IPG Politécnico da Guarda
Escola Superior de Tecnologia e Gestão

RELATÓRIO DE ESTÁGIO

Curso Técnico Superior Profissional
em Testes de Software

Eduardo Augusto Bernardes Almeida

julho | 2018

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

Relatório de Estágio

RELATÓRIO PARA A OBTENÇÃO DO DIPLOMA DE TÉCNICO SUPERIOR DE TESTES DE SOFTWARE

EDUARDO AUGUSTO BERNARDES ALMEIDA

2017/2018

Elementos identificativos:

Aluno:

Eduardo Augusto Bernardes Almeida
Nº1012492
TESP – Testes de Software

Estabelecimento de Ensino:

Escola Superior de Tecnologia e Gestão
Instituto Politécnico da Guarda

Instituição de Estágio:

Nome: Altran - Global Delivery Centre

Morada: Centro de Negócios e Serviços, Praça Amália Rodrigues 6230-350 Fundão

Duração do Estágio:

Inicio: 08 Março 2018

Fim: 13 Julho 2018

Orientador de Estágio:

Marcus José da Silva Pereira

Grau Académico: Licenciatura em Engenharia de Computação

Agradecimentos

Agradeço imenso ao Instituto Politécnico da Guarda, mais propriamente à Escola Superior de Tecnologia e Gestão por todo o apoio e dedicação ao longo do meu percurso. Pela oportunidade que me foi proporcionada que me ajudou a crescer tanto a nível pessoal como a nível profissional.

À Coordenadora do curso e orientadora de estágio Professora Natália Gomes por toda a ajuda que foi dada, empenho, disponibilidade e esforço para que tudo decorre-se da melhor forma.

Especialmente gostaria de agradecer à minha família, amigos e colegas de curso que me apoiaram, pois esse foi fundamental para o concluir deste percurso.

Por fim queria agradecer à equipa do projeto em que fui inserido, ao meu Project Manager Marcus Pereira pelo apoio, conhecimentos, conceitos e experiencia que partilhou e a todos os membros da sala da onde fui alocado, pela integração, pelo ambiente profissional, simpatia e toda ajuda que me foi dada durante o estágio.

Plano de Estágio Curricular

O plano de estágio curricular foi determinado pela Altran tendo com prognóstico a elaboração das seguintes atividades no decorrer do estágio curricular:

- ✓ Integração na equipa de Projeto;
- ✓ Conhecimento e estudo dos objetivos do projeto;
- ✓ Conhecer os objetivos da equipa de testes no projeto;
- ✓ Conhecer as normas e o sistema a ser testado;
- ✓ Estudo dos processos de garantia de qualidade da empresa;
- ✓ Estudo dos processos para um ciclo de vida de desenvolvimento de sistemas relacionados a Software de dispositivos médicos (ISO 13485 e ISSO/IEC 62304);
- ✓ Criação e alteração de testes funcionais (manuais);
- ✓ Execução de testes funcionais (manuais) de acordo com as metodologias da Altran;
- ✓ Report de defeitos (bugs);
- ✓ Suporte na conceção da documentação do projeto como um todo.

Índice

Índice de Figuras

Índice de Tabelas

Glossário de Abreviaturas

BACKLOG – Lista de funcionalidades a serem implementadas

BUG – Erro ou inconsistência

CHANGE REQUEST – Pedido de alteração

COPD - Doença pulmonar inflamatória crônica

DAILY SCRUM – Reunião diária

ISSUE – Tarefa

PRODUCT OWNER – Pessoa responsável pela representação do cliente

SCRUM – Modelo iterativa ou incrementar da metodologia Ágil

SOP´S – Procedimentos de Operação em Sistemas

SPRINTS – Representa um ciclo na metodologia Ágil

SPRINT PLANNING – Reunião de planeamento da Sprint

STAKEHOLDERS – Pessoas interessadas no projeto

TESP – Técnico Superior Profissional

TEST CASE – Condição particular a ser testada

TEST DATA – Conjunto de dados de teste

TEST SUIT - Conjunto de casos de teste

TESTWARE – Conjunto de ficheiros relacionados com a atividade de teste

USER STORY – Histórias do Utilizador

Resumo

Neste relatório irá ser apresentado todo o trabalho realizado ao longo do estágio curricular do Curso TESP – Testes de Software, tendo este ocorrido na Altran – Global Delivery Center (GDC) presente na cidade de Fundão. Este estágio teve a duração de 4 meses, dando-se o seu início no dia 8 de Março de 2018 e findando a 13 de Julho de 2018.

Este relatório pode dividir-se em três partes lógicas sendo que a primeira parte, composta pelos dois primeiros capítulos, sendo o primeiro capítulo dedicado a uma breve introdução ao presente relatório. O Capítulo II é uma descrição da empresa onde o estágio decorreu, a Altran, começando com a sua origem e áreas de negócio, concluindo com a presença da Altran no mercado português e uma breve introdução do escritório onde se deu o estágio, a Altran GDC no Fundão.

A segunda parte, onde apenas está presente o Capítulo III, é uma introdução teórica de termos e conhecimentos adquiridos maioritariamente ao longo do Curso TESP – Testes de Software, frequentado por mim. Descrevendo os vários tipos de testes que existem, bem como as atividades necessárias para a execução os mesmos e culminando com uma breve descrição de algumas ferramentas de teste e metodologias de desenvolvimento de Software.

A terceira e última parte, formada pelos dois últimos capítulos, trata-se de uma descrição pormenorizada do trabalho realizado no âmbito do estágio curricular ao abrigo da Altran. Sendo que o Capítulo IV começa por descrever o contexto do projeto bem como as metodologias adotadas pela equipa, de modo a seguir as medidas de qualidade necessárias no âmbito do projeto. No Capítulo V está a descrição da experiência ganha, por mim, durante toda a fase de testes, ao abrigo do projeto em que estive integrado, bem como, das formações que a Altran facultou tanto para o projeto como ainda formações variadas tais como uma formação de Francês A1.

Capítulo I – Introdução

1.1 – Introdução

Este documento foi realizado durante a unidade curricular de Estágio, prevista no segundo semestre do último ano do curso TESP – Testes de Software da Escola Superior de Tecnologia e Gestão no Instituto Politécnico da Guarda.

O estágio foi realizado na empresa de consultoria Altran Global Delivery Center no Fundão, com a duração de 4 meses tendo início no dia 8 de Março de 2018 e com o seu término no dia 13 de Julho de 2018.

Durante este período de estágio foram desenvolvidas várias atividades tais como o estudo de toda a documentação relativa a um projeto comercial, para clarificar todos os objetivos, riscos e metodologias presentes durante o projeto, como também o planeamento e a execução de scripts de teste.

Para além das atividades referidas, que não constam no plano de estágio, foram também desenvolvidas atividades na área de formação como um Curso de Francês – Nível A1, um Curso de Segurança e Higiene no Trabalho e uma outra formação "Q&EMS" vital no contexto do projeto comercial, pois este envolve integração com dispositivos médicos.

Para concluir o presente relatório tem como objetivo demonstrar que os testes de software vão mais além do que apenas detetar erros de implementação e criar documentação.

Capítulo II – A empresa: Altran – Global Delivery Center

2.1 – Grupo Altran[1]

O grupo Altran destaca-se como líder global em serviços de Engenharia, Pesquisa e Desenvolvimento (ER & D).

Como multinacional francesa, esta oferece aos clientes uma proposta de valor inigualável para atender às suas necessidades de transformação e inovação.

A Altran trabalha ao lado dos seus clientes, desde o conceito inicial até a industrialização, para inventar os produtos e serviços de amanhã.

Conta com mais de 30.000 colaboradores, 500 parceiros de relevo a nível mundial com escritórios em mais de 20 países em 3 continentes diferentes.

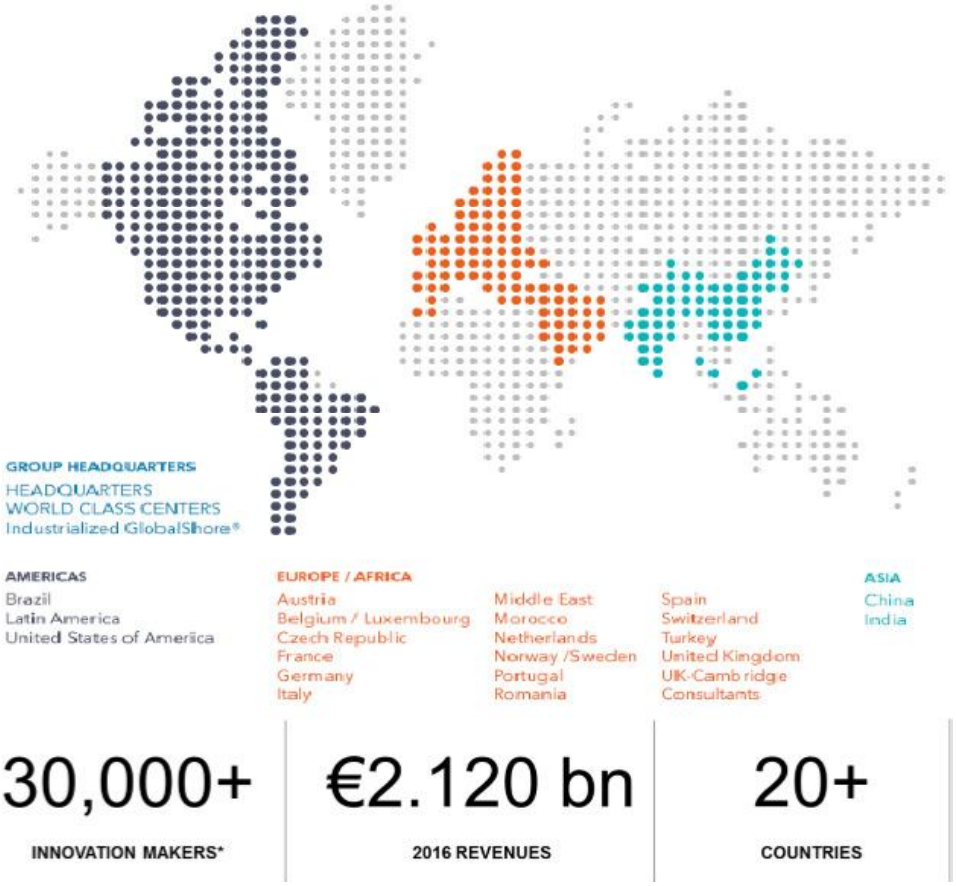

Figura 1 - Grupo Altran no mundo

[1] Toda a informação e imagens constantes deste capítulo foi retirada de www.altran.com

Há mais de 30 anos, a empresa fornece especialização em diversas áreas tais como aeroespacial, automóvel, defesa, energia, finanças, ciências da vida, ferroviária e telecomunicações. Todas estas áreas de especialização estão inseridas nos seguintes domínios, representados na figura 2:

Figura 2 – Domínos Tecnológicos Grupo Altran

Com projetos para algumas das maiores marcas a nível mundial como demonstra a seguinte figura:

Figura 3 - Alguns dos projetos a nível mundial

2.2 – Altran Portugal

Inserida no mercado português desde 1998 mas apenas se consolidando fortemente em 2009, a Altran Portugal conta com mais de 1350 colaboradores como demostra a figura 4, posicionando-se assim como um dos principais *players* do setor da Consultoria em Portugal.

A empresa tem a sua sede em Lisboa estando ainda presente no Porto e Fundão, sendo que no Fundão está presente desde 2013, é classificado como um Global Delivery Center, contado com mais de 470 colaboradores, estando ainda em fase de recrutamento.

Em linha com os objetivos do grupo, a Altran Portugal pretende contribuir para o enriquecimento tecnológico do país.

Figura 4 - Altran em Portugal

De entre os principais fatores para a forte aposta da Altran no GDC encontram-se os fatores económicos, capacidades linguísticas, baixo custo, qualidade das equipas, acessibilidade e ainda a proximidade de vários polos académicos e formação (Universidade da Beira Interior, Institutos Politécnicos de Castelo Branco e da Guarda).

Figura 5 - Altran Fundão (GDC)

2.2.1 – Estrutura Máxima de Organização

Segue-se na figura seguinte a estrutura máxima de organização da Altran Portugal:

Figura 6 - Estrutura de Organização Altran Portugal

Capítulo III – Enquadramento Teórico

3.1 – Introdução

Neste tópico irei abordar alguns dos conceitos que foram lecionados ao longo do TESP de Testes de Software, que considero essenciais para um melhor entender deste mesmo relatório, das atividades que foram desenvolvidas ao longo do estágio e que eu considero que têm uma maior importância.

Os conceitos a apresentar são:

- O que são testes de Software;
- Os tipos de testes de Software;
- Atividades relativas aos testes de Software;
- Ferramentas;
- Metodologias.

3.2 – O que são os testes de Software

Os testes de software são uma das fases mais importantes no processo de desenvolvimento de software pois visam proporcionar um nível mais elevado de qualidade independentemente do produto, seja este um sistema embebido, aplicação móvel, website ou até mesmo um jogo.

Pode ser considerada a etapa de controlo de qualidade, pois assegura que as funcionalidades estão a ser executadas da forma esperada e de acordo com os requisitos contratuais, legais e normas específicas da indústria. Devido a este grau de importância no ciclo de vida de um projeto de software, esta deve ser a última etapa na elaboração de qualquer sistema informático. O principal objetivo dos testes é encontrar defeitos no produto, para que os mesmos possam ser corrigidos pela equipa dos desenvolvedores de software antes do produto ser entregue ao cliente, muitas pessoas pensam que o testes de software servem apenas para demonstrar que o produto está a funcionar da forma esperada, mas na verdade, estes são utilizados como uma forma de encontrar defeitos e garantir a qualidade do produto como um todo e não apenas dos componentes visuais do produto.

As empresas dependem muito dos seus produtos de software e qualquer erro pode significar perdas tanto a nível financeiro como também ao nível de confiança na marca, pois os clientes desse mesmo produto poderão procurar outras alternativas na concorrência.

Estes defeitos ao serem detetados e reportados antes da entrega do sistema podem reduzir o custo, tempo e esforço necessário para a correção do mesmo, pois ao identificar esses erros de forma precoce evita que todo o planeamento e desenvolvimento do código tenha que ser feito à posteriori, podendo até a equipa de desenvolvimento original já não existir.

As informações relativas aos testes devem ser assertivas e concretas sobre o software ou sistema que está a ser testado, de maneira a ajudar na tomada de decisão das partes interessadas sobre a passagem à próxima fase de desenvolvimento ou até mesmo para definir a data de entrega do produto final ao cliente.

3.3 – Tipo de Testes de Software

3.3.1 – Testes Funcionais

Testes funcionais testam os requisitos funcionais da aplicação. De forma resumida, estes verificam se a aplicação reproduz os comportamentos espectáveis e definidos pelos requisitos do projeto, tal como descritos no documento de requisitos de sistema bem como nos *Use Cases*, caso estes existam.

Para tal, são fornecidos dados de entrada, o teste é executado e o resultado é obtido e comparado ao resultado esperado.

O teste funcional pode ser manual, automatizado ou uma mistura dos dois.

3.3.2 – Testes Não Funcionais

São testes que verificam atributos de um componente do sistema que não se relacionam com a funcionalidade por exemplo a confiabilidade, eficiência, usabilidade e portabilidade do produto em questão.

3.3.3 – Testes Automatizados

Os testes automatizados são desenvolvidos através de um software de testes que permite controlar a execução do mesmo, comparando os resultados obtidos com os resultados esperados.

A automação de testes traz diversas vantagens: é mais rápido, diminui a probabilidade de existência de erro humano e reduz o esforço necessário em tarefas repetitivas, deixando os *testers* livres para executarem atividades que exijam raciocínio humano.

3.3.4 – Testes de Sistema

Os testes de sistema são uma das fases de teste de software e hardware no qual o sistema já se encontra completamente integrado e é verificado contra os seus requisitos num ambiente de produção.

Estes testes também podem ser incorporados na técnica de testes de caixa-preta, pois não requerem conhecimento prévio da estrutura lógica do sistema a ser testado.

Este teste normalmente é feito durante a fase de entrega de um produto de software ou de uma funcionalidade em peculiar para garantir que todas especificações estão a ser cumpridas. (TestesW, 2018)

3.3.5 – Testes de Integração

Testes de integração permitem garantir o funcionamento de cada um dos componentes com os demais, sendo principalmente útil como forma de validar a interação entre os vários módulos de um produto de software, tais como a interação entre o módulo de comunicação de rede e o módulo de persistência de dados. Podendo-se dizer que um teste de integração é composto por diversos testes de unidade não só dentro de cada componente mas também às interações "inter-componentes". (TestesW, 2018)

3.3.6 – Testes de Regressão

Teste de regressão não se trata de um nível de teste, mas sim de uma estratégia importante para redução de "efeitos colaterais", especialmente quando se tratam de sistemas legados. Consistem na aplicação, a cada nova versão do software ou a cada ciclo, de todos os testes que já foram elaborados em versões ou ciclos de teste anteriores do sistema com a finalidade de garantir que qualquer funcionalidade já providenciada anteriormente continue a funcionar tal como é espectável. Podendo-se aplicar em qualquer nível de teste. (TestesW, 2018)

3.3.7 – Testes Unitários

Os testes unitários têm por objetivo explorar a menor unidade do projeto de software, procurando identificar falhas geradas em defeitos na implementação da lógica de negócio de cada módulo. O universo alvo deste tipo de teste é o conjunto de métodos de cada objeto independentemente do seu tamanho, em linhas de código. Normalmente são criados e executados pelo programador ao longo do desenvolvimento do sistema. (GSTI, 2018)

3.3.8 – Testes de Segurança

Os testes de segurança servem para identificar possíveis falhas de segurança de uma aplicação. Estes testes podem ser automatizados, bem como, testes manuais. Usualmente concentram-se em explorar os diversos papéis, perfis e permissões para navegar no sistema de forma ilícita. (GSTI, 2018)

3.3.9 – Testes de Stress

Este tipo de teste serve para levar o software ao seu limite de potência e funcionamento, de modo a avaliar possíveis pontos de rutura de entre todos os componentes. Isto é feito para verificar se suas especificações máximas ou mínimas de uso estão dentro dos parâmetros de aceitação por parte do cliente. (GSTI, 2018)

3.3.10 – Testes Exploratórios

Os testes exploratórios tal como o nome indica é um teste que é feito de forma exploratória e de maneira informal, serve para testar componentes de software que não estão nos scripts de teste. Estes testes não requerem qualquer planeamento significativo e tolera uma documentação limitada do objetivo do teste. Esta técnica de teste depende principalmente da habilidade e do conhecimento do tester para guiar o teste e usar um ciclo de feedback ativo para orientar e definir o esforço necessário.

3.4 - Atividades Relativas aos Testes de Software

Os testes de software não se resumem apenas à sua execução, pois, representa a estruturação de etapas, atividade, artefactos, papéis e responsabilidades que procuram um padrão dos trabalhos de forma a aumentar o controlo dos testes.

O processo de teste, como qualquer outro, deve ser revisto continuamente, de forma a ampliar a sua atuação e possibilitar aos profissionais uma maior visibilidade e organização do seu trabalho, o que resulta numa maior agilidade e controlo dos projetos de testes. (TestesW, 2018)

O esquema seguinte demostra as fazes de teste (Figura 7)[2]:

Figura 7 - Esquema Ciclo de Vida de um Teste

[2] Esta imagem foi retirada do seguinte web site: https://pt.dreamstime.com/foto-de-stock-conceito-do-ciclo-de-vida-dos-testes-do-software-image66511531

3.4.1 – Análise de Requisitos

A análise de requisitos é uma das primeiras atividades de desenvolvimento de um software, se não mesmo a primeira. O fruto do seu trabalho é a especificação de requisitos, que define o âmbito do software em duas dimensões: requisitos funcionais e requisitos não-funcionais. É durante esta fase que decorrem as primeiras reuniões com o cliente e/ou utilizadores do software para conhecer as funcionalidades do sistema que será desenvolvido. É também durante esta fase onde ser originam a maior parte dos erros, pois a falta de experiência dos clientes faz com que estes nem sempre sejam claros em relação às funcionalidades que o software final deverá conter. (Quinterio, 2018)

3.4.2 – Planeamento

Nesta etapa define-se uma proposta de testes e os seus objetivos, baseando-se nas expectativas do cliente em relação a prazos, custos e qualidade esperada, possibilitando dimensionar a equipa e estabelecer um esforço de acordo com as necessidades apontadas pelo cliente. Ao longo do planeamento é necessário comparar o plano de desenvolvimento com o plano de testes e se necessário efetuar as devidas correções.

3.4.3 – Desenvolvimento dos Casos de Teste

Após o planeamento o próximo passo é começar a desenvolver os casos de teste de acordo com os requisitos.
Alguns erros podem ser derivados de alguma falha no caso de teste, um caso de teste bem descrito e estruturado é uma mais-valia para o teste.

3.4.4 – Configuração do Ambiente

Esta é uma das partes mais importantes na fase de testes, pois é necessário um ambiente minimamente isolado para executar os testes. Qualquer falha na configuração do ambiente pode provocar efeitos colaterais no ambiente levando a falsos positivos durante a execução dos testes.

3.4.5 – Execução dos Testes

Na execução de testes tal como o nome indica é a fase onde os testes são executados, após a sua execução é necessário fazer um relatório do estado dos testes, se estes falharam, se passaram ou se ficaram pendentes por alguma razão.

Se todos os testes passarem dá-se início à próxima etapa, se não, o defeito deve ser reportado e é necessária uma correção por parte do programador, e após essa correção o teste tem de obrigatoriamente ser executado novamente.

3.4.6 – Finalização ou fecho das Atividades

Esta é a última etapa do processo de um teste em que todos os documentos são arquivados e são entregues à equipa que irá suportar o software em questão.

3.5 - Ferramentas de Teste

3.5.1 – Vantagens e Desvantagens da utilização de ferramentas de teste

A utilização de ferramentas de suporte ao teste pode contribuir consideravelmente para um aumento da qualidade do produto final, permitindo, por exemplo, otimizar de certa forma o processo de testes, bem como para as etapas de ciclo de vida de um teste, porém o uso de ferramentas também acarreta as suas desvantagens. Por isso antes de adotar qualquer ferramenta há que ponderar alguns aspetos. (DevMedia, 2018)

Em baixo, seguem-se algumas das principais vantagens que as ferramentas podem apresentar:

- Facilita de acesso à *testdata*;
- Reduz a repetição de tarefas;
- Simplifica o processo de automatização de testes;
- Aumenta a rastreabilidade dos testes;
- Auxilia no agendamento da etapa de execução dos testes.

No que toca às desvantagens proporcionadas pela utilização de ferramentas para auxiliar nos testes, é necessário ter em conta os seguintes pontos:

- Expectativas irreais acerca da ferramenta;
- Expectativas irreais sobre o retorno da ferramenta;
- Qualquer custo inerente à ferramenta (caso esta seja paga);
- Estimar tempo e esforço que os *testers* vão necessitar para aprendizagem da ferramenta.

3.5.2 – Tipos de Ferramentas de Teste

No mercado de ferramentas de teste existem inúmeras disponíveis, desde pagas a gratuitas.

Estas podendo serem agrupadas em 4 principais categorias indicadas na tabela abaixo:

Tipo de Ferramenta	Função	Exemplo
Gestão de Testes	• Gestão de Requisitos • Registros da execução dos testes • Registro dos defeitos • Relatórios da execução	HP Quality Center
Gestão de Requisitos	• Esta armazena os requesitos e ajudam na identificação de requisitos incositentes ou em falta no sistema	Borland - Atlas
Gestão de Incidentes	• Armazena e gere os relatórios dos defeitos encontrados • Ajuda na gestão do ciclo de vida dos defeitos	Jira
Execução de Testes	• Permite que os testes possam ser executados semi-automaticamente ou automaticamente	Selenium

Tabela 1 - Tipos de Ferramentas de Teste

3.6 – Metodologias de desenvolvimento de Software

Atualmente existem diversos modelos arquiteturais para garantir a qualidade dos seus produtos de software, por parte das empresas. O objetivo principal desses mesmos modelos de qualidade é fazer com que toda a equipa trabalhe, de forma focada, para o mesmo objetivo final, de modo a evitar desperdiçar tempo e por consequência custos.

Para além dos modelos arquiteturais de desenvolvimento de software, também é importante que as organizações também incorporem todo o processo de testes de software durante o desenvolvimento dos seus produtos.

As metodologias de teste especificam padrões sobre como os mesmos devem ser executados, como criar a documentação necessária, aumentar a velocidade de desenvolvimento do software e melhorar a comunicação entre a equipa de forma a tornar mais eficiente todo o processo de desenvolvimento do sistema informático.

A utilização deste tipo de metodologias previne muitos problemas, maioritariamente no que toca à redução de custos de manutenção do produto e minimizando ainda os erros no software entregue ao cliente final, que podem resultar na perda de confiança no produto por parte dos mesmos. Com um aumento do tempo disponível, a equipa pode também investir em melhorias e novos planos para evitar e facilitar todo o processo de validação e teste do produto final.

De entre as diversas metodologias adotadas para o desenvolvimento de software as mais populares e que foram usadas no projeto em que fui inserido são a metodologia em Cascata e ainda a metodologia Ágil. De seguida segue-se uma ligeira introdução às mesmas.

3.6.1 – Metodologia em Cascata

O modelo cascata ou clássico também pode ser conhecido como uma abordagem do tipo "top-down", tendo sido criado na década de 1970, sendo o modelo mais usado no desenvolvimento de sistemas até a meio da década de 1980. Este modelo foi desenvolvido com a finalidade de especificar uma ordem para o desenvolvimento de produtos de software. Comparado com os outros modelos de desenvolvimento de software, o modelo em cascata caracteriza-se por ser mais rígido e menos administrativo.

É um dos modelos mais importantes e é utilizado como referência para a implementação de outros modelos modernos, sendo utilizado como base para a maioria dos projetos modernos. Ao longo do tempo, o modelo cascata foi melhorado ao longo do tempo e continua a ser utilizado atualmente, estando a cair em desuso devido à maior adoção de metodologias ágeis.

O objetivo principal desta metodologia é que as diferentes fases de desenvolvimento seguem uma sequência, da primeira etapa do processo de desenvolvimento segue-se para a segunda e assim sucessivamente até ao final do ciclo de desenvolvimento. Desta forma, as atividades que devem ser executadas são reunidas em tarefas e executadas sequencialmente, onde a próxima tarefa só inicia quando a anterior estiver completamente finalizada. (Consultoria, 2018)

A figura seguinte esquematiza a metodologia em cascata (Figura 8)[3]:

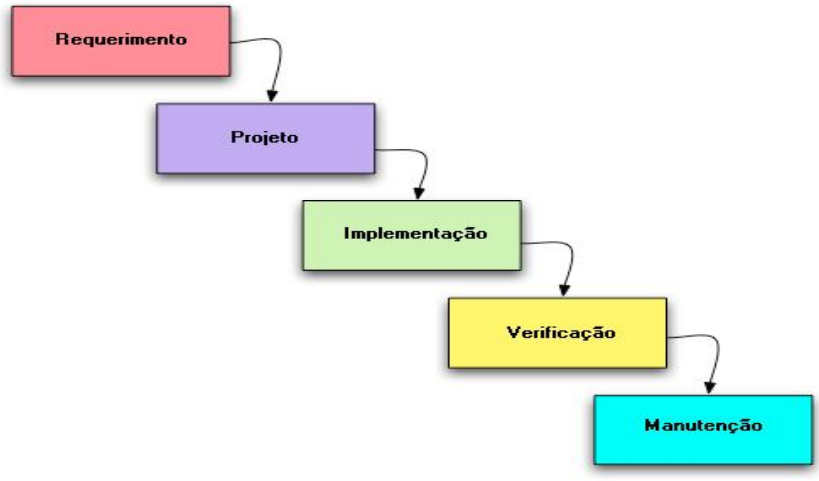

Figura 8 - Esquema Metodologia em Cascata

[3] Esta imagem foi retirada do seguinte web site: https://pt.wikipedia.org/wiki/Modelo_em_cascata

3.6.3 – Metodologias Ágeis

As metodologias ágeis, ou Agile em inglês, é considerada nos últimos anos como uma alternativa para atender às necessidades de clientes e projetos que requerem cada vez mais flexibilidade e dinamismo resultando num grande aumento de produtividade, por parte da equipa de desenvolvimento.

No desenvolvimento ágil, é utilizada uma abordagem de planeamento iterativa. Enquanto nos métodos tradicionais todas as etapas do projeto são documentadas detalhadamente, desde o início até o fim do projeto, no método ágil é realizado em etapas curtas, chamadas iterações (ou *sprints*). (Consulting, 2018)

Os principais princípios das metodologias Agile são:

- Entregas iterativas e contínuas do software, ou dos seus componentes;
- Reagir a mudanças de requisitos, independentemente da fase em que o projeto de encontra;
- Fazer pequenas entregas do sistema ao cliente, permitindo que o cliente faça uma avaliação continua, preferencialmente em semanas;
- Cooperação diária com a equipa do projeto;
- Maior comunicação e paralelismo entre os vários elementos da equipa.

A figura seguinte demonstra o desenvolvimento ágil (Figura 10)[4]:

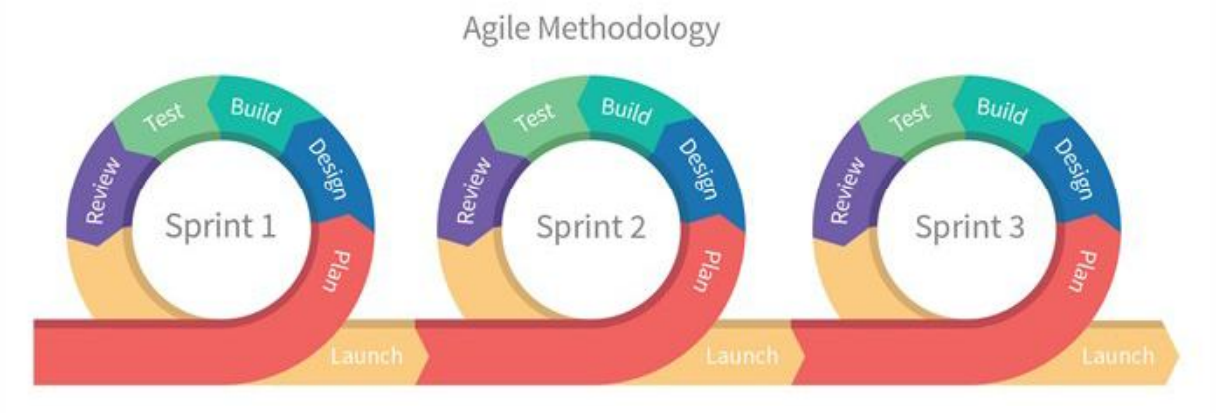

Figura 9 – Esquema Metodologias Ágeis

[4] Esta imagem foi retirada do web site: https://robsoncamargo.com.br/blog/o-que-e-metodologia-agil

De entre as diversas metodologias ágeis usadas, o *Scrum* é uma das mais difundidas, especialmente pela forma dinâmica como as etapas dos projetos são desenvolvidas.

Num projeto realizado utilizando a metodologia *Scrum*, a execução acontece em iterações definidas anteriormente, denominadas por *Sprints*. Estes podem ser vistos como ciclos de desenvolvimento que começam com uma reunião de planeamento do *Sprint* (*Sprint Planning*), onde são definidas as tarefas para o Sprint, bem como quais as prioridades e quem vai implementar cada tarefa. No final do *Sprint*, usualmente 2 ou 3 semanas após o seu começo, há uma reunião para rever as tarefas completadas durante o *Sprint*, intitulado de *Sprint Review*, que usualmente é realizada, por meio de uma demonstração do produto para o resto da equipa e eventualmente para o cliente. Por fim, realiza-se uma reunião de avaliação do *Sprint*, *Sprint Retrospective* ou *Lessons Learned*, onde a equipa compara as tarefas que foram elaboradas versus as tarefas planeadas bem como as estimativas dadas para cada tarefa, tudo com o intuito de corrigir os processos para o próximo *Sprint*. Todo o desenvolvimento num Sprint é acompanhado por reuniões diárias, em pé, chamadas de *Daily Scrum Meeting*. (Udacity, 2018)

Uma *Sprint* pode durar qualquer unidade de tempo, mas a recomendação é que as mais curtas levem uma semana e as mais longas, no máximo um mês. Quando uma equipa define um padrão de duração da *Sprint*, a mesma tem como costume ser utilizada em todas as iterações do projeto. As *Sprints* são sucessivas e, assim, a seguinte somente é iniciada no final da anterior.

A figura seguinte esquematiza a metodologia *Scrum* (Figura 11)[5]:

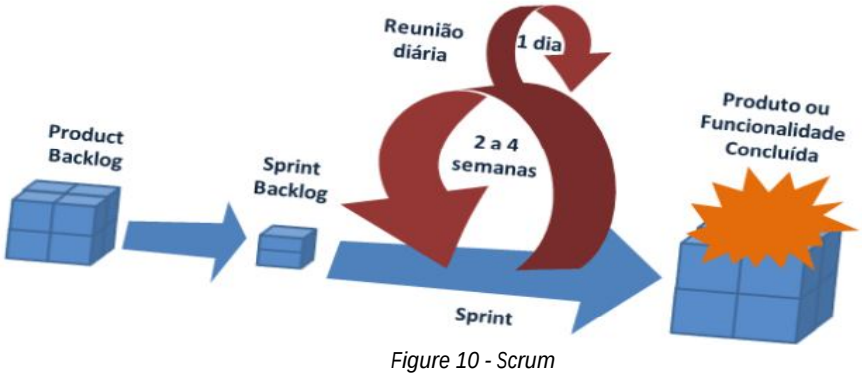

Figure 10 - Scrum

[5] Esta imagem foi retirada do seguinte web site: http://www.mindmaster.com.br/scrum/

Capítulo IV – O projeto

4.1 – O cliente: Confidencial

Infelizmente, por motivos de confidencialidade e a pedido do próprio cliente o seu nome não poderá ser usado no presente relatório.

O que é permitido adiantar sobre o mesmo é que este é uma empresa farmacêutica de relevo global, trabalhando principalmente na área da indústria bio-farmacêutica, focando-se na descoberta, desenvolvimento e comercialização de medicamentos prescritos com o objetivo principal o tratamento de doenças em qualquer uma das 4 áreas seguintes:

- Oncologia;
- Doenças Cardiovasculares;
- Doenças Metabólicas;
- Doenças Respiratórias.

4.2 – Âmbito e descrição do Projeto

Este projeto tem como principal objetivo desenvolver uma plataforma para a monitorização do uso de inaladores para pacientes com Asma ou COPD. Dividindo-se assim em 3 aplicações formais, duas aplicações móveis e um portal de administração.

De forma sucinta, um paciente sofredor de asma ou COPD, terá um inalador inteligente que se conecta à aplicação móvel para reportar a quantidade de inalações que necessitou de tomar ao longo do dia, bem como receberá no seu smartphone notificações para relembrar o mesmo de inalar a uma dada hora, dependendo da receita passada pelo seu médico. Quando este paciente se desloca ao consultório o médico poderá aceder aos registos do paciente, através do portal, e verificar se os sintomas se estão agravando ao longo do tempo.

Acima de tudo, trata-se de um projeto onde a maioria do código, das aplicações móveis e portal, é herdado de um projeto semelhante logo há uma maior ênfase em garantir que qualquer mudança no código legado não crie efeitos secundários nefastos, capazes de comprometer a integridade das aplicações como um todo.

4.3 – Funcionamento interno do projeto

4.3.1 – Metodologia de desenvolvimento

Devido ao projeto lidar com equipamentos médicos é necessário ter um cuidado redobrado, havendo uma série de processos e procedimentos a serem seguidos.A própria metodologia de desenvolvimento adotada para este projeto é também influenciada por consequência da área de foco do projeto, pois a mesma obrigatoriamente tem de ser cascata.

O uso do modelo em cascata neste tipo de projetos tem a vantagem ser necessário terminar cada processo de modo a avançar para o seguinte até que o cliente possa validar e aceitar o produto final como um todo. Apesar de ser este o modelo adotado, houve um esforço por parte de todos os *stakeholders* de modo a que o cliente pudesse participar ativamente ao longo do decurso do projeto, alinhando assim o sistema com a sua visão inicial e minimizando o impacto da compreensão adquirida no seu decurso sendo que para proceder a qualquer alteração, que se encontra fora do âmbito original, terá que formalizar um pedido de *Change Request*.

Ao utilizar este modelo "híbrido" garante-se que todos os processos, tarefas e o sistema como todo esteja de acordo com o requisitado e completamente alinhados com as expectativas do cliente. Para uma melhor comunicação, dinâmica e entendimento entre a equipa foram adotados algumas práticas de Scrum como por exemplo Daily Meetings e divisão de trabalho por Sprints.

4.3.2 – Constituição atual da equipa

A seguinte tabela demonstra a equipe atual do projeto:

Nome	Responsabilidades
Marcus Pereira	Technical Project Manager
Aline Cabral	Test Manager
Gustavo Gomes	Android Devloper
Nissi Miranda	iOS Developer
André Nunes	Web Developer
Alexandre Guerra	Tester
Eduardo Almeida	Tester
Inês Viera	Quality Manager

Tabela 2 - Constituição atual da Equipa

4.3.3 – Fluxo de Trabalho

Tendo em conta que o projeto se insere na área da Saúde foi necessário seguir uma metodologia em cascata, de forma a seguir os padrões definidos pela ISO 13485 e IEC 62304. Na obstante, todo o processo foi agilizado de modo a se aproximar o mais possível com Scrum, tudo com o intuito de tirar proveito das vantagens que esta metodologia ágil confere.

Devido a esta metodologia híbrida, os *Sprints* foram criados e formalizados para o desenvolvimento e subsequentes testes a todos os componentes do sistema. Com esta nova disposição e tirando partido do tamanho menor do projeto, foi possível dividir o projeto em 4 Sprints.

- Sprint 1 - Dedicada ao desenvolvimento do Produto, bem como à formalização e aprovação dos documentos pelo cliente (*Test Plan*, *Test Strategy*, *Test Scrips*).
- Sprint 2 - Com foco exclusivo nos *System Tests* e eventuais correções de defeitos encontrados
- Sprint 3 - Destinada aos testes **informais** de aceitação e alinhamento por parte do cliente
- Sprint 4 - Sprint final de testes de aceitação formais.

Durante a primeira *Sprint* não houve grandes atividades no que toca aos testes, pois o/a *Test Manager* ainda estava a criar os *Test Scripts* baseando-se nas *User Stories* aprovadas pelo cliente. Em paralelo a equipa de desenvolvimento já estava a implementar as funcionalidades necessárias para ir de encontro ao estipulado nas *User Stories.* Todos os dias realizou-se uma daily meeting, típica de um projeto Scrum, para a troca de informação entre toda a equipa do projeto.

Após toda a implementação estar concluída e os *Test Scripts* aprovados pelo cliente, iniciou-se o segundo Sprint dedicado apenas a *System Tests* (Testes de Sistema). Durante esta fase a reunião diária para troca de informação continuou a realizar-se sempre com o objetivo de maximizar a comunicação entre os vários membros da equipa de forma a tornar todo o processo mais ágil. Esta foi a fase de maior esforço, especialmente pela equipa de testes, pois os *Test Scripts* têm que ser realizados de forma formal, isto é, com toda a documentação necessária para um projectão na área da Saúde. O *Test Script* teria que ser seguido num dos dispositivos aprovados de modo a que cada passo fosse registado como sucesso ou erro, estando também sempre em análise qualquer desvio na implementação dos layouts das aplicações como um todo.

O terceiro ficou marcado pela análise e alinhamento dos produtos, aprovados durante o Sprint de *System Tests,* por parte do cliente. Durante este as reuniões diárias eram especialmente usadas para debater qual seria o procedimento para cada bug reportado pelo cliente. Cada novo defeito reportado pelo cliente era validado e reproduzido pela equipa de testes, de modo a garantir a sua existência, e era aberto um defeito no *JIRA* para que o programador pudesse proceder para a sua correção. Após a correção do defeito este assinalava o estado no *JIRA* para que um elemento da equipa de testes pudesse validar a solução encontrada pelo programador.

No quarto, e último, *Sprint* o cliente procedeu à validação formal do sistema, sendo que esta foi a fase menos ativa de toda a equipa.

4.3.4 – Fluxo de Report de Bugs

Durante a fase de execução dos testes de sistema referentes ao projeto onde fui integrado, era espectável que fossem encontrados defeitos, sejam eles de conteúdo, design ou suas nas funcionalidades.

Após as falhas serem encontradas todo um fluxo para a sua correção e verificação deve ser seguido pelos vários elementos da equipa, usando o JIRA como ferramenta de registo e gestão dos erros encontrados, sendo todo este processo descrito na figura seguinte (Figura 12):

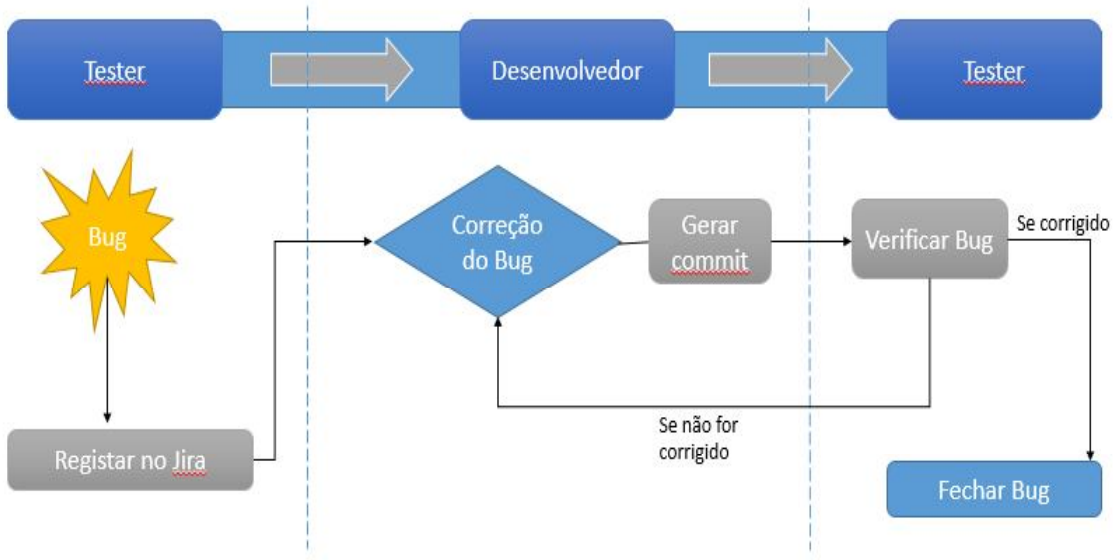

Figure 11 - Fluxo de Report de Defeitos – Fonte própria

Sempre que é reportado um defeito, é necessário descrever o mesmo de forma concisa e clara munindo com todas as evidências precisas, tais como, os passos a seguir para reproduzir a falha, a versão do sistema onde o defeito ocorre, a plataforma onde a mesma foi encontrada (iOS, Android, Web), a sua prioridade, o Sprint atual e por fim, caso vantajoso, capturas de ecrã ou vídeos para demonstrar a existência da falha.

4.4 – Ferramentas utilizadas no âmbito do Projeto

4.4.1 – JIRA

O JIRA é um software comercial desenvolvido pela empresa Australiana Atlassian. A ferramenta permite a monotorização de tarefas e o acompanhamento de projetos garantindo que todas as suas atividades sejam geridas num único repositório.

Um Projeto JIRA é um agrupamento de tarefas e que é definido de acordo com a organização e consoante as suas necessidades.

Alguns exemplos de projetos JIRA:

• Projetos de desenvolvimento de software;

• Projetos de campanha de marketing;

• Controle de processos ITL.

Este é baseado na linguagem de programação Java e que opera em várias bases de dados e sistemas operativos

A ferramenta também apresenta painéis de controlo que podem ser adaptados consoante o projeto como filtros de pesquisa e estatísticas, devido a uma arquitetura flexível de parte do JIRA, permite que o utilizador crie extensões específicas e essas podem ser incluídas numa biblioteca de extensão.

Durante o projeto esta foi utilizada para o reporte e gestão de defeitos, local onde eram anexadas as evidencias, onde se definia a prioridade para a sua correção, a versão afetada pelo defeito, bem como o seu responsável.

Pela experiência obtida no decorrer do projeto, em minha opinião, é uma ferramenta intuitiva e oferece um ótimo suporte no que toca a gestão no desenvolvimento de um software. (Atassian, 2018)

A imagem seguinte representa o logótipo da ferramenta (Figura 13)[6]:

Figure 12 - JIRA - Atlassian

[6] Esta imagem foi retirada do seguinte web site: https://www.atlassian.com/software/jira

4.4.2 – Test Fairy

No desenrolar do projeto esta foi usada como plataforma de armazenamento das aplicações móveis onde os artefactos referentes às várias versões podem ser acedidos.

Aquando do lançamento de uma nova versão era nesta plataforma que os arquivos iriam ser descarregados pela equipa de testes.

Sempre que eram aplicadas correções a defeitos detetados em fases anteriores, era necessário, em seguida, ser gerada uma nova *release*, contendo uma breve descrição do defeito bem como conter novo número de versão correspondente ao novo artefacto gerado. (TestFairy, 2018)

A imagem seguinte representa o logótipo da seguinte ferramenta (Figura 14)[7]:

Figure 13 - TestFairy

[7] Esta imagem foi retirada do seguinte web site: https://www.testfairy.com/

Capítulo V – Trabalho Desenvolvido

5.1 – Introdução

O decorrer do estágio curricular na empresa Altran Portugal foi-me possibilitada a aplicação e o aprofundamento dos meus conhecimentos, técnicas e metodologias, adquiridas durante o TESP – Testes de Software, bem como a aprendizagem de novos, que apenas em ambiente profissional e num projeto de mundo real podem proporcionar.

Além das atividades relacionadas com o projeto e dos conhecimentos técnicos relacionados com a área de testes, ainda me foi proporcionado a participação em formações relacionadas com o projeto em que fui inserido, linguísticas e formações de foro profissional.

Neste capítulo irei abordar de uma forma mais aprofundada todas essas atividades e formações.

5.2 – Trabalho desenvolvido

Neste capítulo serão abordadas, de forma cronológica todas as atividades desenvolvidas ao longo do projeto realizado durante o estágio curricular.

5.2.1 - Estudo do projeto como um todo e seus procedimentos

A primeira tarefa realizada no âmbito do projeto consistiu na leitura e compreensão dos documentos já gerados para o projeto em questão, tais como, o *Test Plan* e o *Test Strategy*. A leitura destes documentos serviu para um conhecimento de todos os procedimentos (SOP´s), bem como, conhecer o projeto como um todo, pois trata-se de um projeto ligado à indústria farmacêutica e por isso também ligado à saúde de modo que necessita de seguir procedimentos estritos durante todas as fases do projeto.

5.2.2 – Adaptação de User Stories para Test Scripts

Por este ser um projeto em que grande parte do seu código fonte foi herdado, as *User Stories* que não sofreram qualquer modificação para o novo projeto estão dispensadas de serem testadas de novo durante a fase de *System Tests*. Contudo para as *User Stories* que sofreram alterações foi necessário redesenhar o *Test Script* para que este consiga validar a *User Story* dentro dos novos parâmetros descritos na mesma. Estes *Test Script* sofreram especialmente alterações no que toca à descrição passo a passo pois existiram modificações especialmente aos fluxos de navegação, bem como, nas pré-condições que são necessárias cumprir.

Esta tarefa necessita de ser tratada com cuidado extremo, pois qualquer erro durante a sua composição ou abrangência relativa a uma funcionalidade pode causar variados prejuízos na fase de testes do projeto. Estes prejuízos podem traduzir-se, por exemplo, em perdas de tempo, entrega de funcionalidades que não traduzem os requisitos na sua totalidade, ou ainda, em falsos positivos durante a fase de *System Tests* que só serão detetados pelo cliente em *UAT*.

Consequentemente por esta que necessitar um elevado grau de conhecimento e experiência é uma tarefa da responsabilidade do/a *Test Manager*, sendo que os membros da equipa de testes podem ajudar na sua consumação, mesmo não sendo da sua responsabilidade.

No decorrer da minha experiência no projeto esta foi uma tarefa que me foi pedida e que envolve algumas fases descritas em baixo:

- Identificação das funcionalidades a serem testadas;
- Identificação dos cenários a serem testados;
- Identificação das condições necessárias para o teste;
- Planeamento do decorrer do Teste.

Depois da sua elaboração, os *Test Scripts* passam pela fase de aceitação, feita por parte do cliente, onde ele pode concordar ou discordar com as validações presentes no *Test Script* e apenas quando este é aceite é que se pode passar para a fase de execução.

5.2.3 – Testes Exploratórios

Os testes exploratórios são, tal como o nome indica, um tipo de teste que é executado de forma exploratória e informal, sem documentação inerente aos mesmos.

Estes servem para testar funcionalidades que não estão especificadas durante a execução dos *Test Scripts* ou estão presentes nas *User Stories*. Esta técnica depende principalmente da habilidade e conhecimento do *tester* pois é ele que tem que ter capacidade para definir o esforço necessário e conduzir estes testes.

Durante todo o projeto, foram executados vários períodos de testes exploratórios. Graças à sua natureza, estes não têm um caminho a seguir, conseguem identificar defeitos e ainda provocar comportamentos que, provavelmente, nunca chegariam a ser encontrados durante desenvolvimento do produto. Sendo que, tudo isto pode levar a que estes defeitos possam ser descobertos pelo próprio cliente ou utilizadores finais, produzindo uma sensação de falta de qualidade em geral no sistema.

Devido ao facto do sistema desenvolvido, no âmbito do projeto, herdar grande parte do seu código fonte de uma outra solução informática, existem por isso uma série de componentes que já tinham sido testados anteriormente. Este tipo de testes foi principalmente usados para combater a falta de acesso às *User Stories* e *Test Scripts* referentes ao código legado, pois esta era a única forma de testar as várias aplicações, Android, iOS e Portal Web, como um todo e não apenas ficar restringido às novas funcionalidades do projeto.

No decorrer do projeto, os testes exploratórios desempenharam um papel fundamental, poupando tempo e esforço necessários para alcançar as metas definidas. Concluindo ainda que esta estratégia de testes permitiu encontrar 80% dos bugs totais reportados.

5.2.4 – Testes de Sistema

Os testes de sistema são das últimas fases do desenvolvimento de um sistema, pode ser chamado de teste final caso não seja encontrado nenhum defeito durante esta etapa ou á posteriori na etapa de aceitação por parte do cliente, o objetivo deste tipo de teste é verificar que todas as funcionalidades implementadas estão de acordo com os requisitos, acordados com o cliente, este processo é feito de maneira formal, ou seja, gerando a documentação relativa ao mesmo, com a finalidade de servir como prova de que a funcionalidade foi implementada conforme os requisitos e validada pela equipa de testes.

Para esta ação é necessário o recorrer a *Test Scritps*, como forma de guiar o teste, definindo o caminho a seguir, passo a passo, bem como os cenários em que este vai ocorrer. Um *Test Script* é composto por várias ações e cada uma delas apenas pode ter um resultado, sendo que esse resultado tem de estar alinhado com os requisitos, definidos anteriormente.

Os passos descritos em cada *Test Script* podem ser assinalados com um rótulo de PASS, caso esse passo esteja correto, ou FAIL, caso exista uma divergência entre o resultado obtido após o passo e o que era esperado. Para os casos de erro, ou seja de FAIL, é necessário gerar evidências de modo a comprovar que o erro existe e ainda uma breve descrição sobre o mesmo.

Quando um defeito é detetado, para além de ser assinalado no *Test Script* é ainda necessário que este seja reportado na plataforma de gestão de erros (JIRA), a fim de ser corrigido pelo programador e validado de novo contra o *Test Script* onde o erro foi detetado. Basta que apenas um dos passos falhe para que o teste seja considerado FAIL, de forma inversa caso todos os passos estejam de acordo com o resultado esperado aí o teste é considerado validado e não será necessário mais nada por parte do Tester, para além, claro, de gerar as evidências de que o teste ocorreu.

Esta documentação, normalmente, tem como seu título a funcionalidade a ser testada e é necessário o preenchimento de vários campos, como o dispositivo em que vai a ser realizado o teste, a sua respetiva versão, quem o executa, a data em que o teste começou a ser executado e a data em que foi terminado.

Após terminada a execução do *Test Script*, este necessita das assinaturas do/a Tester, do/a *Test Manager* e ainda do/a Gerente do Projeto, sendo de seguida sujeito a uma revisão pela equipa de qualidade da empresa, garantindo assim que todas as normas e procedimentos foram seguidos de forma correta e com o rigor esperado, só depois de ser validado por esta entidade este poderá ser considerado como terminado e o sistema seguirá para a aceitação, por parte do cliente.

O meu papel no âmbito desta fase foi principalmente o de executar este tipo de teste em equipamentos móveis, tanto em Android como em iOS, como ainda gerar todas as evidências de teste e preencher toda a documentação necessária, por fim, caso algum erro fosse detetado também teria de proceder ao registo do mesmo na plataforma de gestão de erros.

5.2.5 – Testes de Regressão

Numa das fases do projeto devido a uma inconsistência que estava a provocar variados tipos de danos colaterais no sistema e em concordância com a equipa do projeto a melhor solução estratégica para garantir que o resto das funcionalidades estavam a ter o resultado esperado de acordo com o definido e também identificar qualquer outro tipo de dano colateral em funcionalidades já testadas foi necessário apelar aos testes de regressão, em que todos os testes executados às funcionalidades anteriormente tiveram de ser repetidos.

5.3 – Formações

5.3.1 – Q&EMS

Derivado à natureza do projeto, onde fui alocado, existe uma série de exigências e procedimentos que necessitam de ser obrigatoriamente seguidos, todos estes são providenciados através de uma formação de cariz obrigatório, à qual eu participei.

Esta formação teve como principal objetivo dar a conhecer todos os procedimentos relativos ao desenvolvimento de software para "*Medical Devices*" (dispositivos médicos), bem como, todos os documentos que precisam de ser gerados e ainda a sua correspondente estrutura.

Esta formação proporcionou uma melhor compreensão do projeto, como também, contribuiu para um enriquecimento curricular.

5.3.2 – Formação Linguística de Francês

Durante período em que estive em âmbito de estágio foi também me possibilitado frequentar uma formação linguística de francês, sendo que a mesma teve uma duração total de 50 horas e com o nível adquirido de A1. Esta formação foi providenciada pela Altran como uma formação profissional, incorporada dentro do horário laboral, concluindo a mesma com uma avaliação. Tendo em conta que o Grupo Altran é uma multinacional francesa, com uma panóplia de projetos vasta para França bem como outros países da Europa, as formações linguísticas para colaboradores são um investimento regular por parte do grupo. Esta formação foi uma mais-valia pois mune os colaboradores com novas competências e conhecimentos que certamente serão uteis ao longo do percurso profissional destes.

5.3.3 – Formação de SHT (Segurança e Higiene no Trabalho)

Outra das formações facultadas pelo Grupo Altran, foi a formação em Segurança e Higiene no Trabalho. Esta foi elaborada de forma online, através da plataforma interna da empresa, onde foram fornecidos os conteúdos teóricos e depois feita uma avaliação.

A melhoria das condições de Segurança, Higiene e Saúde no Trabalho é hoje uma preocupação generalizada, não só por razões de natureza humana, mas também por razões económicas.

Esta formação teve como principal finalidade a redução de ocorrências de acidentes de trabalho e de doenças relacionadas com o foro profissional, bem como também da forma a atuar em caso de acidentes ou catástrofes naturais, como por exemplo um incêndio nas instalações. Desta forma os colaboradores irão ter uma melhor qualidade de vida no ambiente de trabalho.

Conclusão

No decorrer deste relatório apresentaram-se os pontos mais importantes do estágio curricular desenvolvido. Este percurso realizado na Altran permitiu-nos desenvolver, na prática, os conteúdos abordados nas unidades curriculares do plano de estudos do Curso Testes de Software.

Esta autoavaliação final, sobre o trabalho desenvolvido, é uma etapa importante, uma vez que, ao incidir sobre este período, pode apontar os aspetos fortes que marcaram o nosso estágio, assim como os aspetos menos fortes e sobre os quais teremos que refletir, com o objetivo de melhorar a nossa evolução.

Como pude experienciar, realizar testes em contexto real, é uma etapa importante e necessária. O estágio serviu para concluirmos que os testes de software têm um papel fulcral na vida de um software, pois é assim que se consegue garantir qualidade no produto final.

Bibliografia

Atassian, J. (02 de Junho de 2018). *Jira*. Obtido de Jira: https://www.atlassian.com/software/jira

Consulting, G. (25 de Junho de 2018). *GAEA Consulting*. Obtido de GAEA Consulting: https://gaea.com.br/o-que-e-metodologia-agile-e-quais-tendencias-voce-deve-ficar-atento/

Consultoria, C. d. (01 de Junho de 2018). *Casa da Consultoria*. Obtido de Casa da Consultoria: https://casadaconsultoria.com.br/modelo-cascata/

DevMedia. (05 de Julho de 2018). *DevMedia*. Obtido de DevMedia: https://www.devmedia.com.br

GSTI, P. (19 de Maio de 2018). *Portal GSTI*. Obtido de Portal GSTI: https://www.portalgsti.com.br/testes-de-software/sobre/

Quinterio, A. P. (02 de Junho de 2018). *InfoEscola*. Obtido de InfoEscola: https://www.infoescola.com/engenharia-de-software/analise-de-requisitos/

TestesW. (15 de Maio de 2018). *TestesW*. Obtido de TestesW: https://testesw.wordpress.com/processo-de-testes/

TestFairy. (20 de Maio de 2018). *TestFairy*. Obtido de TestFairy: https://www.testfairy.com/

Udacity. (10 de Junho de 2018). *Udacity*. Obtido de Udacity: https://br.udacity.com/blog/post/metodologia-scrum-agile

Anexos

Anexos I – Instalações Altran Fundão