_id=$selected_user_id ORDER BY timestamp;"
```

Listing 4.1: Query à Base de Dados

Com este conjunto de instruções é, então, possível extrair a sequência de instruções executadas por um determinado utilizador do *website*, ordenadas por ordem cronológica. Esta ordenação é feita através da variável *timestamp* que fornece a data e hora em que um determinado *click* se sucedeu. As variáveis *\$selected_site_id* e *\$selected_user_id* correspondem, respetivamente, ao *website* e sessão de utilizador introduzidos pelo testador anteriormente. Por fim, como resultado deste conjunto de instruções, obtêm-se, também, atributos do elemento *clickado* relevantes para a próxima etapa deste processo, como é o caso da classe, *id*, *tag*, texto e *url* associados ao elemento acima referido.

4.3 Geração do *Script* de Testes

Após a obtenção das sequências de instruções devidamente ordenada, é efetuada uma verificação, com o objetivo de encontrar todos os *clicks* que tenham sido efetuados num elemento do

Implementação

tipo INPUT. Com esta verificação, obtêm-se todos os passos, em que é necessário que o testador introduza informação para preenchimento de campos de texto. Esta informação é, posteriormente, divulgada ao utilizador, como é possível de observar em 4.3.

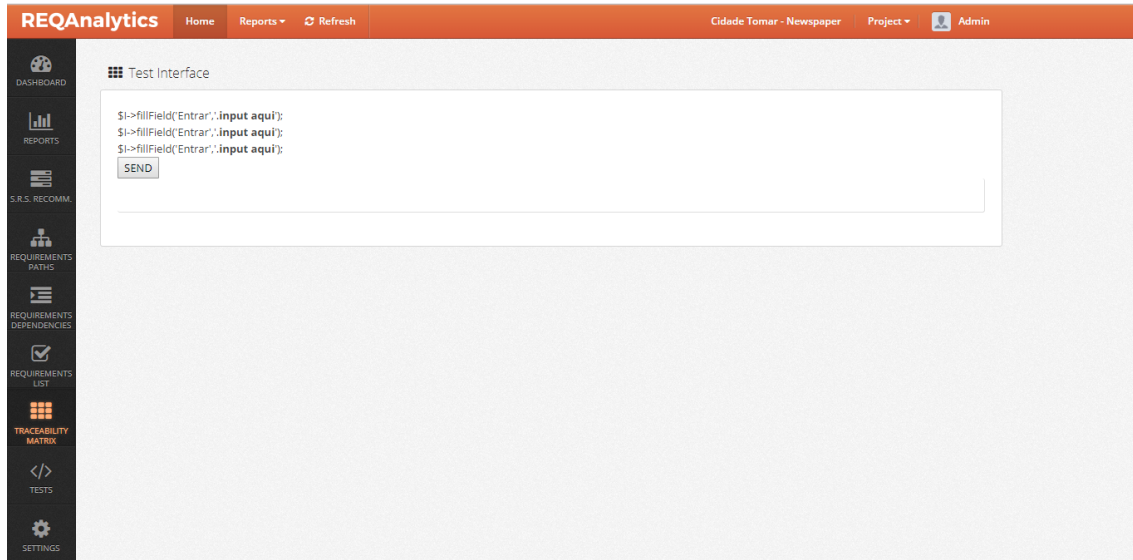


Figura 4.3: Exemplo de informação sobre Inputs

Caso nenhum dos passos de execução possua *clicks* em elementos do tipo INPUT, é apenas apresentado um ecrã sem nenhuma informação extra disponível para o testador, como é possível de observar em 4.4.

Por último, na fase final deste processo, a informação previamente seleccionada e estruturada é transformada num *script* de casos de teste através das seguintes regras de conversão:

1. Um *click* efetuado por um utilizador num determinado elemento HTML, que não seja um INPUT, é transformado numa instrução PHP que em ambiente de teste simulará esse mesmo *click*;

```
1 owa_click(dom_element_id) to I->$click(dom_element_id)
```

Listing 4.2: Exemplo de Click

2. Um *click* efetuado por um utilizador num determinado *INPUT* é transformado numa instrução PHP, que em ambiente de teste, simulará o preenchimento e posterior submissão desse mesmo formulário, com a informação que o testador anteriormente forneceu;

```
1 owa_fill(dom_element_id,value_data) to I->$fillfield(dom_element_id,value_data)
```

Listing 4.3: Exemplo de Formulário

Implementação

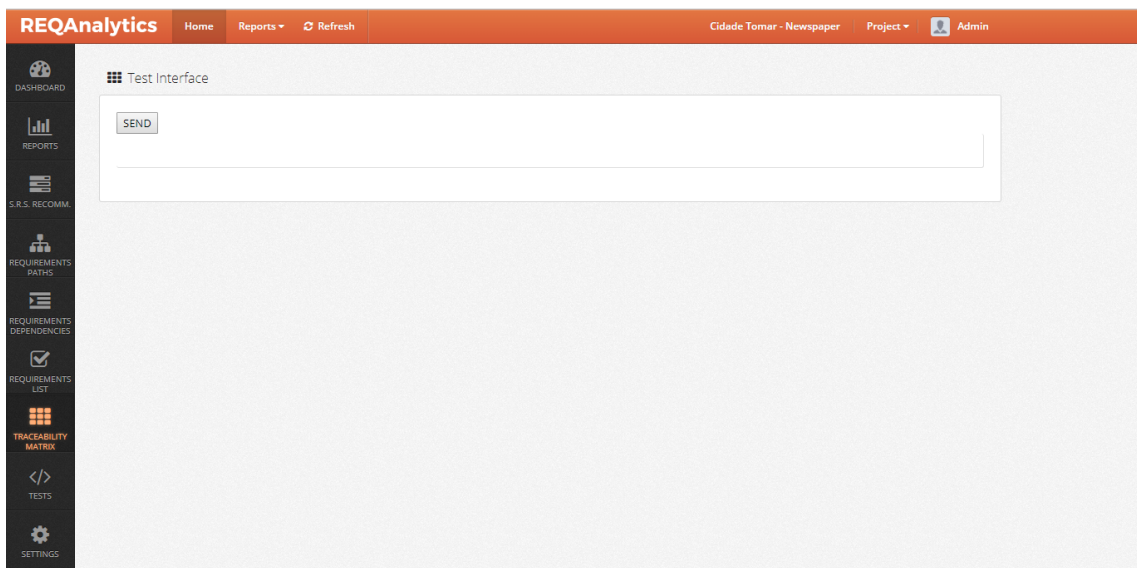


Figura 4.4: Exemplo de informação sem Inputs

Após a aplicação destas regras, são adicionadas duas verificações intermédias para cada passo de execução, de forma a dar uma maior robustez e credibilidade ao *script* gerado. Por passo de execução define-se cada instrução de PHP gerada que corresponda a uma ação anteriormente efetuada por um utilizador. A primeira verificação intermédia adicionada pretende verificar, se antes de se efetuar um *click*, o elemento HTML que se pretende premir está, de facto, presente na página em que o teste está a executar de momento. Esta verificação ocorre de várias formas para preservar a autenticidade do elemento:

1. Em primeiro lugar, é efetuada um *assert* entre o texto do elemento, representado na variável `$dom_element_text`, com o *id* desse mesmo elemento, presente em `$dom_element_id`. Caso a comparação seja bem sucedida, o teste avança para o passo seguinte, sendo que, caso contrário, o teste falha e termina nesse mesmo passo.

```
1 $I->see($dom_element_text,$dom_element_id);
```

Listing 4.4: Exemplo de verificação de elemento através do id

2. Apesar disso, em alguns casos, o elemento HTML poderá não possuir um *id* associado, ou o OWA poderá não ter conseguido capturar o *id* associado a esse elemento. Neste caso, a solução passa por efetuar uma comparação entre o texto do elemento e a sua classe associada, representada por `$dom_element_class`, sendo que as condições de sucesso na execução de teste anteriormente referidas também se aplicam neste caso.

Implementação

```
1 $I->see($dom_element_text,$dom_element_class);
```

Listing 4.5: Exemplo de verificação de elemento através da classe

3. Por último, caso o OWA não consiga capturar o *id* e a classe associadas ao elemento *HTML* em teste, é efetuada uma comparação entre o texto do elemento e a sua *tag HTML* associada, representada por `$dom_element_tag`. No entanto, esta comparação é pouco eficiente e pode afetar a integridade do elemento, visto que esta não preserva a autenticidade do elemento. Os critérios de sucesso aplicados nesta comparação é novamente o acima referido.

```
1 $I->see($dom_element_text,$dom_element_tag);
```

Listing 4.6: Exemplo de verificação de elemento através da tag

A segunda verificação intermédia tem por objetivo verificar se, após um *click* ser efetuado, a página em que o teste se encontra corresponde ao *url* de destino representado pela variável `$target_url` anteriormente recolhida. Caso esta verificação seja bem sucedida, o passo de execução é considerado válido e o teste segue para a instrução seguinte, caso contrário o teste é considerado inválido e termina nesse mesmo momento.

```
1 $I->amOnPage($target_url);
```

Listing 4.7: Exemplo de verificação de URL

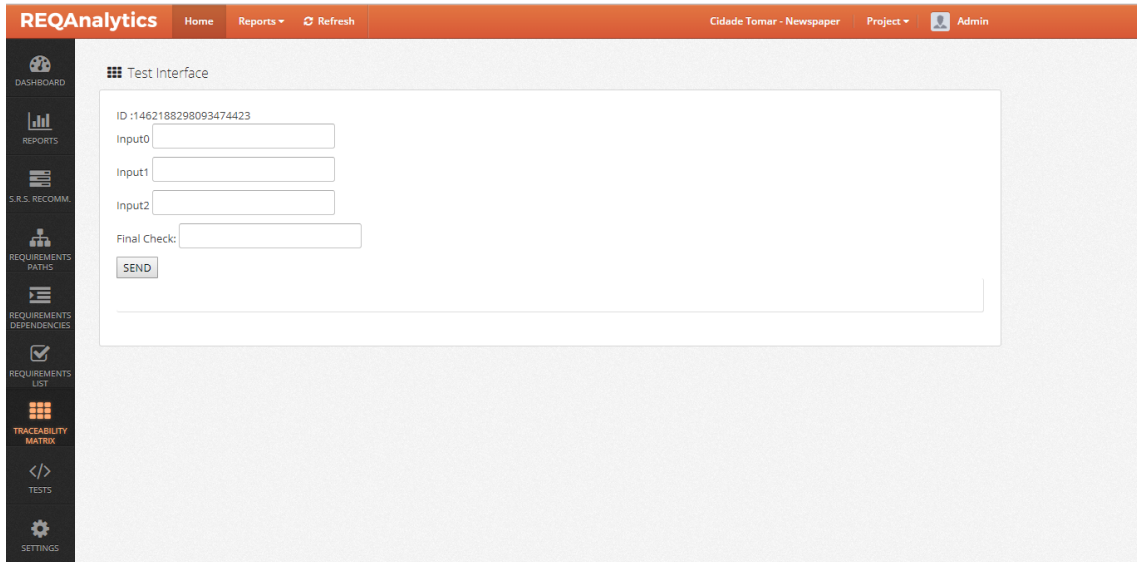
Após a conclusão deste processo, é apresentado ao testador um esboço do *script* gerado. Nesta fase, o testador pode observar o estado atual do seu *script* e efetuar o último passo necessário para a geração de um *script* de testes consistente.

```
1 $I->amOnPage('/');
2 $I->see('E-PAPER','.A');
3 $I->click('E-PAPER');
4 $I->amOnPage('http://epaper.cidadetomar.pt');
5 $I->see('CONTACTOS','.A');
6 $I->click('CONTACTOS');
7 $I->amOnPage('http://www.cidadetomar.pt/contactos');
8 $I->see('ESTATUTO EDITORIAL','.A');
9 $I->click('ESTATUTO EDITORIAL');
10 $I->amOnPage('http://www.cidadetomar.pt/estatuto');
11 $I->fillField('.texto',' ');
12 $I->see(' ');
```

Listing 4.8: Exemplo de um esboço

Implementação

Nesta última etapa é pedido ao testador que insira manualmente na interface a informação que pretende que esteja presente em cada um dos *INPUTS* de formulário existentes no teste. Esta informação será depois remetida para a função `fillField()` associada a cada um dos *INPUTS* acima referidos, e que será responsável por, em ambiente de teste, simular o processo de preenchimento e posterior submissão do formulário em questão, tal como é possível observar em [4.5](#).



The screenshot displays the REQAnalytics web application interface. At the top, there is an orange header bar with the application name 'REQAnalytics' on the left, and navigation links 'Home', 'Reports', and 'Refresh' in the center. On the right side of the header, it shows the current project 'Cidade Tomar - Newspaper', a 'Project' dropdown menu, and a user profile icon labeled 'Admin'. A dark sidebar on the left contains various navigation icons and labels: 'DASHBOARD', 'REPORTS', 'S.R.S. RECOMM.', 'REQUIREMENTS PATHS', 'REQUIREMENTS DEPENDENCIES', 'REQUIREMENTS LIST', 'TRACEABILITY MATRIX', 'TESTS', and 'SETTINGS'. The main content area is titled 'Test Interface' and contains a form with the following elements: an ID field showing 'ID :1462188298093474423', three input fields labeled 'Input0', 'Input1', and 'Input2', a 'Final Check:' label followed by a text input field, and a 'SEND' button. Below the form, there is a large, empty rectangular box.

Figura 4.5: Exemplo de formulário de introdução de informação

Além disto, é pedido ao testador que forneça à aplicação uma verificação final que terá como objetivo encontrar um elemento único ao último passo realizado, de forma a validar todos os traços de execução já realizados. Mais uma vez, caso nenhum traço de execução possua *clicks* associados a elementos do tipo INPUT, é apenas pedido ao testador que introduza uma verificação final, como é possível de observar em [4.6](#).

Implementação

REQAnalytics Home Reports Refresh Cidade Tomar - Newspaper Project Admin

DASHBOARD

REPORTS

S.R.S. RECOMM.

REQUIREMENTS PATHS

REQUIREMENTS DEPENDENCIES

REQUIREMENTS LIST

TRACEABILITY MATRIX

TESTS

SETTINGS

Test Interface

ID: 1420725271811611126

Final Check:

SEND

Figura 4.6: Exemplo de formulário só com verificação final

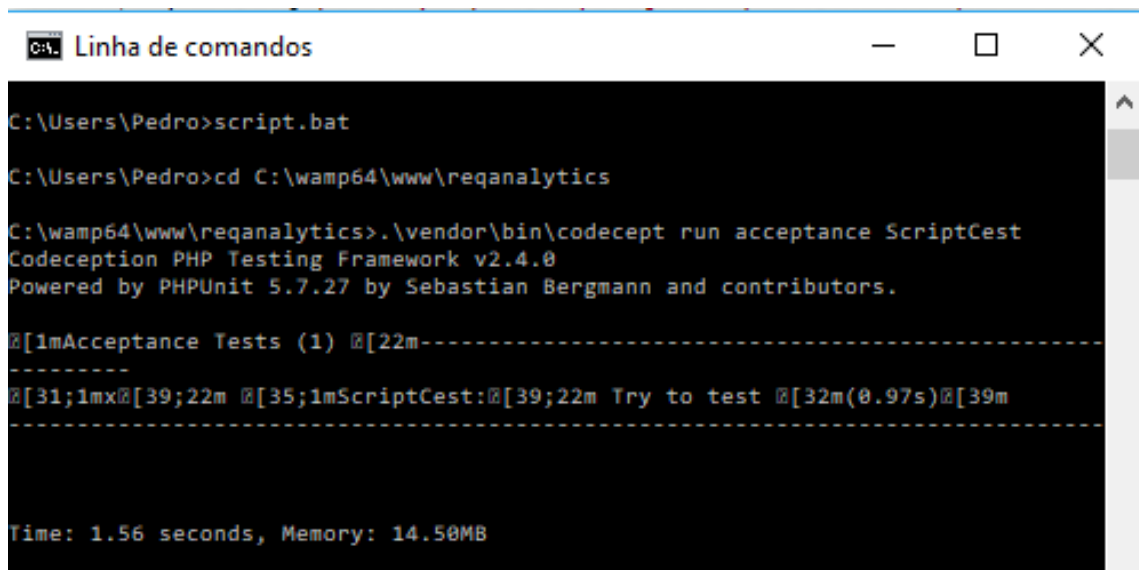
Em 4.9 observa-se *script* de testes completo gerado pela aplicação.

```
1 $I->amOnPage('/');
2 $I->see('E-PAPER', '.A');
3 $I->click('E-PAPER');
4 $I->amOnPage('http://epaper.cidadetomar.pt');
5 $I->see('CONTACTOS', '.A');
6 $I->click('CONTACTOS');
7 $I->amOnPage('http://www.cidadetomar.pt/contactos');
8 $I->see('CONTACTOS', '.A');
9 $I->click('CONTACTOS');
10 $I->amOnPage('http://www.cidadetomar.pt/contactos');
11 $I->see('ESTATUTO EDITORIAL', '.A');
12 $I->click('ESTATUTO EDITORIAL');
13 $I->amOnPage('http://www.cidadetomar.pt/estatuto');
14 $I->fillField('.texto', 'tomar');
15 $I->see('Semana Santa de Tomar');
```

Listing 4.9: Exemplo de um script completo

4.3.1 Execução do *Script* de Testes

Após a geração destes casos de testes, procede-se à execução dos mesmos em ambiente de teste, recorrendo à ferramenta *Codeception*. De forma a automatizar o processo de execução dos referidos casos de teste, foi criado o *script bash* presente em 4.10.



```

C:\Users\Pedro>script.bat

C:\Users\Pedro>cd C:\wamp64\www\reqanalytics

C:\wamp64\www\reqanalytics>.\vendor\bin\codecept run acceptance ScriptCest
Codeception PHP Testing Framework v2.4.0
Powered by PHPUnit 5.7.27 by Sebastian Bergmann and contributors.

[1mAcceptance Tests (1) [22m-----
-----
[31;1mx[39;22m [35;1mScriptCest:[39;22m Try to test [32m(0.97s)[39m
-----

Time: 1.56 seconds, Memory: 14.50MB

```

Figura 4.7: Ambiente de Teste

```

1 cd C:\wamp64\www\reqanalytics
2 .\vendor\bin\codecept run acceptance ScriptCest
3 cmd /k

```

Listing 4.10: Script de Execução dos Testes

Este conjunto de instruções permite executar automaticamente o ambiente de testes, sendo para isso apenas necessário correr a instrução `script.bat` na linha de comandos, como é possível observar em 4.7.

Paralelamente a todo este processo de geração de casos de teste, foram criados casos de teste manuais com o propósito de servirem de comparação. Esta comparação será efetuada ao nível do tempo de execução e semelhança de traços de execução. Os casos de teste exemplificados em 4.11 serão, posteriormente, executados como testes de regressão, sendo que, neste caso em particular, foram criados dois casos de testes responsáveis por testar a existência da secção desportiva e o funcionamento do sistema de pesquisa do *newspaper* da Cidade de Tomar, respetivamente.

Implementação

```
1 public function SportPageWorks(AcceptanceTester $I){
2     $I->amOnPage('/');
3     $I->click('DESPORTO');
4     $I->see('Torneio Interassociativo do Carnaval em Tomar');
5 }
6 public function SearchWorks(AcceptanceTester $I){
7     $I->amOnPage('/');
8     $I->fillField('texto','tomar');
9     $I->click('input[type="image"]');
10    $I->see('Semana Santa em Tomar');
11 }
```

Listing 4.11: Exemplo de Testes de Regressão

Capítulo 5

Resultados

O objetivo principal deste capítulo é apresentar os resultados da aplicação da abordagem proposta em três sistemas diferentes. Na secção 5.1 é apresentado o problema estudado, bem como todos os requisitos técnicos necessários para a realização desta experiência em vários tipos de dispositivos diferentes. Para além disso, é apresentada a metodologia de teste padrão a seguir aquando da realização de cada uma das experiências. De seguida, na secção 5.2 são apresentados os resultados dos três ensaios realizados em cada um dos três sistemas em estudo, sendo eles o Portal de Erasmus do IPVC, o Portal de Multimédia do IPVC e o Jornal da Cidade de Tomar, respetivamente.

Por fim, na secção 5.3 são feitas algumas referências extra aos resultados das experiências realizadas, sendo que a discussão final dos mesmos é realizada no capítulo 6.

5.1 Problema Estudado

Nesta secção serão apresentadas todas as questões que foram surgindo ao longo do desenvolvimento deste projeto de investigação. Com as respostas pretende-se retirar ilações que, no fim da experiência, permitam concluir se os resultados alcançados ao longo desta são ou não satisfatórios. As questões que foram surgindo ao longo do desenvolvimento da experiência foram as seguintes:

1. É possível extrair casos de teste a partir de informação de dados de navegação *web*?
2. Será a informação recolhida pelo OWA suficiente para produzir casos de teste consistentes?

Tal como referido em 3.2.2, de forma a validar a abordagem implementada e responder às questões acima referidas, foram conduzidos três casos de estudo diferentes. Em primeiro lugar, estudou-se a possibilidade de gerar casos de teste a partir de sessões de vários utilizadores do *Newspaper* da Cidade de Tomar. De seguida, conduziu-se um estudo bastante semelhante no Portal do IPVC (Instituto Politécnico de Viana do Castelo) e no Centro de Multimédia de Viana do Castelo.

5.1.1 Especificação Técnica

Durante a realização desta experiência, todos os ensaios realizados através das ferramentas *REQAnalytics* e *Codeception*, foram executados recorrendo a um computador ASUS com as seguintes especificações:

- Sistema Operativo: Windows 10;
- CPU: Intel CORE i7;
- Placa Gráfica: NVIDIA GEFORCE 745m;

No entanto, para proceder à execução desta experiência em qualquer outro dispositivo diferente, é necessário que o mesmo possua as seguintes especificações:

- *PHP* 5.3.29 instalado (versão mínima suportável pela ferramenta *Codeception*);
- *CURL* instalado;
- Ligação VPN à FEUP de forma a poder utilizar a ferramenta *REQAnalytics*;

5.1.2 Metodologia de Teste

De modo a obter uma maior fiabilidade e consistência dos resultados obtidos, é importante definir uma metodologia a utilizar no decorrer desta experiência. Posto isto, foi desenvolvido um plano de testes com os seguintes passos:

1. Seleção das melhores sessões a utilizar para cada um dos casos de estudo;
2. Fornecimento da informação necessária para preenchimento dos campos de texto;
3. Execução do *script* gerado em ambiente de teste;
4. Análise individual aos resultados obtidos para cada execução;
5. Comparação dos resultados obtidos com a execução dos *scripts* de teste manualmente desenvolvidos pelo utilizador, para cada caso de estudo em específico;
6. Análise das conclusões retiradas com todos os resultados obtidos ao longo da experiência.

Devido ao elevado número de sessões de utilizadores para cada *website* em estudo, existente na base de dados, é necessário realizar uma filtragem, que permita obter um conjunto de sessões com dados de navegação consistentes, as quais, por sua vez, permitam realizar uma simulação o mais aproximada possível, em ambiente de teste, de uma sessão real de utilizador. O primeiro passo desta seleção passa por calcular o número de *clicks* executados em cada sessão existente num determinado *website*. Este cálculo é efetuado executando a *query* presente em 5.1.

Resultados

```
1 SELECT $visitor_id, COUNT($visitor_id) AS Occurences FROM `owa_click`  
2 WHERE site_id= $website_id  
3 GROUP BY $visitor_id
```

Listing 5.1: Exemplo de query de cálculo

Como resultado desta instrução é obtida uma vista, em que é possível observar, para cada *id* de utilizador existente, o número de ocorrências que o mesmo possui num determinado sistema em estudo. Em 5.1, é possível observar um excerto dos resultados obtidos após a execução desta mesma *query* aplicada ao *website* de Multimédia de Viana do Castelo.

visitor_id	Occurences
1462188298093474423	55
1462199305394109826	5
1462291192434536672	12
1462292871318766279	87
1462452059293740421	2
1462783227824532269	52
1462868460823355655	1
1462891596367290324	1
1463129802452704361	5
1463145890677976619	6
1463154283288572427	81
1466161544100414301	14

Figura 5.1: Extrato das Ocorrências de cada id de utilizador no *site* de Multimédia

Após este passo, são seleccionadas algumas sessões que contenham um número de *clicks* associados, compreendido entre dez e trinta e cinco. Seleccionaram-se estes intervalos de valor, pois dez *clicks* é um valor mínimo aceitável para simular uma sessão com alguma consistência. Para valor máximo de *clicks* optou-se por trinta e cinco, de forma a não gerar um *script* com demasiada complexidade de análise para o testador e para não ter um tempo de execução demasiado elevado em ambiente de teste.

De seguida, cada uma das sessões é analisada individualmente, de forma a perceber se contém informação suficiente para produzir um caso de teste válido. Para cada sessão verifica-se se o OWA conseguiu recolher todo o tipo de informação associada a um determinado *click* que permita reproduzi-lo em ambiente de teste, ou seja, verificar se para cada sessão foi recolhido o *id*, a classe, a *tag* e o texto associado ao elemento *HTML* *clickado*. A sessão é considerada válida, desde que, pelo menos, para cada um dos *clicks* associados, seja possível reunir pelo menos um dos elementos acima referidos. No caso de existirem sessões que possuam *clicks* sem nenhum elemento associado, as mesmas ainda podem ser consideradas válidas, desde que o número de

Resultados

dom_element_id	dom_element_tag	dom_element_text	dom_element_class
(not set)	A	Em Direto	(not set)
(not set)	A	Em Direto	(not set)
(not set)	A	Arquivo	(not set)
(not set)	A	última »	(not set)
(not set)	A	3	(not set)
(not set)	IMG		(not set)
(not set)	A	4	(not set)
(not set)	A	Em Direto	(not set)
(not set)	A	Arquivo	(not set)
(not set)	A	última »	(not set)
(not set)	A	3	(not set)
(not set)	A	4	(not set)
(not set)	A	3	(not set)
(not set)	A	2	(not set)
(not set)	A	1	(not set)
(not set)	A	Em Direto	(not set)
(not set)	A	Galeria	(not set)

Figura 5.2: Exemplo de uma amostra válida

clicks sem informação não seja superior a três. Caso na amostra de sessões recolhida não seja possível encontrar dez sessões que cumpram os critérios acima referidos, é feita uma nova seleção de sessões de utilizadores. Em 5.2 é apresentada uma amostra de sessão considerada válida, porque, apesar de conter apenas informação relacionada com a *tag* e o texto do elemento *HTML* associado aos *clicks* em questão, cumpre com os requisitos mínimos necessários.

Pelo contrário, em 5.3 é apresentada uma amostra considerada inválida, visto que não apresenta informação suficiente relacionada com os *clicks* efetuados nos elementos *HTML* associados.

Após ser recolhido um número de de sessões suficientes, é introduzido na ferramenta *REQA-nalytics* o id de cada uma dessas sessões, de forma a proceder à extração dos passos de teste para cada uma das sessões em análise. No caso dessas sessões possuírem elementos do tipo *INPUT*, tal como referido em 4, é necessário fornecer informação extra que será utilizada para preencher os campos de texto existentes.

dom_element_id	dom_element_tag	dom_element_text	dom_element_class
(not set)	IMG	View larger image	(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	IMG		(not set)
(not set)	DIV		(not set)

Figura 5.3: Exemplo de uma amostra inválida

Posteriormente à geração dos *scripts* de testes correspondentes a cada uma das sessões em estudo, procede-se à execução dos mesmos em ambiente de teste. Após esta execução, é feita uma análise individual aos resultados provenientes da mesma. Nesta análise são estudados os seguintes pontos:

1. **Consistência do Sistema.** Em alguns casos de teste, com elementos do tipo *INPUT*, introduziu-se informação errada, de forma a testar a segurança do sistema em estudo. Por exemplo, em casos em que fosse pedida informação necessária para autenticação no sistema, foram introduzidas combinações de *usernames* e *passwords* erradas com o intuito de verificar se o sistema permitia a autenticação em situações desse género.
2. **Fiabilidade dos Casos de Teste gerados.** Ao comparar cada um dos *scripts* de testes gerados, para cada um dos sistemas, com o conjunto de casos de testes criado manualmente através da análise dos caminhos mais frequentes em cada um dos *websites*, é possível concluir se a informação de navegação recolhida transmite traços de execução reais realizados pela maioria dos utilizadores dos sistemas em análise.
3. **Qualidade da Informação Recolhida.** Conjugando a informação proveniente da análise das sessões seleccionadas e da execução dos *scripts* em ambiente de teste, podem ser extraídas conclusões sobre a qualidade da informação recolhida pelo OWA.

5.2 Casos de Estudo

5.2.1 Instituto Politécnico de Viana do Castelo

Neste caso de estudo, são analisados os dados de navegação resultantes da secção de ERASMUS do IPVC. Nestes dados, estão presentes as várias ações realizadas por cada estudante aquando da sua interação com o sistema. Após ser efetuado o cálculo do número de *clicks* efetuados por cada utilizador durante a sua sessão, foram sendo seleccionadas e estudadas várias dessas mesmas sessões, de forma a obter um conjunto de dez amostras válidas para extração de casos de teste.

No entanto, e neste caso de estudo em particular, foi necessária uma pesquisa bastante exaustiva, de forma a conseguir obter um conjunto de amostras válidas que perfizesse o número desejado, como é possível de observar em 5.1.

Tabela 5.1: Pesquisa de sessões válidas

Erasmus IPVC	Nº de pesquisas efetuadas	Amostras Válidas	Amostras Inválidas
	60	10	50

Após a execução de cada um dos *scripts* gerados, a partir das amostras válidas recolhidas, em ambiente de teste, conclui-se que dos dez, apenas um não correspondia a uma simulação válida de uma sessão executada por um utilizador no sistema, visto que nesse caso de teste é apenas executado um ciclo infinito de *clicks* num determinado elemento *HTML*.

Resultados

Nos outros casos estudados, foram encontradas várias sessões que simulavam interações reais com o portal de ERASMUS do IPVC, presentes nos caminhos mais comuns do sistema, como por exemplo: submissões de formulários de candidatura, consulta de informações sobre faculdade parceiras, consultas de informações sobre candidaturas, entre outros.

5.2.1.1 Qualidade dos Resultados Obtidos

Em relação a este caso de estudo em particular, conclui-se que a informação de dados de navegação presente na base de dados era de fraca qualidade, uma vez que continha uma quantidade bastante elevada de amostras inválidas, que não permitiam a extração de casos de teste consistentes. No entanto, das amostras válidas conseguidas, a maior parte (9 em 10) consegue simular, em ambiente de teste, hipóteses fiáveis e consistentes de sessões de utilizadores, o que, para este caso em específico, leva a afirmar que é possível extrair casos de teste fiáveis e consistentes a partir de dados de navegação.

A comparação destes casos de teste com os testes de regressão desenvolvidos manualmente validou a conclusão acima extraída, uma vez que conjugados, todos os casos de teste gerados cobrem quase na totalidade as ações executadas nos casos de teste desenvolvidos manualmente.

5.2.2 Multimédia IPVC

De seguida, procedeu-se à análise dos dados de navegação referentes à secção de Multimédia do Portal do IPVC. Novamente, nestes dados estão presentes todas interações existentes entre utilizador e sistema representadas pelos *clicks* efetuados em determinados elementos do *website*. Tal como no caso de estudo anterior, e seguindo a metodologia de teste proposta, o primeiro passo desta experiência foi o cálculo do número de *clicks* efetuados por cada utilizador durante a sua sessão no *website* em estudo. De seguida, foram selecionadas, para estudo individual, várias amostras de sessões de utilizador com valores de *clicks* compreendidos no intervalo referido na metodologia proposta.

Para este caso de estudo em particular não foi necessário executar uma pesquisa tão aprofundada para encontrar um número de amostras suficientes para executar em ambiente de teste, como é possível observar em 5.2.

Tabela 5.2: Pesquisa de sessões válidas

Multimédia IPVC	Nº de pesquisas efetuadas	Amostras Válidas	Amostras Inválidas
	34	10	24

Apesar disso, e, após as amostras recolhidas em ambiente de teste serem executadas, observa-se que em 60% das amostras recolhidas não é possível obter uma simulação real de uma sessão de utilizador neste sistema em específico. Este aspeto está relacionado com o facto de que nestas sessões ocorrerem muitos ciclos infinitos, em que o teste apenas prime os títulos das secções existentes na página principal.

5.2.2.1 Qualidade dos Resultados Obtidos

Sobre os resultados obtidos com este caso de estudo, concluiu-se que, para este sistema, através da amostra recolhida, não foi possível obter um número suficiente de casos de teste consistentes que pudessem ser validados pelos testes de regressão desenvolvidos manualmente pelo utilizador. Esta conclusão está relacionada com o facto de que, apesar de existirem quatro *scripts* de testes que, em ambiente de teste, simulam uma sessão de utilizador aproximada da realidade, a conjugação destes mesmo *scripts* não cobre o número de casos de teste suficientes, do que o *script* desenvolvido manualmente cobre.

No entanto, nesta experiência, os dados de navegação recolhidos pelo OWA permitiram obter um rácio entre amostras válidas e inválidas bastante menor do que no caso de estudo anterior, o que permite afirmar que para este sistema os dados de navegação possuem uma maior qualidade, apesar de perderem em aspetos como fiabilidade e consistência, quando comparados com o caso de estudo anterior.

5.2.3 Newspaper Cidade de Tomar

Por último, foi executada uma análise aos dados de navegação provenientes do Jornal da Cidade de Tomar. Tal como nos casos de estudo anteriores, os dados recolhidos pelo OWA contêm todos os *clicks* realizados por todos os utilizadores, que utilizaram este sistema durante o período em que o mesmo esteve sujeito a análise. Novamente, e seguindo a metodologia de teste proposta, procedeu-se ao cálculo do número de *clicks* efetuados por cada utilizador durante a sua sessão no sistema em análise. Posteriormente, foram selecionadas, para análise individual, várias amostras presentes na base de dados.

Neste caso de estudo em específico, tal como no anterior, não foi necessário efetuar uma pesquisa muito extensa, de forma a encontrar um conjunto suficiente de amostras de utilizadores válidas para análise individual, tal como é possível observar em [5.3](#)

Tabela 5.3: Pesquisa de sessões válidas

Cidade Tomar	Nº de pesquisas efetuadas	Amostras Válidas	Amostras Inválidas
	32	10	22

Das amostras recolhidas e, após a sua execução em ambiente de teste, observou-se que 90% das mesmas simulava com precisão, em ambiente de teste, uma sessão de utilizador hipoteticamente real, o que transmite uma melhor consistência aos resultados obtidos neste estudo.

5.2.3.1 Qualidade dos Resultados

Após uma análise aos resultados obtidos com este caso de estudo, conclui-se que, para este sistema foi possível obter um número elevado de casos de teste consistentes, passíveis de serem validados pelos testes de regressão desenvolvidos manualmente. Para este sistema em específico, testou-se também a segurança do mesmo, introduzindo um conjunto de credenciais erradas

aquando da simulação de uma ação de *Login*, observando-se que o sistema não permitiu a autenticação do testador neste momento, o que leva a afirmar que se está perante um projeto consistente.

Sobre a recolha de dados de navegação pelo OWA, e apesar do rácio amostras válidas/amostras inválidas ser novamente pequeno, em alguns casos torna-se mais complicado encontrar amostras que contenham todos elementos *HTML* associados a um determinado *click*, o que dificulta a tarefa de extração de casos de teste válidos.

5.3 Conclusões

Nesta secção, foram apresentados os resultados da aplicação da abordagem proposta, a três casos de estudo diferentes. Cada um destes casos de estudo foi conduzido com o objetivo de comprovar se seria possível gerar casos de teste consistentes a partir de informação de dados de navegação *web*. No Anexo A é possível observar os *scripts* de testes gerados para cada um dos conjuntos de amostras selecionado para cada um dos casos de estudo, bem como os resultados da sua execução em ambiente de teste. Para além disso, é também possível observar o *script* de testes de regressão desenvolvido manualmente pelo testador para cada um dos sistemas estudados.

Conclui-se também que, dos três casos de estudo conduzidos, o *Newspaper* da Cidade de Tomar foi o que produziu casos de teste mais consistentes. Isto deve-se ao facto de, os dados de navegação recolhidos para este sistema, serem mais completos quando comparados com os outros.

Por outro lado, os dados de navegação recolhidos durante a condução do caso de estudo da secção de Multimédia do IPVC não permitiram produzir casos de teste com a consistência e fiabilidade desejadas como é possível de observar na tabela 5.4.

Tabela 5.4: Comparação de Resultados

	Nº Testes Executados	Nº Testes Consistentes	% Cobertura de Funcionalidades (Estimada)
Erasmus IPVC	10	9	70
Multimédia IPVC	10	6	45
Cidade de Tomar	10	9	98

As respostas para as questões levantadas durante a realização desta experiência, bem como as conclusões finais do trabalho desenvolvido ao longo desta Dissertação, estão disponíveis no Capítulo 6.

Capítulo 6

Discussão e Conclusão

Os objetivos principais deste capítulo passam por responder às questões que foram surgindo com o decorrer da experiência, tendo em conta os resultados obtidos nos casos de estudo apresentados no capítulo 5 e, posteriormente, efetuar um balanço dos resultados obtidos com este trabalho de investigação, propondo também, algum trabalho futuro a realizar nesta área de investigação. Estes objetivos propostos são abordados nas secções 6.1 e 6.2 respetivamente.

6.1 Discussão

6.1.1 Q1 - É possível extrair casos de teste a partir de informação de dados de navegação *web*?

Tendo em conta os resultados obtidos nos casos de estudo do capítulo 5, conclui-se que é possível gerar casos de estudo a partir de dados de navegação *web* com relativa facilidade. Em cada um dos 3 casos de estudo conduzidos, foi possível extrair, com sucesso, vários *scripts* de casos de teste, executáveis em ambiente de teste, mesmo tendo em conta as limitações técnicas referidas.

6.1.2 Q2 - Será a informação recolhida pelo OWA suficiente para produzir casos de teste consistentes?

Após uma análise aos resultados dos casos de estudo conduzidos, conclui-se que, apesar de existirem casos de teste gerados capazes de simularem sessões hipoteticamente reais, em ambiente de teste, a informação recolhida pelo OWA possui várias restrições.

Ao efetuar pesquisas por sessões válidas na base de dados, observou-se que, na maioria das sessões existentes, a ferramenta não capturava toda a informação associada a um determinado *click* num elemento. Em muitos destes casos, não estavam presentes, principalmente, os *ids* e as classes dos elementos HTML *clickados*, tornando assim mais difícil a simulação de um determinado *click* em ambiente de teste.

Por outro lado, observou-se que o OWA não recolhe alguma informação associada ao processo de preenchimento de formulários necessária para simular o processo de preenchimento de campos de texto, desses mesmos formulários, em ambiente de teste.

Assim sendo, torna-se impossível simular, com a consistência necessária, situações de teste que passem pelo preenchimento e posterior submissão de um determinado formulário, o que, conjugado com as falhas na recolha de informação de certos tipos de elementos associados a *clicks*, leva a afirmar que a informação recolhida pelo OWA não é suficiente para produzir casos de teste consistentes.

6.2 Conclusão e Trabalho Futuro

Com o trabalho desenvolvido ao longo desta Dissertação, tornou-se bastante óbvio que uma boa gestão e validação de requisitos é, cada vez mais, um fator decisivo na obtenção de uma maior consistência e garantia de qualidade em sistemas de *software*.

Para uma maior qualidade na execução deste processo de gestão e validação de requisitos, concluiu-se que é bastante importante a aplicação de uma boa metodologia de teste que permita a automação do processo de testes a *software*.

Assim sendo, neste trabalho propôs-se uma nova abordagem, que passa pela extração de casos de teste automatizados a partir de dados de navegação *web*. Estes casos de teste foram posteriormente executados, de forma a analisar a sua consistência e fiabilidade. Após a análise efetuada ao comportamento destes mesmos casos de teste, em ambiente de teste, observou-se que, apesar de existirem casos de teste fiáveis, ainda existem aspetos a melhorar nesta área, de forma a produzir casos de testes ainda mais consistentes capazes de simularem ao máximo uma sessão real de utilizador.

Posto isto, propõe-se um desenvolvimento de uma nova ferramenta de *Web Analytics*, que seja capaz de efetuar uma recolha de dados mais profunda, de forma a que seja possível obter dados de navegação mais consistentes. Por outro lado, seria também interessante a integração da ferramenta *Codeception* com o *Selenium*, de forma a que o testador possa visualizar a execução destes mesmos *scripts* de teste, aumentando assim ainda mais a interação entre testador e sistema.

Referências

- [AD97] Larry Apfelbaum e John Doyle. Model Based Testing. *Proceedings of the 10th International Software Quality Week*, pages 1–14, 1997.
- [AOV16] Paul Ammann, Jeff Offutt e Instructor Version. Introduction to Software Testing Edition 2 Paul Ammann and Jeff Offutt Instructor Version. pages 2002–2009, 2016.
- [Ban11] Ansuman Banerjee. Requirement evolution management: A systematic approach. *2011 IEEE Computer Society Annual Symposium on VLSI*, pages 150–155, Dec 2011.
- [Ber07] Antonia Bertolino. Software Testing Research: Achievements, Challenges, Dreams. *Future of Software Engineering*, (September):85–103, 2007. URL: [http://portal.acm.org/citation.cfm?id=1253532.1254712{%&coll=portal{%&dl=ACM{%&CFID=27180219{%&CFTOKEN=34816711,](http://portal.acm.org/citation.cfm?id=1253532.1254712{%&coll=portal{%&dl=ACM{%&CFID=27180219{%&CFTOKEN=34816711, doi:10.1109/FOSE.2007.25.) doi:10.1109/FOSE.2007.25.
- [BS97a] Marko Balabanovic e Yoav Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM* 40 (1997), pages 66–72, 1997.
- [BS97b] Marko Balabanovic e Yoav Shoham. Information extractionn. *Communications of the ACM* 39 (1996), pages 66–72, 1997.
- [CB] andWilliam Cohen Chumki Basu, Haym Hirsh. *Recommendation as Classification: Using Social and Content-Based Information in Recommendation*.
- [CHCH09] Carlos Castro-Herrera e Jane Cleland-Huang. A machine learning approach for identifying expert stakeholders. *2009 Second International Workshop on Managing Requirements Knowledge*, pages 45–49, Sep 2009.
- [CHCH10] Carlos Castro-Herrera e Jane Cleland-Huang. Utilizing recommender systems to support software requirements elicitation. *Proceedings of the 2nd International Workshop on Recommendation Systems for Software Engineering - RSSE '10 (New York, New York, USA)*, pages 6–10, May 2010.
- [CoE] CoEST. *CoEST: Center of excellence for software traceability*.
- [CPFA10] Marco Cunha, Ana C.R. Paiva, Hugo Sereno Ferreira e Rui Abreu. PETTool: A pattern-based GUI testing tool. *ICSTE 2010 - 2010 2nd International Conference on Software Technology and Engineering, Proceedings*, 1(November), 2010. doi:10.1109/ICSTE.2010.5608882.
- [DCH10] Alex Dekhtyar David Cuddeback e Jane Hayes. Automated requirements traceability: The study of human analysts. *2010 18th IEEE International Requirements Engineering Conference*, pages 231–240, Sep 2010.

REFERÊNCIAS

- [DDgP] Lakshminarayanan Vaidyanathasamy Daniela Damian, James Chisan e Yo gendra Pal. *An Industrial Case Study of the Impact of Requirements Engineering on Downstream Development*.
- [DM03] H Dai e Bamshad Mobasher. A road map to more effective web personalization: Integrating domain knowledge with web usage mining. *Proceedings Of The International Conference On Internet Computing*, pages 1–8, 2003.
- [Don] Qi Dong. *Predicting and managing system interactions at early phase of the product development process*.
- [Dug08] Gaurav Duggal. Understanding regression testing techniques. *2nd National Conference on Challenges and Opportunities in Information Technology*, (1), 2008.
- [GA15] S. Gowri e G. S. Anandha Mala. Efficacious IR system for investigation in digital textual data. *Indian Journal of Science and Technology*, 8(12):290–293, 2015. doi:10.17485/ijst/2015/v8i.
- [GP16a] J. Esparteiro Garcia e Ana C. R. Paiva. An automated approach for requirements specification maintenance. pages 827–833, 2016.
- [GP16b] Jorge Esparteiro Garcia e Ana C.R. Paiva. REQAnalytics: A recommender system for requirements maintenance. *International Journal of Software Engineering and its Applications*, 10(1):129–140, 2016. doi:10.14257/ijseia.2016.10.1.13.
- [GP18a] Jorge Esparteiro Garcia e Ana C. R. Paiva. Manage software requirements specification using web analytics data. In Álvaro Rocha, Hojjat Adeli, Luís Paulo Reis e Sandra Costanzo, editors, *Trends and Advances in Information Systems and Technologies*, pages 257–266, Cham, 2018. Springer International Publishing.
- [GP18b] Jorge Esparteiro Garcia e Ana C R Paiva. *Manage Software Requirements Specification Using Web Analytics Data*, volume 746. Springer International Publishing, 2018. URL: <http://link.springer.com/10.1007/978-3-319-77712-2>.
- [Gro95] Standish Group. *Chaos Report*. Group, 1995.
- [KGH10] Massila Kamalrudin, John Grundy e John Hosking. Tool Support For Essential Use Cases To Better Capture Software Requirements. *ACM The International Conference on Automated Software Engineering, ASE 2010*, (January):255–264, 2010. URL: <http://portal.acm.org/citation.cfm?doid=1858996.1859047>, doi:10.1145/1858996.1859047.
- [KNG⁺14] Massila Kamalrudin, M. Noraiza, John Grundy, John Hosking e Mark Robinson. Automatic acceptance test case generation from essential use cases. *Frontiers in Artificial Intelligence and Applications*, 265(October):246–255, 2014. doi:10.3233/978-1-61499-434-3-246.
- [KSK12] Lakhwinder Kumar, Hardeep Singh e Ramandeep Kaur. Web analytics and metrics: A Survey. *Proceedings of the International Conference on Advances in Computing, Communications and Informatics - ICACCI '12*, page 966, 2012. doi:10.1145/2345396.2345552.

REFERÊNCIAS

- [MA15] Bhupendra Kumar Malviya e Jitendra Agrawal. A study on web usage mining theory and applications. *2015 Fifth International Conference on Communication Systems and Network Technologies*, pages 935–939, Apr 2015.
- [McF] Chris McFadden. *Optimizing the Online Business Channel with Web Analytics*.
- [MP13] Tiago Monteiro e Ana C.R. Paiva. Pattern Based GUI Testing Modeling Environment. *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation Workshops*, pages 140–143, 2013. URL: <http://ieeexplore.ieee.org/document/6571623/>.
- [MP14a] M. L. M. Rodrigo Moreira e Ana C. R. Paiva. Towards a Pattern Language for Model-Based GUI Testing. *19th European Conference on Pattern Languages of Programs (EuroPLoP 2014)*, 2014. doi:10.1145/2721956.2721972.
- [MP14b] Rodrigo M.L.M. Moreira e Ana C.R. Paiva. PBGT tool. *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering - ASE '14*, (September):863–866, 2014. doi:10.1145/2642937.2648618.
- [MPM13] Rodrigo M L M Moreira, Ana C R Paiva e Atif Memon. A pattern-based approach for GUI modeling and testing. *2013 IEEE 24th International Symposium on Software Reliability Engineering, ISSRE 2013*, (October):288–297, 2013.
- [MRE12] Jamshaid G. Mohebzada, Guenther Ruhe e Armin Eberlein. Systematic mapping of recommendation systems for requirements engineering. *2012 International Conference on Software and System Process (ICSSP)*, pages 200–209, 2012. URL: <http://ieeexplore.ieee.org/document/6225965/>, doi:10.1109/ICSSP.2012.6225965.
- [MT09] Walid Maalej e Anil Kumar Thurimella. Towards a Research Agenda for Recommendation Systems in Requirements Engineering. *2009 Second International Workshop on Managing Requirements Knowledge*, (section 3):32–39, 2009. URL: <http://ieeexplore.ieee.org/document/5457346/>.
- [Mye04] Glenford Myers. *The Art of Software Testing, Second edition*, volume 15. 2004. arXiv:9809069v1, doi:10.1002/stvr.322.
- [NPF14] Miguel Nabuco, Ana C.R. Paiva e João Pascoal Faria. Inferring user interface patterns from execution traces of web applications. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8583 LNCS(PART 5):311–326, 2014.
- [PB07] Michael J Pazzani e Daniel Billsus. Content-based recommendation systems. *The adaptive web*, pages 325–341, 2007. arXiv:9780201398298.
- [Poh10] Klaus Pohl. *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer Publishing Company, Incorporated, 1st edition, 2010.
- [PV17] Ana C. R. Paiva e Liliana Vilela. Multidimensional test coverage analysis: Paradigm-cov tool. *Cluster Computing*, 20(1):633–649, Mar 2017. URL: <https://doi.org/10.1007/s10586-017-0728-4>, doi:10.1007/s10586-017-0728-4.

REFERÊNCIAS

- [SFG99] H. Sharp, A. Finkelstein e G. Galal. Stakeholder identification in the requirements engineering process. *Proceedings. Tenth International Workshop on Database and Expert Systems Applications. DEXA 99*, pages 387–391, 1999. URL: <http://ieeexplore.ieee.org/document/795198/>, doi:10.1109/DEXA.1999.795198.
- [Som04] Ian Sommerville. *Software Engineering*. 7th edt edition, 2004.
- [Som07] Ian Sommerville. *Software Engineering*. 2007.
- [SS10] Brijendra Singh e Hemant Kumar Singh. Web data mining research: A survey. *2010 IEEE International Conference on Computational Intelligence and Computing Research*, page 1 and 40, Dec 2010.
- [VMKR13] Chintan R. Varnagar, Nirali N. Madhak, Trupti M. Kodinariya e Jayesh N. Rathod. Web usage mining: A review on process, methods and techniques. *2013 International Conference on Information Communication and Embedded Systems (ICICES)*, pages 40–46, 2013. URL: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6508399>, doi:10.1109/ICICES.2013.6508399.
- [YH07] S Yoo e M Harman. Regression Testing Minimisation, Selection and Prioritisation : A Survey. *Test. Verif. Reliab*, 00:1–7, 2007. doi:10.1002/000.

Anexo A

Scripts de Teste

Neste anexo são apresentados todos os *scripts* de teste gerados para cada sessão de utilizador selecionada para avaliação. De seguida, para cada caso de estudo efetuado, é apresentado o *script* de teste manualmente desenvolvido pelo testador, responsável por cobrir todos os cenários de teste possíveis em cada um dos sistemas.

Por fim, é possível observar relatórios de execução, onde está presente toda a informação relacionada com a execução destes mesmos *script* de teste.

A.1 Caso de Estudo I - Portal de Erasmus do IPVC

Para a sessão com o id **1476692099050467278** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('UNIVERSITA DELLA VALLE D'AOSTA - Italy');
8         $I->click('UNIVERSITA DELLA VALLE D'AOSTA - Italy');
9         $I->amOnPage('http://www.univda.it/context_with_sublink.jsp?ID_LINK=360&area=6');
10        $I->see('Universitat de Valencia');
11        $I->click('Universitat de Valencia');
12        $I->amOnPage('http://www.uv.es/en');
13        $I->see('Universitat de Valencia');
14        $I->click('Universitat de Valencia');
15        $I->amOnPage('http://www.uv.es/en');
16        $I->see('Apply', '.form-submit');
17        $I->click('Apply');
18        $I->amOnPage('');
19        $I->fillField('INPUT', 'Notas');
20        $I->see('Universidades parceiras');
21        $I->click('Universidades parceiras');
```

Scripts de Teste

```
22 $I->amOnPage('http://internacional.ipvc.pt/pt/univparc');
23 $I->see('Erasmus ');
24 $I->click('Erasmus ');
25 $I->amOnPage('http://internacional.ipvc.pt/pt/node/25');
26 $I->see(' Aluno');
27 }
28 }
```

Listing A.1: Script gerado para o id 1476692099050467278

Para a sessão com o id **1483376669883908696** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Apply', '.form-submit');
8         $I->click('Apply');
9         $I->amOnPage('/');
10        $I->fillField('INPUT', 'Aluno');
11        $I->see(' Turma');
12    }
13 }
```

Listing A.2: Script gerado para o id 1483376669883908696

Para a sessão com o id **1483797676063236883** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Erasmus Mundus');
8         $I->click('Erasmus Mundus');
9         $I->amOnPage('http://internacional.ipvc.pt/pt/node/73');
10        $I->see('Erasmus ');
11        $I->click('Erasmus ');
12        $I->amOnPage('http://internacional.ipvc.pt/pt/node/25');
13        $I->see('Políticas de Estratégia Europeias');
14        $I->click('Políticas de Estratégia Europeias');
15        $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/
        politica_estrategia_europeia_.pdf');
16        $I->see('Programas Internacionais');
17        $I->click('Programas Internacionais');
```

Scripts de Teste

```
18 $I->amOnPage('http://internacional.ipvc.pt/pt/node/15');
19     $I->see(' Cadeira');
20 }
21 }
```

Listing A.3: Script gerado para o id 1483797676063236883

Para a sessão com o id **1483802647412483403** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Tabela de Bolsas de Mobilidade - Estudos 2015/2016');
8         $I->click('Tabela de Bolsas de Mobilidade - Estudos 2015/2016');
9         $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/ERASMUS+
10             _BolsasKA103_2015_Jul2015.pdf');
11         $I->see('Despacho n 10973-D/2014 de 27 de agosto');
12         $I->click('Despacho n 10973-D/2014 de 27 de agosto');
13         $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/
14             Despacho_10973_D_14.pdf');
15         $I->see('Tabela de Bolsas de Mobilidade - Estudos 2015/2016');
16         $I->click('Tabela de Bolsas de Mobilidade - Estudos 2015/2016');
17         $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/ERASMUS+
18             _BolsasKA103_2015_Jul2015.pdf');
19         $I->see('Erasmus ');
20         $I->click('Erasmus ');
21         $I->amOnPage('http://internacional.ipvc.pt/pt/node/25');
22         $I->see('Programs');
23         $I->click('Programs');
24         $I->see(' Janela');
```

Listing A.4: Script gerado para o id 1483802647412483403

Para a sessão com o id **1483807711999921387** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('International Student Guide');
8         $I->click('International Student Guide');
```

Scripts de Teste

```
9      $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/International
      Students Guide compressed.pdf');
10     $I->see('Recognition of Qualifications - Guide for foreigners');
11     $I->click('Recognition of Qualifications - Guide for foreigners');
12     $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/Reconhecimento
      de qualificacoes_EN.pdf');
13     $I->see('Guide for Incoming Students and Staff');
14     $I->click('Guide for Incoming Students and Staff');
15     $I->amOnPage('http://internacional.ipvc.pt/en/node/494');
16     $I->see('Guide for Incoming Students and Staff');
17     $I->click('Guide for Incoming Students and Staff');
18     $I->amOnPage('http://internacional.ipvc.pt/en/node/494');
19     $I->see('Guide for Incoming Students and Staff');
20     $I->click('Guide for Incoming Students and Staff');
21     $I->amOnPage('http://internacional.ipvc.pt/en/node/494');
22     $I->see('Guide for Incoming Students and Staff');
23     $I->click('Guide for Incoming Students and Staff');
24     $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/Practical Guide
      incomig students_VF_portal.pdf');
25     $I->see(' Porta');
26 }
27 }
```

Listing A.5: Script gerado para o id 1483807711999921387

Para a sessão com o id **1483826005131239545** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage(' ');
7         $I->see('Guide for Incoming Students and Staff');
8         $I->click('Guide for Incoming Students and Staff');
9         $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/Practical Guide
      incomig students_VF_portal.pdf');
10        $I->see('Learning Agreement ');
11        $I->click('Learning Agreement ');
12        $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/
      Learning_Agreement_for_Studies_EN.docx');
13        $I->see(' Agreement');
14    }
15 }
```

Listing A.6: Script gerado para o id 1483826005131239545

Para a sessão com o id **1483837027424137557** foi gerado o seguinte *script*:

Scripts de Teste

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Critérios de seriação para atribuição de bolsa de mobilidade estudante
            2014-2020');
8         $I->click('Critérios de seriação para atribuição de bolsa de mobilidade
            estudante 2014-2020');
9         $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/Criterios
            seriacao para atribuição de bolsas de mobilidade_2015_2020.pdf');
10        $I->see('Critérios de seleção');
11        $I->click('Critérios de seleção');
12        $I->amOnPage('http://internacional.ipvc.pt/pt/node/435');
13        $I->see('Admissão');
14        $I->click('Admissão');
15        $I->amOnPage('http://internacional.ipvc.pt/pt/node/27');
16        $I->see('Admissão');
17    }
18 }
```

Listing A.7: Script gerado para o id 1483837027424137557

Para a sessão com o id **1483895776104055303** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Guide for Incoming Students and Staff');
8         $I->click('Guide for Incoming Students and Staff');
9         $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/Practical Guide
            incomig students_VF_portal.pdf');
10        $I->see('Recognition of Qualifications - Guide for foreigners');
11        $I->click('Recognition of Qualifications - Guide for foreigners');
12        $I->amOnPage('http://internacional.ipvc.pt/sites/default/files/Reconhecimento
            de qualificacoes_EN.pdf');
13        $I->see('IPVC');
14        $I->click('IPVC');
15        $I->amOnPage('http://internacional.ipvc.pt/en/node/17');
16        $I->see('Estudante');
17    }
18 }
```

Listing A.8: Script gerado para o id 1483895776104055303

Scripts de Teste

Para a sessão com o id **1483902088527582623** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Contactos');
8         $I->click('Contactos');
9         $I->amOnPage('http://internacional.ipvc.pt/pt/node/135');
10        $I->see('Equipa');
11        $I->click('Equipa');
12        $I->amOnPage('http://internacional.ipvc.pt/pt/node/87');
13        $I->see('Quem somos');
14        $I->click('Quem somos');
15        $I->amOnPage('http://internacional.ipvc.pt/pt/node/85');
16        $I->see('Equipa');
17        $I->click('Equipa');
18        $I->amOnPage('http://internacional.ipvc.pt/pt/node/87');
19        $I->see('Quem somos');
20        $I->click('Quem somos');
21        $I->amOnPage('http://internacional.ipvc.pt/pt/node/85');
22        $I->see(' Servicos');
23    }
24 }
```

Listing A.9: Script gerado para o id 1483902088527582623

Para a sessão com o id **1483905261373245907** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Erasmus ');
8         $I->click('Erasmus ');
9         $I->amOnPage('http://internacional.ipvc.pt/pt/node/25');
10        $I->see('ERASMUS ICM International Credit Mobility');
11        $I->click('ERASMUS ICM International Credit Mobility');
12        $I->amOnPage('http://internacional.ipvc.pt/pt/node/886');
13        $I->see('Erasmus Mundus');
14        $I->click('Erasmus Mundus');
15        $I->amOnPage('http://internacional.ipvc.pt/pt/node/73');
16        $I->see('Erasmus Mundus');
17        $I->click('Erasmus Mundus');
```

Scripts de Teste

```
18 $I->amOnPage('http://internacional.ipvc.pt/pt/node/73');
19 $I->see('Candidaturas');
20 $I->click('Candidaturas');
21 $I->amOnPage('http://internacional.ipvc.pt/pt/node/39');
22 $I->see('Estudos - SMS');
23 $I->click('Estudos - SMS');
24 $I->amOnPage('http://internacional.ipvc.pt/pt/node/26');
25 $I->see('Erasmus ');
26 $I->click('Erasmus ');
27 $I->amOnPage('http://internacional.ipvc.pt/pt/node/25');
28 $I->see('Programas Internacionais');
29 $I->click('Programas Internacionais');
30 $I->amOnPage('http://internacional.ipvc.pt/pt/node/15');
31 $I->see('Programas');
32 $I->click('Programas');
33 $I->amOnPage('http://internacional.ipvc.pt/pt/node/14');
34 $I->see('ERASMUS ICM International Credit Mobility');
35 $I->click('ERASMUS ICM International Credit Mobility');
36 $I->amOnPage('http://internacional.ipvc.pt/pt/node/886');
37 $I->see(' Docentes');
38 }
39 }
```

Listing A.10: Script gerado para o id 1483905261373245907

Script Manual Desenvolvido pelo Testador:

```
1 class FirstCest
2 {
3     public function frontpageWorks(AcceptanceTester $I)
4     {
5         $I->amOnPage('/');
6         $I->see('Internacional');
7     }
8     public function UniversityPageWorks(AcceptanceTester $I)
9     {
10         $I->amOnPage('/');
11         $I->click('Unviersidades Parceiras');
12         $I->see('A.T.E.I. of Thessaloniki - Greece');
13         $I->click('A.T.E.I. of Thessaloniki - Greece');
14         $I->amOnPage('https://www.teithe.gr/');
15     }
16     public function ProgramsPageWorks(AcceptanceTester $I)
17     {
18         $I->amOnPage('/');
19         $I->click('Programas Internacionais');
20         $I->see('Politicas de Estrategias Europeias');
21     }
22     public function SearchWorks(AcceptanceTester $I)
```

Scripts de Teste

```
23 {
24     $I->amOnPage('/');
25     $I->fillField('texto','franca');
26     $I->click('input[type="image"]');
27     $I->see('Bordeaux Institute of Technology - FR');
28 }
29 public function ErasmusPageWorks(AcceptanceTester $I)
30 {
31     $I->amOnPage('/');
32     $I->click("Erasmus+");
33     $I->see("Descricao do Programa")
34 }
35 public function ErasmusResults(AcceptanceTester $I)
36 {
37     $I->amOnPage('/');
38     $I->click("Seriacao Erasmus");
39     $I->click("Lista de Sериacao para mobilidade Docente para Lecionacao
        2017-2018")
40 }
41 }
42 public function Regulamentation(AcceptanceTester $I)
43 {
44     $I->amOnPage('/');
45     $I->click('SALA DE IMPRENSA ');
46     $I->see('Regulamento de Mobilidade do IPVC');
47 }
48 }
49 public function ProceduresWorks(AcceptanceTester $I)
50 {
51     $I->amOnPage('/');
52     $I->click('Procedimentos Qualidade');
53     $I->see('Mobilidade Alunos');
54     $I->click('Ler Procedimentos')
55 }
56 }
```

Listing A.11: Script desenvolvido pelo Testador

A.2 Caso de Estudo II - Portal de Multimédia do IPVC

Para a sessão com o id **1462291192434536672** foi gerado o seguinte *script*:

```

1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('4th INTERNATIONAL WEEK IPVC');
8         $I->click('4th INTERNATIONAL WEEK IPVC');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/node/1186');
10        $I->see('Arquivo');
11        $I->click('Arquivo');
12        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto-arquivo');
13        $I->see('Em Direto');
14        $I->click('Em Direto');
15        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
16        $I->see('Galeria');
17        $I->click('Galeria');
18        $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
19        $I->see('Galeria');
20        $I->click('Galeria');
21        $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
22        $I->see('Galeria');
23        $I->click('Galeria');
24        $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
25        $I->see('Sala de Imprensa');
26        $I->click('Sala de Imprensa');
27        $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
28        $I->see('Em Direto');
29        $I->click('Em Direto');
30        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
31        $I->see('Galeria');
32        $I->click('Galeria');
33        $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
34        $I->see('Galeria');
35    }
36 }
```

Listing A.12: Script gerado para o id 1462291192434536672

Para a sessão com o id **1473691747078064903** foi gerado o seguinte *script*:

```

1 class ScriptCest
2 {
3     // tests
```

Scripts de Teste

```
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('ForuM');
8         $I->click('ForuM');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/forum');
10        $I->see('Galeria');
11        $I->click('Galeria');
12        $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
13        $I->see('Inicio');
14        $I->click('Inicio');
15        $I->amOnPage('http://multimedia.ipvc.pt');
16        $I->see('Inicio');
17        $I->click('Inicio');
18        $I->amOnPage('http://multimedia.ipvc.pt');
19        $I->see('Inicio');
20        $I->click('Inicio');
21        $I->amOnPage('http://multimedia.ipvc.pt');
22        $I->see('Inicio');
23        $I->click('Inicio');
24        $I->amOnPage('http://multimedia.ipvc.pt');
25        $I->see('Sala de Imprensa');
26        $I->click('Sala de Imprensa');
27        $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
28        $I->see('Arte');
29    }
30 }
```

Listing A.13: Script gerado para o id 1473691747078064903

Para a sessão com o id **1476957482050062904** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('2');
8         $I->click('2');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/notas-imprensa?field_categorias_tid=
            All&keys=&date_filter[min]=&date_filter[max]=&page=1');
10        $I->see('1');
11        $I->click('1');
12        $I->amOnPage('http://multimedia.ipvc.pt/pt/notas-imprensa?field_categorias_tid=
            All&keys=&date_filter[min]=&date_filter[max]='');
13        $I->see('2');
14        $I->click('2');
```

Scripts de Teste

```
15 $I->amOnPage('http://multimedia.ipvc.pt/pt/notas-imprensa?field_categorias_tid=
    All&keys=&date_filter[min]&date_filter[max]&page=1');
16 $I->see('Notas de Imprensa');
17 $I->click('Notas de Imprensa');
18 $I->amOnPage('http://multimedia.ipvc.pt/pt/notas-imprensa');
19 $I->see('Sala de Imprensa');
20 $I->click('Sala de Imprensa');
21 $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
22 $I->see('Entrar');
23 $I->click('Entrar');
24 $I->amOnPage('');
25 $I->fillField('INPUT','Galeria');
26 $I->see(' Login ');
27 $I->click(' Login ');
28 $I->amOnPage('http://multimedia.ipvc.pt/pt/user');
29 $I->see('Sala de Imprensa');
30 $I->click('Sala de Imprensa');
31 $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
32 $I->see(' Inicio');
33 }
34 }
```

Listing A.14: Script gerado para o id 1476957482050062904

Para a sessão com o id **1480070106784850124** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Arquivo');
8         $I->click('Arquivo');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/direto-arquivo');
10        $I->see('Em Direto');
11        $I->click('Em Direto');
12        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
13        $I->see('Em Direto');
14        $I->click('Em Direto');
15        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
16        $I->see(' Arquivo');
17    }
18 }
```

Listing A.15: Script gerado para o id 1480070106784850124

Para a sessão com o id **1483485945338752649** foi gerado o seguinte *script*:

Scripts de Teste

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Galeria');
8         $I->click('Galeria');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
10        $I->see('https://www.youtube.com/watch?v=zD8SxjjVddI');
11        $I->click('https://www.youtube.com/watch?v=zD8SxjjVddI');
12        $I->amOnPage('https://www.youtube.com/watch?v=zD8SxjjVddI');
13        $I->see('Arquivo');
14        $I->click('Arquivo');
15        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto-arquivo');
16        $I->see('Proximas emissoes');
17        $I->click('Proximas emissoes');
18        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
19        $I->see('Proximas emissoes');
20        $I->click('Proximas emissoes');
21        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
22        $I->see('Em Direto');
23        $I->click('Em Direto');
24        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
25        $I->see('Video');
26    }
27 }
```

Listing A.16: Script gerado para o id 1483485945338752649

Para a sessão com o id **1484250825266519193** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Read more about Instalacoes ESCE-IPVC');
8         $I->click('Read more about Instalacoes ESCE-IPVC');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/node/83');
10        $I->see('Instalacoes ESCE-IPVC');
11        $I->click('Instalacoes ESCE-IPVC');
12        $I->amOnPage('http://multimedia.ipvc.pt/pt/node/83');
13        $I->see('Instalacoes ESCE-IPVC');
14        $I->click('Instalacoes ESCE-IPVC');
15        $I->amOnPage('http://multimedia.ipvc.pt/pt/node/83');
16        $I->see('Instalacoes ESCE-IPVC');
```

Scripts de Teste

```
17 $I->click('Instalacoes ESCE-IPVC');
18 $I->amOnPage('http://multimedia.ipvc.pt/pt/node/83');
19 $I->see('Instalacoes ESCE-IPVC');
20 $I->click('Instalacoes ESCE-IPVC');
21 $I->amOnPage('http://multimedia.ipvc.pt/pt/node/83');
22 $I->see('Sala de Imprensa');
23 $I->click('Sala de Imprensa');
24 $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
25 $I->see('Em Direto');
26 $I->click('Em Direto');
27 $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
28 $I->see('Sala de Imprensa');
29 $I->click('Sala de Imprensa');
30 $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
31 $I->see(' Imprensa');
32 }
33 }
```

Listing A.17: Script gerado para o id 1484250825266519193

Para a sessão com o id **1485180232158538855** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('2');
8         $I->click('2');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias?&&date_filter[min]&date_filter[
10             max]&keys=&page=1');
11         $I->see('Galeria');
12         $I->click('Galeria');
13         $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
14         $I->see('ForuM');
15         $I->click('ForuM');
16         $I->amOnPage('http://multimedia.ipvc.pt/pt/forum');
17         $I->see('Sala de Imprensa');
18         $I->click('Sala de Imprensa');
19         $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
20         $I->see('Sala de Imprensa');
21         $I->click('Sala de Imprensa');
22         $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
23         $I->see('OBSERVATORIO DA CRIANCA E JOVEM DE MELGACO '16');
24         $I->click('OBSERVATORIO DA CRIANCA E JOVEM DE MELGACO '16');
25         $I->amOnPage('http://multimedia.ipvc.pt/pt/node/1275');
26         $I->see(' Observatorio');
27     }
28 }
```

27 }

Listing A.18: Script gerado para o id 1485180232158538855

Para a sessão com o id **1485201094774105969** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Instalacoes ESA-IPVC');
8         $I->click('Instalacoes ESA-IPVC');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/node/76');
10        $I->see('Instalacoes ESA-IPVC');
11        $I->click('Instalacoes ESA-IPVC');
12        $I->amOnPage('http://multimedia.ipvc.pt/pt/node/76');
13        $I->see('Instalacoes ESA-IPVC');
14        $I->click('Instalacoes ESA-IPVC');
15        $I->amOnPage('http://multimedia.ipvc.pt/pt/node/76');
16        $I->see('Instalacoes ESA-IPVC');
17        $I->click('Instalacoes ESA-IPVC');
18        $I->amOnPage('http://multimedia.ipvc.pt/pt/node/76');
19        $I->see('Sala de Imprensa');
20        $I->click('Sala de Imprensa');
21        $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
22        $I->see('Instalacoes');
23    }
24 }
```

Listing A.19: Script gerado para o id 1485201094774105969

Para a sessão com o id **1462199305394109826** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Galeria');
8         $I->click('Galeria');
9         $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
10        $I->see('Galeria');
11        $I->click('Galeria');
12        $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
13        $I->see('Galeria');
```

Scripts de Teste

```
14 $I->click('Galeria');
15 $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
16 $I->see('Galeria');
17 $I->click('Galeria');
18 $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
19 $I->see('Galeria');
20 $I->click('Galeria');
21 $I->amOnPage('http://multimedia.ipvc.pt/pt/galerias');
22 $I->see('Inicio');
23 }
24 }
```

Listing A.20: Script gerado para o id 1462199305394109826

Para a sessão com o id **1482589474144548088** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Inicio');
8         $I->click('Inicio');
9         $I->amOnPage('http://multimedia.ipvc.pt');
10        $I->see('Sala de Imprensa');
11        $I->click('Sala de Imprensa');
12        $I->amOnPage('http://multimedia.ipvc.pt/pt/sala-imprensa');
13        $I->see('Em Direto');
14        $I->click('Em Direto');
15        $I->amOnPage('http://multimedia.ipvc.pt/pt/direto');
16        $I->see(' Sala');
17    }
18 }
```

Listing A.21: Script gerado para o id 1482589474144548088

Script Manual Desenvolvido pelo Testador:

```
1 class FirstCest
2 {
3     public function frontpageWorks(AcceptanceTester $I)
4     {
5         $I->amOnPage('/');
6         $I->see('IPVC-Multimedia');
7     }
8     public function GalleryPageWorks(AcceptanceTester $I)
9     {
```

Scripts de Teste

```
10     $I->amOnPage('/');
11     $I->click('GALERIA');
12     $I->see('36 Aniversario ESE-IPVC');
13 }
14 public function ForumPageWorks(AcceptanceTester $I)
15 {
16     $I->amOnPage('/');
17     $I->click('FORUM');
18     $I->see('Forum IPVC');
19 }
20 public function SearchWorks(AcceptanceTester $I)
21 {
22     $I->amOnPage('/');
23     $I->fillField('texto','ipvc');
24     $I->click('input[type="image"]');
25     $I->see('IPVC RECEBE ESTUDANTES LUSO-CHINESES e CABO-VERDIANOS');
26 }
27 public function forumSearchWorks(AcceptanceTester $I)
28 {
29     $I->amOnPage('/');
30     $I->click("Forum");
31     $I->fillField('texto','ipvc');
32     $I->click('input[type="image"]');
33     $I->click('FORUM IPVC PROGRAMA 371');
34 }
35 public function pageSwitchWorks(AcceptanceTester $I)
36 {
37     $I->amOnPage('/');
38     $I->click("Forum");
39     $I->fillField('texto','ipvc');
40     $I->click('2');
41     $I->click('proxima');
42     $I->see('FORUM IPVC PROGRAMA 355');
43 }
44 }
45 public function ImprensaWorks(AcceptanceTester $I)
46 {
47     $I->amOnPage('/');
48     $I->click('SALA DE IMPRENSA ');
49     $I->see('NOTAS DE IMPRENSA');
50 }
51 }
52 public function MinhoAcademicoWorks(AcceptanceTester $I)
53 {
54     $I->amOnPage('/');
55     $I->click('MINHO ACADEMICO');
56     $I->see('O MINHO ACADEMICO 111');
57 }
58 }
```

59 }

Listing A.22: Script Manual desenvolvido pelo Testador

A.3 Caso de Estudo III - *Newspaper* da Cidade de Tomar

Para a sessão com o id **1416577906572708104** foi gerado o seguinte *script*:

```

1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->amOnPage('/');
8         $I->see('Mais de meia centena de ovelhas aparecem mortas em Paialvo');
9         $I->click('Mais de meia centena de ovelhas aparecem mortas em Paialvo');
10        $I->amOnPage('http://www.cidadetomar.pt/noticia/5927/mais-de-meia-centena-de-
        ovelhas-aparecem-mortas-em-paialvo');
11        $I->see('Alguns elementos de grupos etnograficos');
12        $I->click('Alguns elementos de grupos etnograficos');
13        $I->amOnPage('http://www.cidadetomar.pt/noticia/5930/iii-mostra-gastronomica-do-
        grao-de-bico-juntou-mais-de-350-pessoas-em-convivio');
14        $I->see('FREGUESIAS');
15        $I->click('FREGUESIAS');
16        $I->amOnPage('http://www.cidadetomar.pt/freguesias');
17        $I->see('Funeral de Carina Rocha realiza-se nesta terca-feira');
18        $I->click('Funeral de Carina Rocha realiza-se nesta terca-feira ');
19        $I->amOnPage('http://www.cidadetomar.pt/noticia/5916/funeral-de-carina-rocha-
        realiza-se-nesta-terca-feira');
20        $I->see('O aparato de meios na Rua de Coimbra pelas nove da manha');
21        $I->click('O aparato de meios na Rua de Coimbra pelas nove da manha');
22        $I->amOnPage('http://www.cidadetomar.pt/noticia/5849/mega-operacao-policial-em-
        tomar');
23        $I->see('Noticias');
24    }
25 }
```

Listing A.23: Script gerado para o id 1416577906572708104

Para a sessão com o id **1416579638764395142** foi gerado o seguinte *script*:

```

1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
```

Scripts de Teste

```
5 {
6     $I->amOnPage('/');
7 $I->see('Mais desacetos durante esta noite no Bairro 1.de Maio');
8 $I->click('Mais desacetos durante esta noite no Bairro 1.de Maio');
9 $I->amOnPage('http://www.cidadetomar.pt/noticia/5908/mais-desacetos-durante-esta-
    noite-no-bairro-1-de-maio');
10 $I->see('Acidente ocorreu na Av. D. Angela Tamagnini perto da rotunda do Bonjardim'
    );
11 $I->click('Acidente ocorreu na Av. D. Angela Tamagnini perto da rotunda do
    Bonjardim');
12 $I->amOnPage('http://www.cidadetomar.pt/noticia/5881/mae-e-dois-filhos-atropelados-
    na-passadeira');
13 $I->see('ULTIMAS');
14 $I->click('ULTIMAS');
15 $I->amOnPage('http://www.cidadetomar.pt/ultimas');
16 $I->see('Familia atropelada com gravidade quando atravessava na passadeira');
17 $I->click('Familia atropelada com gravidade quando atravessava na passadeira');
18 $I->amOnPage('http://www.cidadetomar.pt/noticia/7424/familia-atropelada-com-
    gravidade-quando-atravessava-na-passadeira/e');
19 $I->see('Familia atropelada foi evacuada para Abrantes');
20 $I->click('Familia atropelada foi evacuada para Abrantes');
21 $I->amOnPage('http://www.cidadetomar.pt/noticia/5883/familia-atropelada-foi-
    evacuada-para-abrantes');
22 $I->see('Uma das muitas bancas participantes neste certame');
23 $I->click('Uma das muitas bancas participantes neste certame');
24 $I->amOnPage('http://www.cidadetomar.pt/noticia/5843/xiv-feira-dos-santos-de-
    olalhas-em-tomar');
25 $I->see('ULTIMAS');
26 $I->click('ULTIMAS');
27 $I->amOnPage('http://www.cidadetomar.pt/ultimas');
28 $I->see('FREGUESIAS');
29 $I->click('FREGUESIAS');
30 $I->amOnPage('http://www.cidadetomar.pt/freguesias');
31 $I->see('FREGUESIAS');
32 $I->click('FREGUESIAS');
33 $I->amOnPage('http://www.cidadetomar.pt/freguesias');
34 $I->see('EDICAO IMPRESSA');
35 $I->click('EDICAO IMPRESSA');
36 $I->amOnPage('http://www.cidadetomar.pt/edicaoimpressa');
37 $I->see('Santa Casa da Misericordia de Tomar compra dois edificios');
38 $I->click('Santa Casa da Misericordia de Tomar compra dois edificios');
39 $I->amOnPage('http://www.cidadetomar.pt/noticia/5853/santa-casa-da-misericordia-de-
    tomar-compra-dois-edificios');
40 $I->see('Ha mais de quatro mil colmeias no concelho em Tomar');
41 $I->click('Ha mais de quatro mil colmeias no concelho em Tomar');
42 $I->amOnPage('http://www.cidadetomar.pt/noticia/5854/ha-mais-de-quatro-mil-colmeias
    -no-concelho-em-tomar');
43 $I->see('Mega-operacao policial em Tomar');
44 $I->click('Mega-operacao policial em Tomar');
```

Scripts de Teste

```
45 $I->amOnPage('http://www.cidadetomar.pt/noticia/5849/mega-operacao-policial-em-
    tomar');
46 $I->see('Mega-operacao policial em Tomar');
47 $I->click('Mega-operacao policial em Tomar');
48 $I->amOnPage('http://www.cidadetomar.pt/noticia/5849/mega-operacao-policial-em-
    tomar');
49 $I->see('Rusga policial termina com sete detencoes');
50 $I->click('Rusga policial termina com sete detencoes');
51 $I->amOnPage('http://www.cidadetomar.pt/noticia/5850/rusga-policial-termina-com-
    sete-detencoes');
52 $I->see('Grupo que posou para a objectiva do nosso jornal antes da partida');
53 $I->click('Grupo que posou para a objectiva do nosso jornal antes da partida');
54 $I->amOnPage('http://www.cidadetomar.pt/noticia/5836/benfiquistas-de-tomar-em-
    excursao-rumo-ao-estadio-da-luz');
55 $I->see('ULTIMAS');
56 $I->click('ULTIMAS');
57 $I->amOnPage('http://www.cidadetomar.pt/ultimas');
58 $I->see('Madrugada de sabado marcada por desacatos na Praca da Republica');
59 $I->click('Madrugada de sabado marcada por desacatos na Praca da Republica');
60 $I->amOnPage('http://www.cidadetomar.pt/noticia/5840/madrugada-de-sabado-marcada-
    por-desacatos-na-praca-da-republica');
61 $I->see(' Freguesias');
62 }
63 }
```

Listing A.24: Script gerado para o id 1416579638764395142

Para a sessão com o id **1416582532131830449** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Uma das imagens do evento');
8         $I->click('Uma das imagens do evento');
9         $I->amOnPage('http://www.cidadetomar.pt/noticia/5939/centro-recreativo-e-cultural-
            do-coito-assinalou-39-anos-de-existencia');
10        $I->see('II Workshop sobre comida vegetariana em Cem Soldos');
11        $I->click('II Workshop sobre comida vegetariana em Cem Soldos');
12        $I->amOnPage('http://www.cidadetomar.pt/noticia/5918/ii-workshop-sobre-comida-
            vegetariana-em-cem-soldos');
13        $I->see('Funeral de Carina Rocha realiza-se nesta terca-feira');
14        $I->click('Funeral de Carina Rocha realiza-se nesta terca-feira');
15        $I->amOnPage('http://www.cidadetomar.pt/noticia/5916/funeral-de-carina-rocha-
            realiza-se-nesta-terca-feira');
16        $I->see('Morte de jovem provoca onda de consternacao na comunidade tomarense');
17        $I->click('Morte de jovem provoca onda de consternacao na comunidade tomarense');
```

Scripts de Teste

```
18 $I->amOnPage('http://www.cidadetomar.pt/noticia/5903/morte-de-jovem-provoca-onda-de-
    -consternacao-na-comunidade-tomarense');
19 $I->see('Movimento de Esclerose Múltipla do Médio Tejo assinalou 1 aniversário');
20 $I->click('Movimento de Esclerose Múltipla do Médio Tejo assinalou 1 aniversário');
21 $I->amOnPage('http://www.cidadetomar.pt/noticia/7427/movimento-de-esclerose-
    multipla-do-medio-tejo-assinalou-1-aniversario/e');
22 $I->see('Mais um atropelamento numa passadeira em Tomar');
23 $I->click('Mais um atropelamento numa passadeira em Tomar');
24 $I->amOnPage('http://www.cidadetomar.pt/noticia/5900/mais-um-atropelamento-numa-
    passadeira-em-tomar');
25 $I->see('Um dos sete detidos em Tomar ficou preso preventivamente');
26 $I->click('Um dos sete detidos em Tomar ficou preso preventivamente');
27 $I->amOnPage('http://www.cidadetomar.pt/noticia/5874/um-dos-sete-detidos-em-tomar-
    ficou-presos-preventivamente');
28 $I->see('Sara Marques Costa lidera jovens socialistas de Tomar');
29 $I->click('Sara Marques Costa lidera jovens socialistas de Tomar');
30 $I->amOnPage('http://www.cidadetomar.pt/noticia/5870/sara-marques-costa-lidera-
    jovens-socialistas-de-tomar');
31 $I->see('Tomar esta a receber curso de iniciacao a apicultura neste sabado ');
32 $I->click('Tomar esta a receber curso de iniciacao a apicultura neste sabado ');
33 $I->amOnPage('http://www.cidadetomar.pt/noticia/5835/tomar-esta-a-receber-curso-de-
    iniciacao-a-apicultura-neste-sabado');
34 $I->see('Menino que caiu numa fossa apresenta mais melhorias');
35 $I->click('Menino que caiu numa fossa apresenta mais melhorias');
36 $I->amOnPage('http://www.cidadetomar.pt/noticia/5825/menino-que-caiu-numa-fossa-
    apresenta-mais-melhorias');
37 $I->see('PSP apenas identificou um individuo nos desacatos de sabado');
38 $I->click('PSP apenas identificou um individuo nos desacatos de sabado');
39 $I->amOnPage('http://www.cidadetomar.pt/noticia/5842/psp-apenas-identificou-um-
    individuo-nos-desacatos-de-sabado');
40 $I->see('Madrugada de sabado marcada por desacatos na Praca da Republica ');
41 $I->click('Madrugada de sabado marcada por desacatos na Praca da Republica ');
42 $I->amOnPage('http://www.cidadetomar.pt/noticia/5840/madrugada-de-sabado-marcada-
    por-desacatos-na-praca-da-republica');
43 $I->see(' Imagens');
44 }
45 }
```

Listing A.25: Script gerado para o id 1416582532131830449

Para a sessão com o id **1416583193437363022** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('');
7         $I->see('Carina Rocha era aluna do Politecnico de Tomar');
```

Scripts de Teste

```
8 $I->click('Carina Rocha era aluna do Politecnico de Tomar');
9 $I->amOnPage('http://www.cidadetomar.pt/noticia/5903/morte-de-jovem-provoca-onda-de-
  -consternacao-na-comunidade-tomarense');
10 $I->see('Morte de jovem provoca onda de consternacao na comunidade tomarense');
11 $I->click('Morte de jovem provoca onda de consternacao na comunidade tomarense');
12 $I->amOnPage('http://www.cidadetomar.pt/noticia/5903/morte-de-jovem-provoca-onda-de-
  -consternacao-na-comunidade-tomarense');
13 $I->see('Mais um atropelamento numa passadeira em Tomar');
14 $I->click('Mais um atropelamento numa passadeira em Tomar');
15 $I->amOnPage('http://www.cidadetomar.pt/noticia/5900/mais-um-atropelamento-numa-
  passadeira-em-tomar');
16 $I->see('Mais um atropelamento numa passadeira em Tomar');
17 $I->click('Mais um atropelamento numa passadeira em Tomar');
18 $I->amOnPage('http://www.cidadetomar.pt/noticia/5900/mais-um-atropelamento-numa-
  passadeira-em-tomar');
19 $I->see('Familia atropelada foi evacuada para Abrantes ');
20 $I->click('Familia atropelada foi evacuada para Abrantes ');
21 $I->amOnPage('http://www.cidadetomar.pt/noticia/5883/familia-atropelada-foi-
  evacuada-para-abrantes');
22 $I->see('Mae e dois filhos atropelados na passadeira ');
23 $I->click('Mae e dois filhos atropelados na passadeira ');
24 $I->amOnPage('http://www.cidadetomar.pt/noticia/5881/mae-e-dois-filhos-atropelados-
  na-passadeira');
25 $I->see(' Familia');
26 }
27 }
```

Listing A.26: Script gerado para o id 1416583193437363022

Para a sessão com o id **1416588780569525930** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('Funeral de Carina Rocha realiza-se nesta terca-feira');
8         $I->click('Funeral de Carina Rocha realiza-se nesta terca-feira');
9         $I->amOnPage('http://www.cidadetomar.pt/noticia/5916/funeral-de-carina-rocha-
  realiza-se-nesta-terca-feira');
10        $I->see('Um dos sete detidos em Tomar ficou preso preventivamente ');
11        $I->click('Um dos sete detidos em Tomar ficou preso preventivamente ');
12        $I->amOnPage('http://www.cidadetomar.pt/noticia/5874/um-dos-sete-detidos-em-tomar-
  ficou-preso-preventivamente');
13        $I->see('Leoes ficaram novamente isolados na frente');
14        $I->click('Leoes ficaram novamente isolados na frente');
15        $I->amOnPage('http://www.cidadetomar.pt/noticia/7428/leoes-ficaram-novamente-
  isolados-na-frente/e');
```

Scripts de Teste

```
16 $I->see('Rusga policial termina com sete detencoes ');
17 $I->click('Rusga policial termina com sete detencoes ');
18 $I->amOnPage('http://www.cidadetomar.pt/noticia/5850/rusga-policial-termina-com-
    sete-detencoes');
19 $I->see('PSP apenas identificou um individuo nos desacatos de sabado');
20 $I->click('PSP apenas identificou um individuo nos desacatos de sabado');
21 $I->amOnPage('http://www.cidadetomar.pt/noticia/5842/psp-apenas-identificou-um-
    individuo-nos-desacatos-de-sabado');
22 $I->see('Madrugada de sabado marcada por desacatos na Praca da Republica ');
23 $I->click('Madrugada de sabado marcada por desacatos na Praca da Republica ');
24 $I->amOnPage('http://www.cidadetomar.pt/noticia/5840/madrugada-de-sabado-marcada-
    por-desacatos-na-praca-da-republica');
25 $I->see('PSP ');
26 }
27 }
```

Listing A.27: Script gerado para o id 1416588780569525930

Para a sessão com o id **1416591367482257479** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('ULTIMAS');
8         $I->click('ULTIMAS');
9         $I->amOnPage('http://www.cidadetomar.pt/ultimas');
10        $I->amOnPage('http://www.cidadetomar.pt/noticia/5881/mae-e-dois-filhos-atropelados-
            na-passadeira');
11        $I->see('Alunos do Politecnico doam brinquedos a Misericordia de Tomar ');
12        $I->click('
13        Alunos do Politecnico doam brinquedos a Misericordia de Tomar ');
14        $I->amOnPage('http://www.cidadetomar.pt/noticia/5872/alunos-do-politecnico-doam-
            brinquedos-a-misericordia-de-tomar');
15        $I->see('O aparato de meios na Rua de Coimbra pelas nove da manha');
16        $I->click('O aparato de meios na Rua de Coimbra pelas nove da manha');
17        $I->amOnPage('http://www.cidadetomar.pt/noticia/5849/mega-operacao-policial-em-
            tomar');
18        $I->see('Ultimas');
19    }
20 }
```

Listing A.28: Script gerado para o id 1416591367482257479

Para a sessão com o id **1418985609993016717** foi gerado o seguinte *script*:

Scripts de Teste

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('FREGUESIAS');
8         $I->click('FREGUESIAS');
9         $I->amOnPage('http://www.cidadetomar.pt/freguesias');
10        $I->see('FREGUESIAS');
11        $I->click('FREGUESIAS');
12        $I->amOnPage('http://www.cidadetomar.pt/freguesias');
13        $I->see('FREGUESIAS');
14        $I->click('FREGUESIAS');
15        $I->amOnPage('http://www.cidadetomar.pt/freguesias');
16        $I->see('Município de Tomar vai adquirir veículo eléctrico para unidade móvel de
            saúde');
17        $I->click('Município de Tomar vai adquirir veículo eléctrico para unidade móvel de
            saúde ');
18        $I->amOnPage('http://www.cidadetomar.pt/noticia/7367/municipio-de-tomar-vai-
            adquirir-veiculo-electrico-para-unidade-movel-de-saude');
19        $I->see('Município de Tomar vai adquirir veículo eléctrico para unidade móvel de
            saúde');
20        $I->click('Município de Tomar vai adquirir veículo eléctrico para unidade móvel de
            saúde');
21        $I->amOnPage('http://www.cidadetomar.pt/noticia/7367/municipio-de-tomar-vai-
            adquirir-veiculo-electrico-para-unidade-movel-de-saude');
22        $I->see('ULTIMAS');
23        $I->click('ULTIMAS');
24        $I->amOnPage('http://www.cidadetomar.pt/ultimas');
25        $I->see('Aprovadas candidaturas para concluir abastecimento de água e ampliar rede
            de esgotos');
26        $I->click('Aprovadas candidaturas para concluir abastecimento de água e ampliar
            rede de esgotos');
27        $I->amOnPage('http://www.cidadetomar.pt/noticia/7328/aprovadas-candidaturas-para-
            concluir-abastecimento-de-agua-e-ampliar-rede-de-esgotos');
28        $I->see('Seis distritos de Portugal continental estão hoje sob Aviso Amarelo');
29        $I->click('Seis distritos de Portugal continental estão hoje sob Aviso Amarelo');
30        $I->amOnPage('http://www.cidadetomar.pt/noticia/7311/seis-distritos-de-portugal-
            continental-estao-hoje-sob-aviso-amarelo');
31        $I->see('Incêndio em lareira no Alvito');
32        $I->click('Incêndio em lareira no Alvito');
33        $I->amOnPage('http://www.cidadetomar.pt/noticia/7310/incendio-em-lareira-no-alvito'
            );
34        $I->see('Incêndio em lareira no Alvito');
35        $I->click('Incêndio em lareira no Alvito');
36        $I->amOnPage('http://www.cidadetomar.pt/noticia/7310/incendio-em-lareira-no-alvito'
            );
37        $I->see('Incêndio em lareira no Alvito');
```

Scripts de Teste

```
38 $I->click('Incendio em lareira no Alvito');
39 $I->amOnPage('http://www.cidadetomar.pt/noticia/7310/incendio-em-lareira-no-alvito'
);
40 $I->see('Incendio em lareira no Alvito');
41 $I->click('Incendio em lareira no Alvito');
42 $I->amOnPage('http://www.cidadetomar.pt/noticia/7310/incendio-em-lareira-no-alvito'
);
43 $I->see('FREGUESIAS');
44 $I->click('FREGUESIAS');
45 $I->amOnPage('http://www.cidadetomar.pt/freguesias');
46 $I->see('Município');
47 }
48 }
```

Listing A.29: Script gerado para o id 1418985609993016717

Para a sessão com o id **1419272542371602330** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('');
7         $I->amOnPage('http://www.cidadetomar.pt/noticia/5850/rusga-policial-termina-com-
se-
sete-detencoes');
8         $I->see('Francisco Faria preside a ACITOFEBa ha varios anos');
9         $I->click('Francisco Faria preside a ACITOFEBa ha varios anos');
10        $I->amOnPage('http://www.cidadetomar.pt/noticia/5851/vem-ai-a-mostra-do-bacalhau-da
-acitofeba');
11        $I->see('Santa Casa da Misericordia de Tomar compra dois edificios');
12        $I->click('Santa Casa da Misericordia de Tomar compra dois edificios');
13        $I->amOnPage('http://www.cidadetomar.pt/noticia/5853/santa-casa-da-misericordia-de-
tomar-compra-dois-edificios');
14        $I->see('Município corta duas arvores na zona do Mercado ');
15        $I->click('Município corta duas arvores na zona do Mercado ');
16        $I->amOnPage('http://www.cidadetomar.pt/noticia/5852/municipio-corta-duas-arvores-
na-zona-do-mercado');
17        $I->see('12a Edicao da Biodiversidade a 13 de novembro em Tomar');
18        $I->click('12a Edicao da Biodiversidade a 13 de novembro em Tomar ');
19        $I->amOnPage('http://www.cidadetomar.pt/noticia/5844/12-edicao-da-biodiversidade-a
-13-de-novembro-em-tomar');
20        $I->see('Mega-operacao policial em Tomar');
21        $I->click('Mega-operacao policial em Tomar');
22        $I->amOnPage('http://www.cidadetomar.pt/noticia/5849/mega-operacao-policial-em-
tomar');
23        $I->see('Rusga policial termina com sete detencoes');
24        $I->click('Rusga policial termina com sete detencoes');
```

Scripts de Teste

```
25 $I->amOnPage('http://www.cidadetomar.pt/noticia/5850/rusga-policial-termina-com-
    sete-detencoes');
26 $I->see('Camara de Tomar nao paga a Parq T desde o inicio do ano');
27 $I->click('Camara de Tomar nao paga a Parq T desde o inicio do ano');
28 $I->amOnPage('http://www.cidadetomar.pt/noticia/5848/camara-de-tomar-nao-paga-a-
    parq-t-desde-o-inicio-do-ano');
29 $I->see('Madrugada de sabado marcada por desacatos na Praca da Republica ');
30 $I->click('Madrugada de sabado marcada por desacatos na Praca da Republica ');
31 $I->amOnPage('http://www.cidadetomar.pt/noticia/5840/madrugada-de-sabado-marcada-
    por-desacatos-na-praca-da-republica');
32 $I->see('Fotografia dos desacatos de sabado (todos os direitos reservados)');
33 $I->click('Fotografia dos desacatos de sabado (todos os direitos reservados)');
34 $I->amOnPage('http://www.cidadetomar.pt/noticia/5842/psp-apanas-identificou-um-
    individuo-nos-desacatos-de-sabado');
35 $I->see('Dom Quixote e Sancho Panca "desfilaram" na Corredoura ');
36 $I->click('Dom Quixote e Sancho Panca "desfilaram" na Corredoura ');
37 $I->amOnPage('http://www.cidadetomar.pt/noticia/5838/dom-quixote-e-sancho-panca-
    desfilaram-na-corredoura');
38 $I->see(' Fotografia');
39 }
40 }
```

Listing A.30: Script gerado para o id 1419272542371602330

Para a sessão com o id **141927254235486213** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('FREGUESIAS');
8         $I->click('FREGUESIAS');
9         $I->amOnPage('http://www.cidadetomar.pt/freguesias');
10        $I->see('FREGUESIAS');
11        $I->click('FREGUESIAS');
12        $I->amOnPage('http://www.cidadetomar.pt/freguesias');
13        $I->see('FREGUESIAS');
14        $I->click('FREGUESIAS');
15        $I->amOnPage('http://www.cidadetomar.pt/freguesias');
16        $I->see('Municipio de Tomar vai adquirir veiculo electrico para unidade movel de
            saude');
17        $I->click('Municipio de Tomar vai adquirir veiculo electrico para unidade movel de
            saude ');
18        $I->see('Aprovadas candidaturas para concluir abastecimento de agua e ampliar rede
            de esgotos');
19        $I->click('Aprovadas candidaturas para concluir abastecimento de agua e ampliar
            rede de esgotos');
```

Scripts de Teste

```
20 $I->amOnPage('http://www.cidadetomar.pt/noticia/7328/aprovadas-candidaturas-para-
    concluir-abastecimento-de-agua-e-ampliar-rede-de-esgotos');
21 $I->see('Seis distritos de Portugal continental estao hoje sob Aviso Amarelo');
22 $I->click('Seis distritos de Portugal continental estao hoje sob Aviso Amarelo');
23 $I->amOnPage('http://www.cidadetomar.pt/noticia/7311/seis-distritos-de-portugal-
    continental-estao-hoje-sob-aviso-amarelo');
24 $I->see('FREGUESIAS');
25 $I->click('FREGUESIAS');
26 $I->amOnPage('http://www.cidadetomar.pt/freguesias');
27 $I->see('Camara');
28 }
29 }
```

Listing A.31: Script gerado para o id 141927254235486213

Para a sessão com o id **14192214579632145** foi gerado o seguinte *script*:

```
1 class ScriptCest
2 {
3     // tests
4     public function tryToTest(AcceptanceTester $I)
5     {
6         $I->amOnPage('/');
7         $I->see('ULTIMAS');
8         $I->click('ULTIMAS');
9         $I->amOnPage('http://www.cidadetomar.pt/ultimas');
10        $I->amOnPage('http://www.cidadetomar.pt/noticia/5881/mae-e-dois-filhos-atropelados-
            na-passadeira');
11        $I->see('Alunos do Politecnico doam brinquedos a Misericordia de Tomar ');
12        $I->click('
13        Alunos do Politecnico doam brinquedos a Misericordia de Tomar ');
14        $I->amOnPage('http://www.cidadetomar.pt/noticia/5872/alunos-do-politecnico-doam-
            brinquedos-a-misericordia-de-tomar');
15        $I->see('O aparato de meios na Rua de Coimbra pelas nove da manha');
16        $I->fillField('user_username','ola@gmail.com');
17        $I->fillField('user_password','cenas45');
18        $I->click('Login');
19        $I->click('O aparato de meios na Rua de Coimbra pelas nove da manha');
20        $I->amOnPage('http://www.cidadetomar.pt/noticia/5849/mega-operacao-policial-em-
            tomar');
21        $I->see('Ultimas');
22    }
23 }
```

Listing A.32: Script gerado para o id 14192214579632145

Script Manual Desenvolvido pelo Testador:

Scripts de Teste

```
1 class FirstCest
2 {
3     public function frontpageWorks(AcceptanceTester $I)
4     {
5         $I->amOnPage('/');
6         $I->see('Cidade de Tomar');
7     }
8     public function SportPageWorks(AcceptanceTester $I)
9     {
10         $I->amOnPage('/');
11         $I->click('DESPORTO');
12         $I->see('Torneio Interassociativo do Carnaval em Tomar');
13     }
14     public function SearchWorks(AcceptanceTester $I)
15     {
16         $I->amOnPage('/');
17         $I->fillField('texto','tomar');
18         $I->click('input[type="image"]');
19         $I->see('Semana Santa em Tomar');
20     }
21     public function EpaperRegisterTesting(AcceptanceTester $I)
22     {
23         $I->amOnPage('/');
24         $I->click("E-PAPER");
25         $I->click('Minha conta');
26         $I->fillField('email','up201306032@fe.up.pt');
27         $I->fillField('password','dinamite45');
28         $I->click('Registar nova conta');
29     }
30     public function EpaperLoginTesting(AcceptanceTester$I)
31     {
32         $I->amOnPage('/');
33         $I->click("E-PAPER");
34         $I->click('Minha conta');
35         $I->fillField('user_username','up201306032@fe.up.pt');
36         $I->fillField('user_password','dinamite45');
37         $I->click('Entrar');
38         $I->fillField('pwd','dinamite45');
39         $I->click('input[type="submit"]');
40         $I->see('Minha Conta');
41     }
42     public function FacebookTest(AcceptanceTester $I)
43     {
44         $I->amOnPage('/');
45         $I->click("Facebook");
46         $I->see("Facebook");
47     }
48     public function TwitterTest(AcceptanceTester $I)
49     {
```

Scripts de Teste

```
50     $I->amOnPage('/');
51     $I->see("Twitter");
52     $I->click("Twitter");
53     $I->seeElement("#search_submit");
54
55 }
56 public function PharmacyTest(AcceptanceTester $I)
57 {
58     $I->amOnPage('/');
59     $I->see('FARMACIA DE SERVICIO','.tituloultimas');
60
61 }
62
63 }
```

Listing A.33: Script desenvolvido pelo Testador