IPG Politécnico da Guarda
Escola Superior de Tecnologia e Gestão

RELATÓRIO DE ESTÁGIO

Curso Técnico Superior Profissional em
Testes de Software

Andreia Filipa Brazete Esteves

julho | 2016

ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO
Instituto Politécnico da Guarda

RELATÓRIO DE ESTÁGIO

ANDREIA FILIPA BRAZETE ESTEVES

RELATÓRIO PARA A OBTENÇÃO DO DIPLOMA DE

TÉCNICO SUPERIOR PROFISSIONAL

EM TESTES DE *S*OFTWARE

06/2016

Índice

Lista de Tabelas

Lista de figuras

Elementos Identificativos

Aluno:

Andreia Filipa Brazete Esteves
Nº1011854
TESP: Testes de *Software*
Email: andreiah.estevex@gmail.com

Estabelecimento de Ensino:

Escola Superior de Tecnologia e Gestão
Instituto Politécnico da Guarda

Duração de estágio:

Início: 01 março, 2016
Fim: 30 junho, 2016

Instituição Acolhedora do Estágio:

Nome: Altran, Centro de Negócios e Serviços,
Praça Amália Rodrigues 6230-350 Fundão
Telefone: 210 331 673

Orientador de Estágio:

Nome: Eng.º. Pedro Manuel Teixeira Pinto
Grau académico: Mestre em Computação Móvel

Orientador de Estágio Altran**:**

Nome: Hugo Firmino
Cargo: Gestor de Projeto Daikin ITC

Agradecimentos

Em primeiro lugar gostaria de deixar o meu reconhecimento ao Instituto Politécnico da Guarda, e um especial agradecimento ao meu Coordenador de curso, Profº José Fonseca e Dr.ª Clara Silveira, pelo seu profissionalismo e por terem sido elementos essenciais no meu percurso académico. Também ao meu orientador de estágio, Profº Pedro Pinto, gostaria de deixar um enorme agradecimento pela sua simpatia e capacidade de orientação.

Agradecer o apoio incondicional da minha família, em especial da minha mãe, padrasto, dos meus irmãos e também do meu namorado.

Aos meus colegas de curso, Gonçalo Amaral e Márcia Reverendo, um carinhoso agradecimento pelo espírito de entre ajuda, no que diz respeito á consolidação de conceitos, e pelo apoio na minha integração na Altran PT.

Para finalizar, a toda a equipa do projeto Daikin ITC, onde estive integrada neste período de estágio, porque foi neste saudável espírito de equipa que me foi possível adquirir todos os conhecimentos. Um especial agradecimento aos meus colegas André Matos, Cristiana Vieira e Eduardo Marcelo, e aos meus gestores de projeto Hugo Firmino e André Freitas.

E principalmente à Altran por toda a sua colaboração para que o tudo o processo de estágio corresse bem e pela oportunidade que me deu ao tornar possível este estágio.

A todos o meu sincero obrigado.

CAPÍTULO I- INTRODUÇÃO

1.1. Enquadramento

No âmbito do plano curricular do Curso Técnico Superior de Testes de *Software*, realizou - se um estágio curricular que decorreu na empresa Altran Fundão.
Fui integrada no projeto Daikin iTC, que tem como objetivo a implementação de um produto na *cloud* para a monitorização e controlo de dispositivos de climatização da marca.

1.2. Objetivos

De seguida indica-se os objetivos gerais deste estágio, onde se pretendia adquirir novos conhecimentos teóricos e práticos, mas também dar uso aos conhecimentos obtidos durante o ano letivo, e também os objetivos específicos, sendo estes:

- Integração na equipa de projeto;
- Conhecer os objetivos do projeto;
- Conhecer os objetivos da equipa de *testers* no projeto;
- Criação de planos de testes funcionais (manuais) de acordo com as metodologias Altran;
- Criação de relatórios de testes funcionais de acordo com as metodologias adotadas pela Altran;
- Comunicação de defeitos (erros) de forma estruturada e concisa;
- Participação na criação de testes automáticos, utilizando a framework *SpecFlow* e a ferramenta *Visual Studio* 2015.

1.3. Empresa Altran

O estágio decorreu na empresa Altran, no Centro de Negócios do Fundão, que é hoje uma das principais empresas na Consultoria de Inovação e Tecnológica em Portugal. Está presente em vários setores de atividade como o Financeiro, Telecomunicações & Media, Administração Pública, Indústria, *Energy & Life Sciences*, *Intelligent Systems e Utilities*, sendo que a nossa atividade tem uma estrutura assente na venda de soluções inovadoras. [1]

Com um modelo de negócio diferenciado, oferta está estruturada em quatro linhas de negócio:

- *Intelligent Systems*
- *Information Systems*
- *Lifecycle Experience*
- *Mechanical Engineering*

A Altran opera em mais de 20 países. Como parceiro estratégico, a Altran apoia os projetos globais dos seus clientes, mantendo a sua dimensão regional através das várias divisões geográficas que compõem o Grupo.

FIGURA 1- PRESENÇA DA ALTRAN NO MUNDO [1]

CAPÍTULO II - CONTEXTO DO PROJETO

2.1. Cliente: Daikin

A Daikin Europe N.V é uma empresa que oferece soluções de climatização inovadoras e da mais elevada qualidade e eficiência energética para corresponder às necessidades em mudança dos clientes ao nível residencial, comercial e dos serviços, mas também soluções de elevada eficiência energética e fiabilidade, para o controlo de temperatura em processos industriais.

FIGURA 2- DAIKIN LOGÓTTIPO

2.2. Produto: Daikin iTC

2.2.1. Objetivo

O projeto Daikin ITC (Intelligent Tablet Controller) consiste num produto implementado na *Cloud* que tem como objetivo monitorizar e controlar dispositivos de climatização da marca. [8]

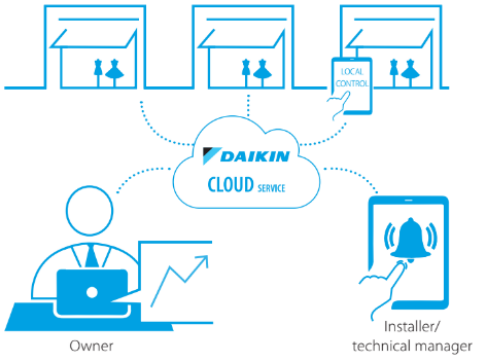

FIGURA 3- ESQUEMA DAIKIN ITC *CLOUD*[8]

2.2.2. Metodologia do projeto

Para a gestão e planeamento deste projeto foi utilizado a metodologia *agile* onde a alteração de requisitos é aceite, é fornecido frequentemente software funcional e a equipa de desenvolvimento e o cliente trabalham juntos, neste caso usou-se uma das técnicas denominada por *Scrum* que consiste numa organização especifica.

O projeto é divido em ciclos (semanais) chamados de *sprints* que são conjuntos de funcionalidades que devem ser desenvolvidas/executadas.

As funcionalidades a serem implementadas no projeto são permanecidas numa lista que chamada *Product Backlog.*

No início de cada *Sprint*, faz-se uma reunião de planeamento um (*Sprint Planning Meeting*) na qual o *Product Owner* (cliente) prioriza os itens do *Product Backlog* e a equipa seleciona as atividades que ela será capaz de implementar durante o Sprint que se inicia.

As tarefas alocadas numa *Sprint* são transferidas do *Product Backlog* para o Sprint *Backlog*.

A cada dia de uma *Sprint*, a equipa faz uma breve reunião (normalmente de manhã), chamada *Daily Scrum*. O objetivo é partilhar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho do dia que se inicia. [10]

2.2.3. Colaboração e equipa

Na generalidade, cada equipa é associada a um determinado projeto, e que por sua vez são alocadas a salas individuais ou a *openspaces*, que é o caso da equipa a qual integrei.

Neste projeto a equipa é mista, sendo qua há *developers* e *testers* no mesmo espaço e há uma colaboração com uma equipa da Altran Bélgica. [8]

A constituição da equipa é a seguinte:

- *Testers (PT)*
- *Web developers (PT)*
- *.Net Developer (PT)*
- *Scrum Master (PT)*
- *Project Manager (PT)*
- *Senior Project Manager PT,*
- *Solutions Architect (PT)*
- *UI/UX Designer (PT)*
- *Proxy Product Owner (BE)*
- *Product Owner (BE)*
- *API Architect (BE)*
- *API expert (BE)*

2.2.4. Funcionalidades do produto

Ao longo dos vários *sprints* foram desenvolvidas as funcionalidades definidas no início do projeto, tanto a nível de administração como de segurança e monitorização de equipamentos, etc.

- Personalização do perfil de utilizador
- Gestão de utilizadores;
- Disponibilização de gráficos com dados estatísticos
- Notificações de erros de *cloud*;
- Configuração dos equipamentos (temperatura, modo de ventilação, velocidade, etc);
- Cálculo do *SCOP*;
- Gestão de lojas onde estão implementados os equipamentos;
- Definição de horários para monitorização automática dos equipamentos;
- Visualização das lojas no mapa;
- Organizar os equipamentos por zonas/pisos. [8]

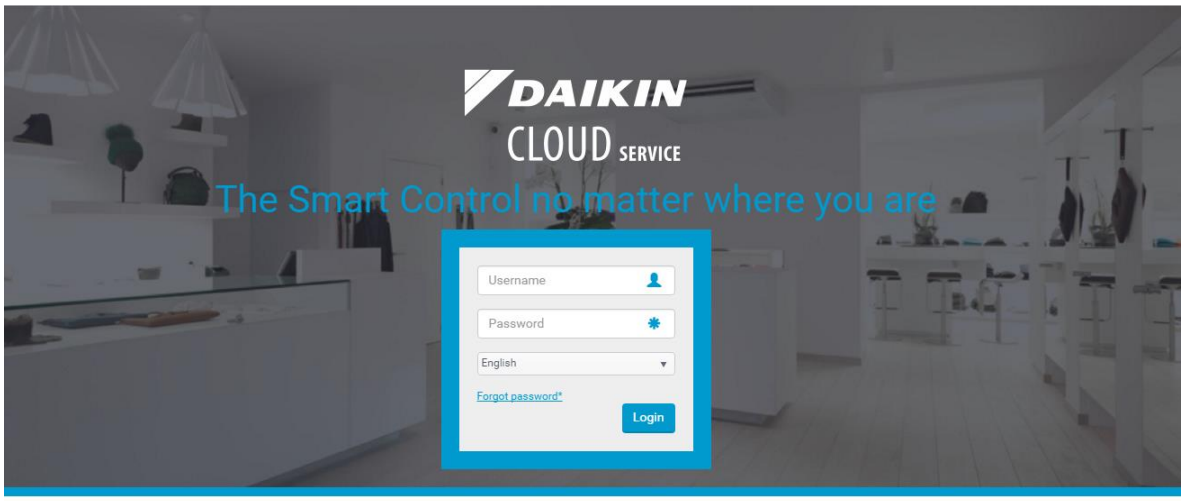

FIGURA 4- PÁGINA LOGIN DA DAIKIN ITC**[8]**

2.2.5. Processo de trabalho

Ao longo de uma sprint há diferentes etapas e dentro delas existem várias tarefas, tanto para os elementos da equipa que desenvolvem e os que testam. As principais etapas são :

a) TO DO

- Analisar a história do utilizador descrição / aceitação;
- Ver se as informações relativas às funcionalidades, pois não é suficiente para iniciar o desenvolvimento
- Elaboração dos critérios de aceitação.
- Definir a solução técnica para assegurar a entrega do produto
- Atualizar os estados da história conforme se estiver em progresso ou não.
- Atribuição de pontos às histórias (funcionalidades) conforme a complexidade e a importância no projeto.

b) IN PROGRESS

- Desenvolvimento das funcionalidades;
- Testes Unitários;
- Elaboração dos *TestCases*;
- Preparação do ambiente de testes.

c) TESTS

- Testes de sistema;
- Testes de Integração;
- Testes manuais;
- Testes de exploração;
- Registo de *bugs* caso sejam encontrados.

d) VERIFYING

- Caso seja encontrado um *bug* é registado e é necessário proceder à sua resolução;
- Os *developers* têm de rever a funcionalidade caso seja encontrado *bug* nela;
- Os *testers* têm a responsabilidade de assegurar que é tudo testado.

e) IN REVIEW

- O *Product* Owner executa os testes que passaram e verifica que estão as funcionalidades estão de acordo com os critérios de aceitação.

f) DONE

- Finalização do processo da história, tendo em conta que está desenvolvida e testada sem *bug*s.

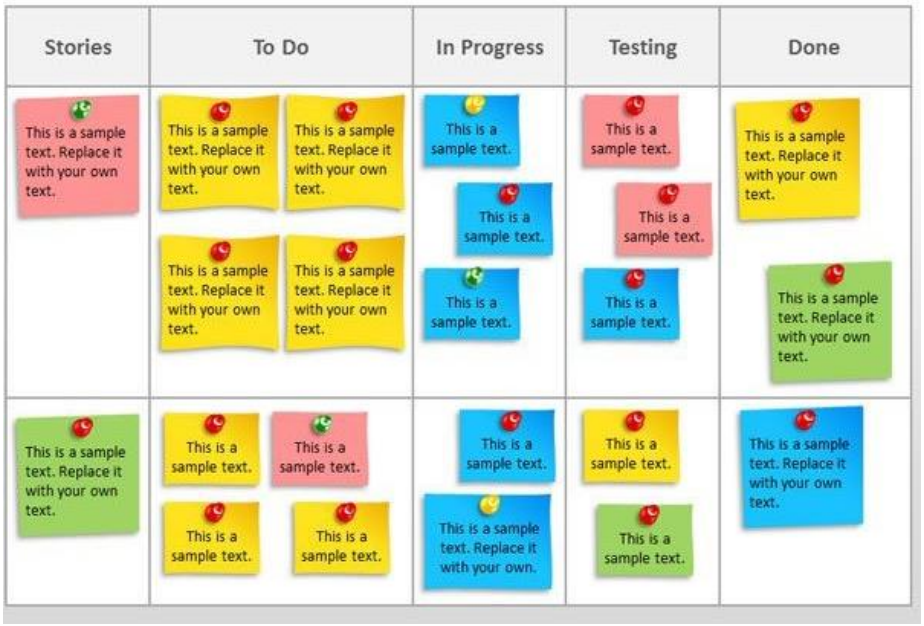

FIGURA 5-QUADRO DO WORKFLOW [8]

CAPÍTULO III- TESTES DE *SOFTWARE*

3.1. Introdução

Cada vez mais o *software* está presente no nosso quotidiano, nas mais diferentes áreas. Nesse sentido está a tornar-se mais complexo devido ao crescimento das novas tecnologias. Muitas vezes um produto de software apresenta falhas no seu funcionamento que colocam em causa a sua reputação e assim sendo não inspiram confiança aos utilizadores.

Existem vários exemplos que geraram problemas na fase de produção, trazendo custos altos, má reputação nos negócios e até mesmo colocou em risco a vida de seres humanos.

Para evitar problemas como os citados acima é necessário investir em testes, pois, testes em sistemas e em documentações reduzem os riscos da ocorrência de defeitos do *software* no ambiente de produção onde são encontrados pelo cliente, contribuindo assim para a qualidade dos sistemas, pois, quanto mais cedo os defeitos forem encontrados antes da implantação do sistema o custo de correção é menor em relação ao encontrado da fase de produção. E por isso os testes podem se dividir em vários géneros para uma determinada especificação. [5]

3.2. Tipos testes de *software*

Testes caixa branca e testes caixa preta. Os testes caixa preta são realizados sem conhecimento da operação interna (código) do s*oftware*, é feito analisando o resultado obtido, geralmente tomando por base casos de uso e especificações de requisitos.

Já os testes caixa branca são feitos com conhecimento da operação interna, onde o tester pode desenvolver códigos para realizar os testes de todas as aplicações de cada componente.

3.2.2. Testes Funcionais

Também considerados como testes manuais, estes foram o tipo de testes que realizados ao longo do estágio. São definidos os *Test Cases* num *TestPlan* e estes são executados manualmente, registando o resultado no respetivo documento. É o único tipo de teste que não é automatizado, mas, contudo, tem suas vantagens pois são encontrados *bug*s que eventualmente utilizando uma ferramenta de automação de testes não consegue detetar, como por exemplo o design do *software* se está de acordo com as especificações do projeto.

3.2.3. Testes Unitários

São aqueles que normalmente são implementados pelos *developers* para examinar o código sob as diferentes condições. Em termos de linguagem orientada a objeto (OO), estas unidades de trabalho são métodos de uma classe. Previnem regressão, dão uma maior cobertura e podem servir como documentação.

3.2.4. Testes de integração

É nesta fase do teste de *software* em que módulos são combinados e testados em grupo. É verificar os requisitos funcionais, de desempenho e de confiabilidade na modelagem do sistema. Com ele é possível descobrir erros de interface entre os componentes do sistema.

3.2.5. Testes de sistema

É executado o sistema sob ponto de vista de seu utilizador final, executando as funcionalidades e procurar falhas em relação aos objetivos originais.
Estes tipos de testes podem ser executados automaticamente com o auxílio de ferramentas para garantir a qualidade do produto final. [7]

4.3.6. Testes de aceitação

Estes tipos de testes baseiam-se na análise específica de uma funcionalidade de um componente ou sistema, mais concretamente numa de uma especificação contida na *user storie* e critérios de aceitação. Este nível de testes é realizado no *back-end.*

4.3.7. Testes de performance

Antes de cada *release,* estes tipos de testes podem ser executados para validar os requisitos de desempenho estão a ser cumpridos. Com a ferramenta indicada permite simular o trafego, a carga no sistema e o tempo de resposta.

FIGURA 6-CONCEITOS TESTES DE SOFTWARE [9]

CAPÍTULO IV- TRABALHO DESENVOLVIDO

4.1. Desenvolvimento de *TestPlan*

Para executar os testes manuais foi elaborado vários planos de teste (anexo 1) utilizando a ferramenta da Microsoft, seguintes sempre uma estrutura específica para que a informação fique concisa e percetível. Tendo os seguintes campos:

***ID_WITCA*:** Identificação da *user story*
TEST CASE: Caso de teste que quero executar
***PRECONDITIONS*:** Pré-requisitos que tenho de ter antes da execução
STEPS: Descrição dos passos para a execução
***EXPECT RESULT*:** resultado que é esperado depois de executar o *test case*
***RESULTS*:** Resultados finais dos testes nos diversos browsers.

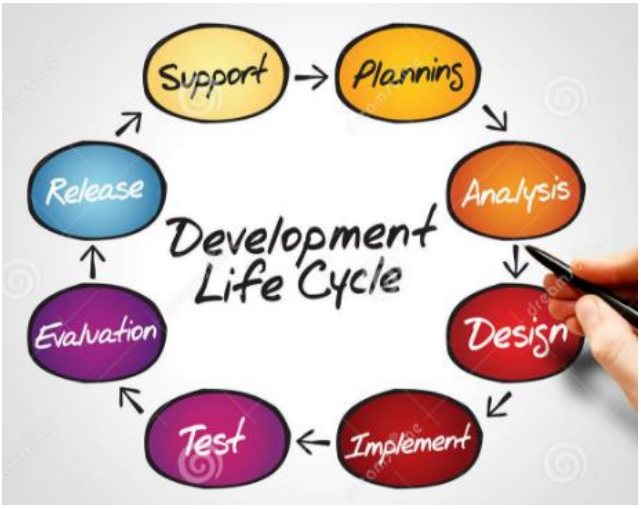

FIGURA 7- CICLO DE VIDA DE UM SOFTWARE [7]

4.2. Ferramentas do projeto

4.2.1. Jenkins

É um dos *softwares* utilizados quando se fala em "Integração Contínua", que é o responsável pelo desenvolvimento *agile*, ou seja, a facilidade em deteção de *bug*s em códigos e o rápido *deploy*, em caso de falha por parte de um ou de um time de *developers*. Cuja as vantagens são as seguintes:

- ✓ *Builds* periódicos
- ✓ Testes Automatizados
- ✓ *Builds* em ambientes diferentes do *developers*
- ✓ Possibilita análise de código
- ✓ Reduzir retrabalho
- ✓ Reduzir custo da entrega
- ✓ Identificar erros mais cedo
- ✓ Fácil de operar e configurar
- ✓ Interface agradável
- ✓ Integração com outras ferramentas através de *plugins* existentes na própria aplicação

.

.

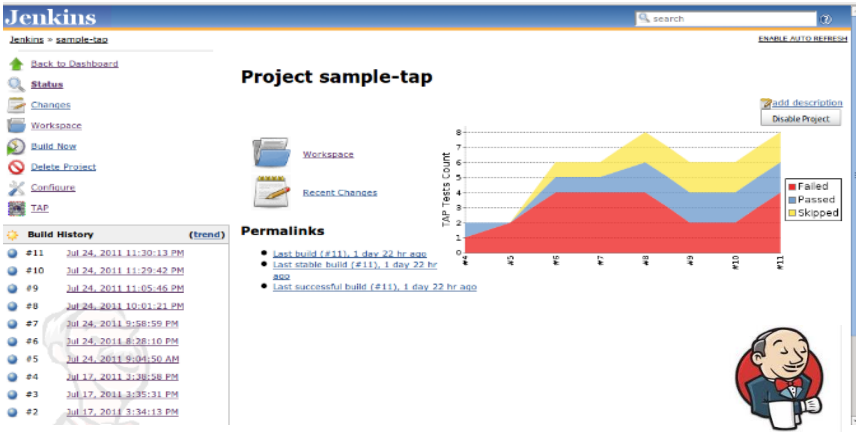

FIGURA 8- DESIGN DA FERRAMENTA JENKINS [4]

4.2.2. *Swagger*

Com a especificação de recursos declarativa de *Swagger*, os clientes podem entender e consumir serviços sem o conhecimento da implementação do servidor ou o acesso ao código do servidor. O quadro *Swagger* UI permite que os *developers* e *testers* consigam interagir com a *API* através de uma caixa de areia que dá uma visão clara de como a *API* responde a parâmetros e opções. Permite fazer a modelagem da *API*, geração de documentação da mesma e de códigos do cliente e do servidor com suporte a várias linguagens de programação

Esta foi uma das ferramentas que mais se utilizou para fazer testes de integração manuais.

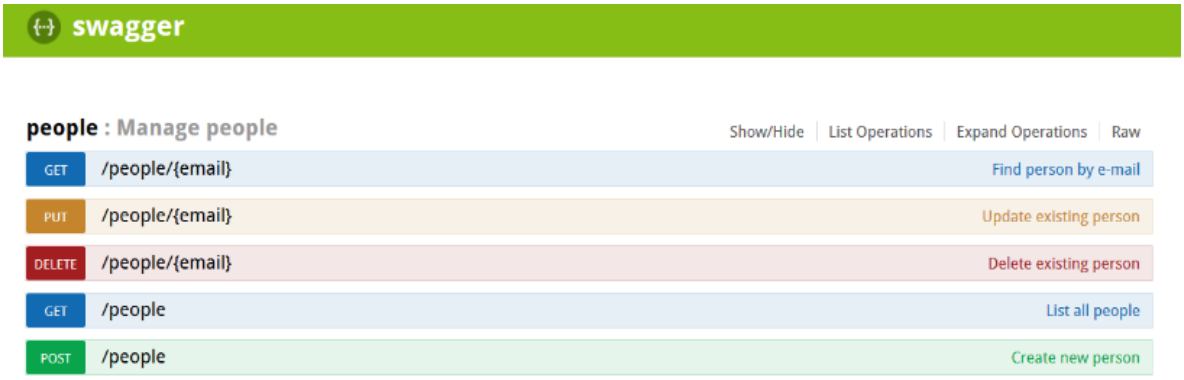

FIGURA 10-LISTA DAS *API* NO *SWAGGER*

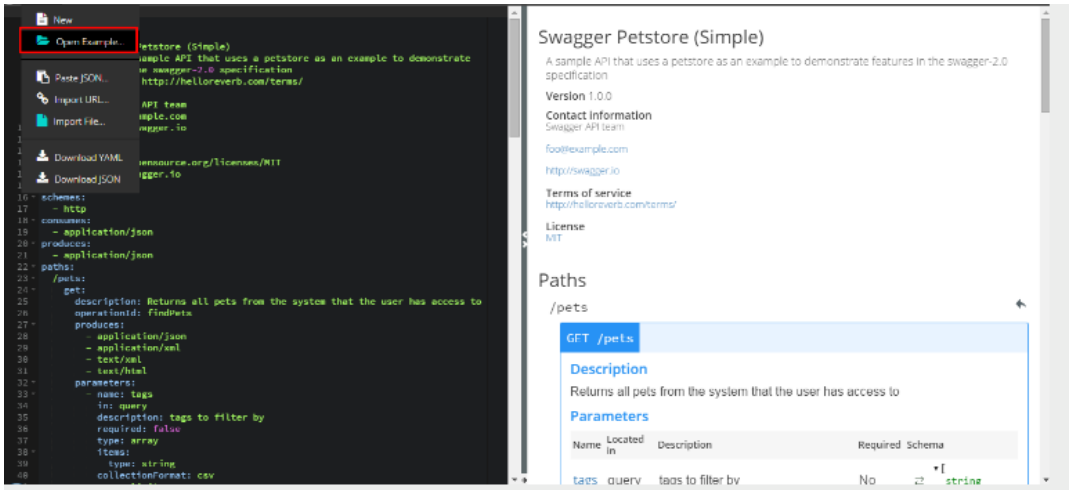

FIGURA 9- AMBIENTE DE DESENVOLVIMENTO NO *SWAGGER*

4.2.3. *Visual Studio*

A plataforma *Microsoft Visual Studio* é um pacote de programas da Microsoft para desenvolvimento de *software* especialmente dedicado ao *.NET Framework* e às linguagens *Visual Basic* ,C, C++, C# (C *Sharp*) e J# (J *Sharp*). Também é um grande produto de desenvolvimento na área web, usando a plataforma do *ASP.NET*. As linguagens com maior frequência nessa plataforma são: *VB.NET* (*Visual Basic*.Net) e o C♯ (lê-se C *Sharp*).

Em relação a esta ferramenta foi efetuada a configuração para o ambiente de testes de sistema, instalando um conjunto de extensões nomeadamente o *SpecFlow*, *Selenium* e *Nunit* que são importantes para a execução de testes. [3]

Vantagens para os *developers*:

- Construir personalização para *SharePoint*;
- Desenvolver aplicações para *Windows* 7, utilizando seus recursos (*Multitouch*, *Pin Bar*, etc.);
- Compreender de forma mais simples códigos e arquitetura já existentes;
- Identificar por testes, os impactos de alterações no código.
- Configurar o *Visual Studio* para atender às suas necessidades.

Vantagens para os *testers*:

- Aproveite uma colaboração profunda com a equipa de desenvolvimento;
- Avançar (*Fast Forward*) por testes manuais;
- Reproduzir *bug*s em um ambiente virtualizado;
- Automaticamente incluir contexto aos *bug*s;
- Total visibilidade do progresso do teste.

4.2.4. *Jira*

É um dos *softwares* da *Atlasian*, e essa é uma das principais vantagens do *Jira* sobre seus concorrentes, a mesma empresa oferece vários outros *softwares* e extensões para ajudar no dia-a-dia de maneira integrada, mais propriamente sobre gestão de tarefas, tem como ponto forte a gestão da equipa, projetos e funções, facilita muito a visualização do uso de recursos, consegue se ter um feedback e acompanhar as tarefas em lista do backlog.

Esta foi das principais ferramentas que foi utilizada para auxiliar na execução dos testes e na orientação do estado de cada user *story*, tendo outras também outras principais funcionalidades, tais como:

- ✓ Registos de *Bug*s;
- ✓ Estado do sprint;
- ✓ Visualização das tarefas atribuídas para cada elemento da equipa;
- ✓ Assinação das tarefas ao respetivo elemento de equipa;

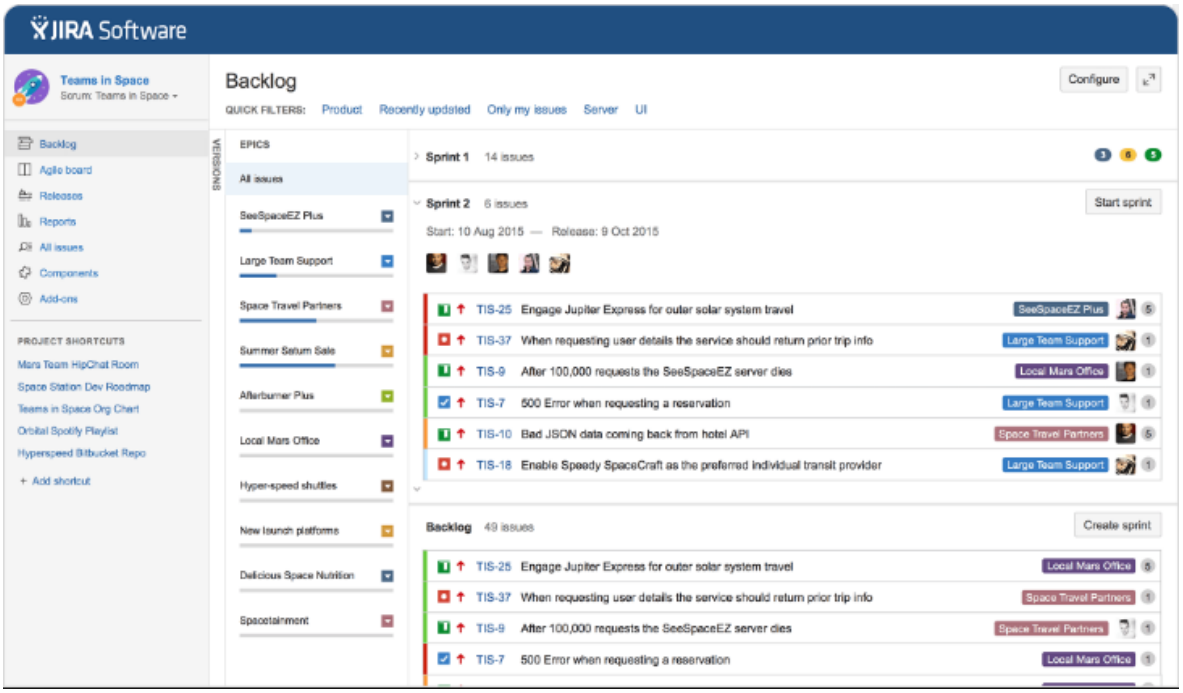

FIGURA 11- DESIGN DA FERRAMENTA JIRA [2]

4.3.3. Browser *tools*

Esta ferramenta permite uma análise do de todos os elementos carregados pela pagina web desde *Html, Css, JavaScrip*, permitindo a sua edição diretamente no browser.

Permite também fazer uma analise de performance da página. Analisando o tempo que cada elemento leva até ser carregado e outras funcionalidades tal como ver os comandos e os conteúdos enviados para o servidor (comandos HTTP).

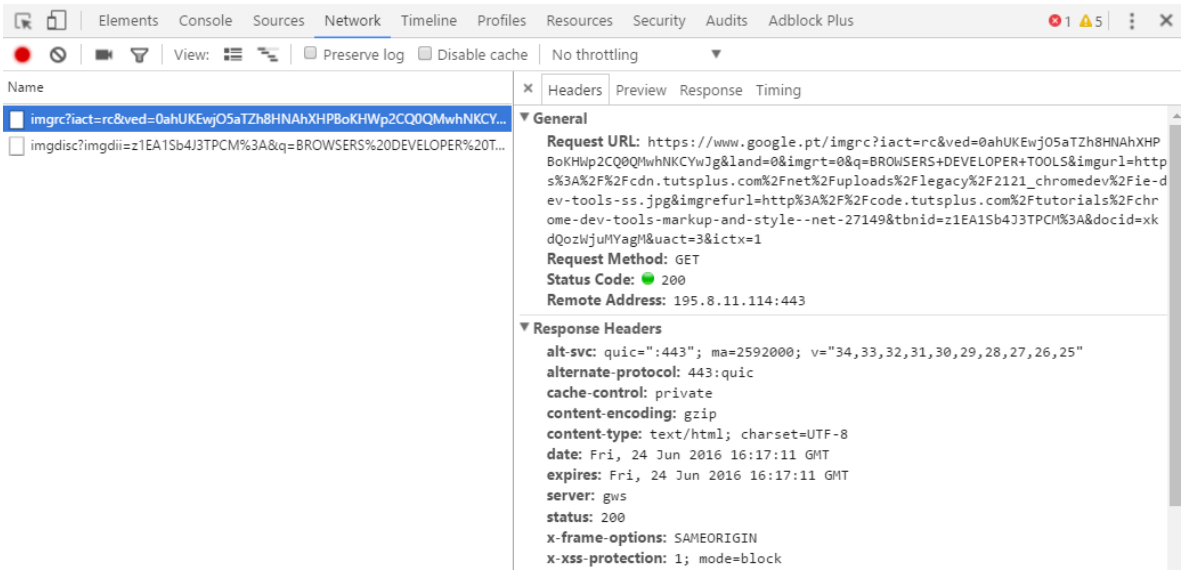

FIGURA 13- NETWORK

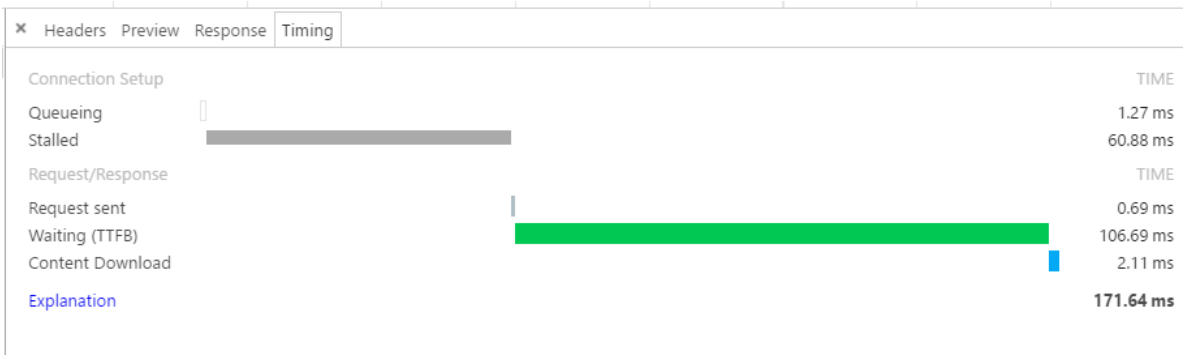

FIGURA 12-TIMING

4.3. Gestão de *Bugs*

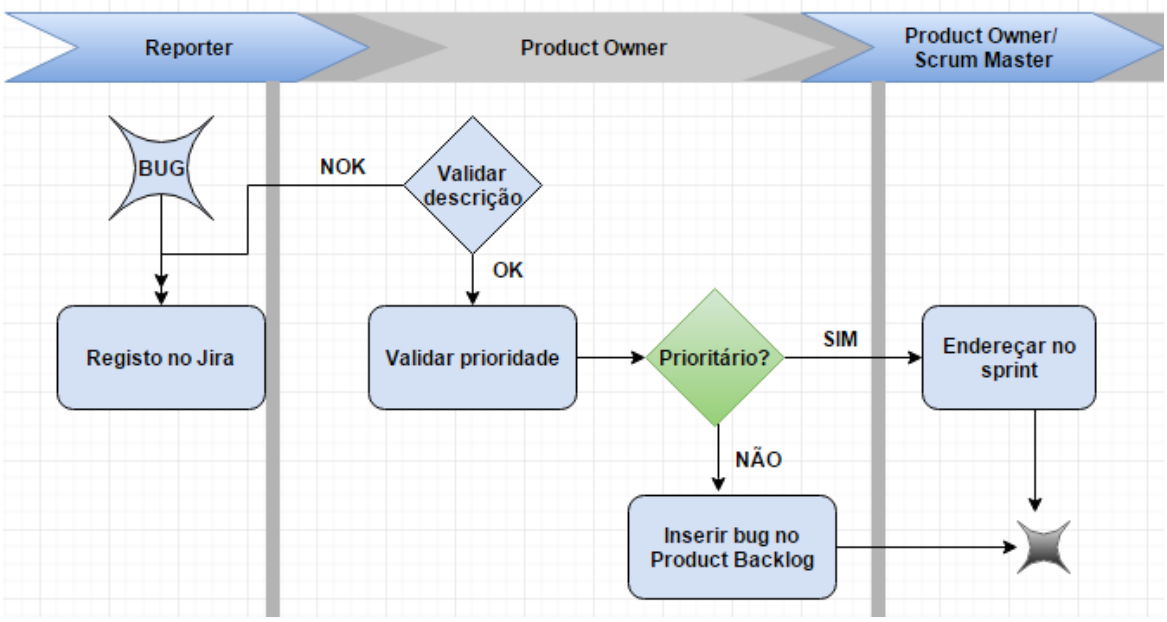

FIGURA 14- ESQUEMA FLUXO DE UM BUG

Durante a execução dos testes, foram encontradas várias funcionalidades que não estavam a ser executadas corretamente, os *bug*s quando são problemas mais críticos ou também os chamados de *issue*s que são comportamentos inesperados e por isso é sempre importante reportá-los para um lugar específico que neste caso foi utilizando a plataforma *Jira* para que fique estruturada e concisa.

O repórter precisa de atualizar a descrição do *bug* no *Jira* com a seguinte informação:

✓ Definir o *issue*;

✓ Definir na especificação o nível de prioridade, a razão e o valor de atribuição;

✓ Sequencia dos passos para reproduzir o *issue*;

4.3.1. Identificação de níveis de prioridade:

Valor	Descrição
10	Perda/ corrupção de dados ou sistema indisponível
8	Funcionalidade importante não está disponível e não tem solução
7	Funcionalidade importante indisponível, mas há uma solução para resolver
5	Funcionalidade menos relevante indisponível, mas há uma solução para resolver
2	Erros de interface ou funcionalidade com uma fácil resolução.

TABELA 1- NÍVEL DE SEVERIDADE

Valor	Descrição
10	Possibilidade forte de repetição do erro (ex: diariamente)
5	Possibilidade Moderada possibilidade de repetição do erro (ex: uma vez por semana)
3	Funcionalidade importante indisponível, mas há uma solução para resolver (ex: uma vez por mês)
2	Improbabilidade de acontecer conforme a operação

TABELA 2-NÍVEL DE OCORRÊNCIA

Valor	Descrição
10	Não é possível fazer recuperação
8	Interação mais específica (ex: modificação na base de dados)
5	Interação pequena dada pelo utilizador (ex: F5)
2	Recuperação automática
1	Não é necessária recuperação

TABELA 3-NÍVEL DE RECUPERAÇÃO

Valor	Descrição
>= 320 ***Must***	*Critical* Avaliação do sprint *backlog* com a equipa e *product owner*
100 <>200 ***Should***	*Serious* Tem de ser inserido no próximo sprint
10 <>= 100 ***Could***	*Moderate* Devem ser agendados para os próximos dois ou três *sprints*
<=10 ***Won't***	*Low priority* É resolvido quando houver tempo

TABELA 4-ESCALA DE BUGS

Após a análise em relação à severidade, ocorrência e recuperação é necessário aplicar a seguinte fórmula: **Impacto= severidade x ocorrência x recuperação.**

O resultado final será um valor entre 4 e 1000. E depois sim é analisado o grau de prioridade do *issue* encontrado.

Esta avaliação é muito importante num ciclo de vida de um software, porque se um bug for mal reportado irá haver muitos riscos no projeto e o importante é que os mesmos sejam feitos com qualidade e atenção. Quando um bug é reportado com qualidade, o trabalho de quem precisa receber essa informação fica mais *agile* e tende a facilitar o processo de validação daquela suposta falha no software.

CAPÍTULO V- EVENTOS E FORMAÇÃO

5.1. "Induction Day"

No primeiro dia do estágio houve uma formação geral e teve o se*gui*nte programa com os seguintes objetivos de conhecer o Mundo Altran, a sua cultura, os seus valores, o que fazemos, como crescemos e progredimos na carreira, etc.

Esta sessão enquadrou-se no âmbito do Programa Action4, um programa Altran que define os objetivos estratégicos a longo prazo baseados em 4 grandes eixos – *Employeee Differentiation, Growth Engine, Profitability Engine e Customer Differentiation.*

Agenda:

Abertura
Grupo Altran (história, cultura, valores, linhas de oferta)
Estratégia Action4
Altran em Portugal
Comunicação interna
Ser Consultor (testemunhos)
Almoço (a convite da Altran)
Programas Altran
Conduta Altran
Quiz´Altran
Papel do consultor na vida da empresa
FAQ´s

5.2. Formação IEFP-Francês Básico

Esta formação teve uma duração de 50 horas, onde foram abordados vários conceitos nomeadamente muitos conhecimentos por isso foi uma mais valia para a minha formação linguística visto que a Altran é uma empresa francesa e é sempre positivo ter versatilidade nas línguas, tanto a nível pessoal e de aprendizagem.

5.3. Evento: *Ignite your future*

Participação no evento tecnológico *"Ignite Your Future"*, organizado pela Altran, Câmara Municipal do Fundão e Universidade da Beira Interior (UBI). Foi direcionado para alunos do 9º ao 12º de diversas escolas do país, que foram divididos por equipas e tinham de participar nos diversos exercícios.

Fiquei inserida num dos exercícios promovidos pela Altran, no qual o principal objetivo era construir uma cidade com peças de Lego de acordo com os requisitos do cliente, fazer o planeamento e conse*gui*r fazer a gestão das tarefas pelas 3 equipas.

FIGURA 15-CARTAZ EVENTO

GLOSSÁRIO DE CONCEITOS

Defeito: resultado de um erro encontrado num código ou num documento.

Erro: engano cometido por seres humanos.

Falha: resultado ou manifestação de um ou mais defeitos.

Testware: define toda a documentação de teste.

Test case: uma descrição de um teste a ser executado. Um ou mais casos de teste costumam estar relacionados a um caso de uso.

Test Suite: pacote de casos de teste relacionados. Por exemplo: Suíte de cadastro, suíte de consulta.

***Test Plan*:** documento de planeamento do projeto de teste é equivalente ao Plano de Projeto definido pelo PMI (*Project Management Institute*).

Script de Teste: automação da execução de um caso de teste.

Ambiente de Testes: ambiente de testes deve ser definido pelo nível de testes a ser executado, ou seja, quanto maior o nível, mas, o ambiente de teste deverá ser capaz de reproduzir as características do ambiente de produção.
Front-end: é toda a parte da apresentação visual de um site.

***Back- end*:** código do lado servidor.

***GUI* :** tipo de interface do utilizador que permite a interação com dispositivos digitais por meio de elementos gráficos como ícones e outros indicadores visuais.

***API*:** contexto de desenvolvimento Web, uma *API* é um conjunto definido de mensagens de requisição e resposta *HTTP*, geralmente expresso nos formatos XML ou JSON.

***Workflow*:** sequência de passos necessários para se automatizar processos de negócio, de acordo com um conjunto de regras definidas.

SpecFlow: extensão para o *Visual Studio* usa esta estrutura e vai gerar um código (*gherkin*).

***Gherkin*:** forma estruturada de descrever cenários reais de uso e permite a geração de testes automatizados de aceitação.

.NET Framework: ambiente de execução de gestão que fornece uma variedade de serviços para os aplicativos em execução.

Branchs: cópia do código, pudendo fazer alterações para testar sem interferir sem alterar o código original pudendo fazer testes.

SVN: controlador de versões.

***Comit*:** atualização no servidor das alterações.

CONCLUSÃO

O estágio realizado na Altran foi muito enriquecedor pois permitiu a consolidação dos conhecimentos obtidos na fase das aulas teóricas como também contribui para desenvolvimento de novas competências no qual me proporcionou uma evolução de aprendizagem.

Com esta experiência foi enriquecedora no seu âmbito testes de software pelo qual me motivou a querer evoluir mais nesta área tal como automatização de testes.

Os objetivos gerais do meu estágio foram concretizados e em termos específicos devo salientar que a nível de criação de testes automáticos *framework SpecFlow* e a ferramenta *Visual Studio* 2015, não me foi possível concluir com sucesso, contudo pude concluir a mesma configuração do ambiente.

A elaboração de *test plans*, reportação de *bugs*, utilização da ferramenta *Jira* para auxilio de orientação, análise de documentação foram as principais atividades que efetuei durante este período de estágio. Também a língua inglesa esteve bastante presente visto que o projeto tinha uma colaboração com uma equipa da Altran Bélgica e tanto a nível de reuniões e elaboração de documentos foi tudo baseado nesta língua.

Relativamente a dificuldades devo referir a dificuldade natural de adaptação do projeto que apenas no meu ver estiveram presentes no inicio, em relação à adaptação no projeto e à sua metodologia, de todo o modo consegui ultrapassar essa dificuldade e permitiu me concretizar as tarefas que foram incumbidas.

Em suma, este estágio curricular proporcionou me um leque variado de experiências, revelando-se uma fonte inestimável de conhecimentos a nível pessoal e profissional e também alargando a minha visão sobre testes de *software*.

BIBLIOGRAFIA

[1] Altran, “Altran,Líder em Tecnologia e Inovação,” [Online]. Available: http://www.Altran.pt/.

[2] A. C. A. Almeida, “Aspiração manual de trombos,” [Online]. Available: http://repositorio.ipl.pt/bitstream/10400.21/1592/2/Aspira%C3%A7%C3%A3o%20manual%20de%20trombos%20no%20EAMST2.pdf. [Acedido em 24 junho 2016].

[3] “Trabalhando com Testes Manuais,” [Online]. Available: https://msdn.microsoft.com/pt-br/library/ms182615(v=vs.90).aspx. [Acedido em 20 junho 2016].

[4] Daikin, “Project Management Plan,” 2015.

[5] Matera systems, [Online]. Available: http://www.matera.com/br/.

[6] “Jenkins- Integração Contínua, Cursos, Target Trust,” [Online]. Available: http://www.targettrust.com.br/curso/jenkins-integracao-continua.

[7] “Ferramentas Gratuitas para Desenvolvedores- Visual Studio Community 2015,” [Online]. Available: https://www.visualstudio.com/pt-br/products/visual-studio-community-vs.aspx. [Acedido em 2016].

[8] “Artigo Engenharia de Software- Introdução a Teste de Software,” [Online]. Available: http://www.devmedia.com.br/artigo-engenharia-de-software-introducao-a-teste-de-software/8035. [Acedido em 24 junho 2016].

[9] renttesters.com. [Online]. Available: http://renttesters.com/wp-content/uploads/2014/12/16084033-abstract-word-*cloud*-for-software-testing-with-related-tags-and-terms.jpg.

[10] J. Gregory, Agile Testing: A Practical Guide for Testers and Agile Teams, 2010.

(

]

ANEXOS

ANEXO 1- Exemplo de um Test Plan

	Test Cases					Results			
WITCA	#	Test case	Preconditions	Steps	Expected result	Chrome	Firefox	IE11	
WITCA-11	1	As an user (any role) I want to associate Management Points to an Area so that they can be controlled/monitored through the area		Scenario: Add a management point to a zone Given I am an user (any role) And a site is assigned to me And I have a zone already created When I drag and drop a management point into the created zone Then the zone has this management point associated		Y 09/06 11:12	Y 16:17 09/06	Y 15:20 09/06	
WITCA-11	2	As an user (any role) I want to associate Management Points to an Area so that they can be controlled/monitored through the area		Scenario: Remove a management point from a zone Given I am an user (any role) And a site is assigned to me And I have a zone already created And I drag and drop a management point into the created zone When I drag and drop a management point out of the zone Then the zone does not have this management point associated		Y 09/06 11:12	Y 16:22 09/06	Y 15:30 09/06	
WITCA-11	3	As an user (any role) I want to associate Management Points to an Area so that they can be controlled/monitored through the area		Scenario: Remove a zone with associated management points Given I am an user (any role) And a site is assigned to me And I have a zone already created And I drag and drop a management point into the created zone When I delete the created zone Then the management point is assigned to the top level zone		Y 09/06 11:12	Y 16:24 09/06	Y 15:40 09/06	
WITCA-11	4	When there is no zones the MP's are allocated to the top level	1- Logged in as an user (any role) and site associated with MP 2- The user is in Building Administration page 3- Delete all zones	- Go to Equipment List and count the MP's - Go to Zones page - Check that the Management Point's on the top level are the same on the equipament list	All the MPs are allocated to the top level	Y 09/06 15:12	Y 16:25 09/06	Y 16:00 09/06	
WITCA-11	5	Check that one MP is present only in one zone	1- Logged in as an user (any role) and site associated with MP 2-The user is in Building Administration page 3- Delete all zones	- Go to Building Administration - Go to Zones page - Open a created parent area and associated him one Management Point	The MP associated is only present in new parent area	Y 09/06 15:48	Y 16:34 09/06	Y 16:15 09/06	