Mestrado em Engenharia Informática
Estágio
Relatório Final

Desenvolvimento de uma aplicação web para processamento de dados sensoriais móveis

Telmo José Santos Neves
tjsneves@student.dei.uc.pt

Orientador:
Prof. Bruno Miguel Brás Cabral

Co-orientador:
Prof. António Jorge da Costa Granjal

2 de Setembro de 2015

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Mestrado em Engenharia Informática
Estágio
Relatório Final

Desenvolvimento de uma aplicação web para processamento de dados sensoriais móveis

Telmo José Santos Neves
tjsneves@student.dei.uc.pt

Orientador:
Prof. Bruno Miguel Brás Cabral

Co-orientador:
Prof. António Jorge da Costa Granjal

Júri Arguente:
Prof. Jorge Miguel Sá Silva

Júri Vogal:
Prof. João Pedro Morais de Matos Moniz Ramos

2 de Setembro de 2015

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Conteúdo

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

O presente relatório descreve o trabalho desenvolvido pelo aluno Telmo José Santos Neves, no âmbito do da unidade curricular do Mestrado em Engenharia Informática do Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra, no ano lectivo de 2014/2015, integrado no projecto Ecomobile.

1.1 Enquadramento e Motivação

Na realização deste trabalho o estagiário estará integrado na equipa da Sentilant, que surgiu durante o projecto Ecomobile. Este projecto surge com a preocupação de melhorar a condução e aumentar a segurança rodoviária dos condutores, através da obtenção de dados sensoriais a partir de um *smartphone*, para monitorização da condução.

Numa fase inicial do projecto, venceu dois prémios de inovação tecnológica, em 2011 no concurso PT Galp Innovation Challenge e em 2012 no concurso Arrisca C, dando assim origem à empresa Sentilant [1]. Actualmente, a empresa é constituída por uma equipa de desenvolvimento dividida entre: desenvolvimento móvel (IOS e Android), desenvolvimento web e desenvolvimento de soluções de BI. Actualmente incubada no Instituto Pedro Nunes, este projecto deu origem a dois produtos: Drivian e DrivianPro.

O produto *Drivian*[1], desenvolvido na vertente *Business-to-Consumer* (B2C), tem uma componente social bastante forte, com o intuito de melhorar a segurança rodoviária dos condutores e produzir uma condução mais eficiente para consequentemente reduzir despesas de combustível e manutenção [38] [39]. Actualmente pretende-se expandir o projecto Ecomobile na

[1]Sentilant: http://www.sentilant.com/

vertente *Business-to-Business* (B2B) com o produto *DrivianPro*, destinado a empresas.

Numa empresa em que a gestão de veículos é fundamental, uma gestão de frotas automatizada tem uma importância extrema para o sucesso da organização. Para isso é necessário software para lidar com as diferentes componentes existentes com a finalidade de optimizar os recursos, aumentar a performance e reduzir os custos. Cada vez mais as empresas tem a necessidade deste software para controlar os funcionários e veículos, analisar desempenho, gerir tarefas, aumentar a segurança e produtividade. Apesar de existirem soluções especificas para gestão de frotas, gestão de pessoas, recursos e tarefas, não existe uma solução que reúna todas estas funcionalidades. Assim as empresas necessitam de adquirir múltiplos sistemas a diferentes fornecedores e pagar elevadas quantias pela sua integração.

O produto DrivianPro, vai permitir realizar simultaneamente uma gestão estratégica e operacional das respectivas frotas, tarefas, e outros recursos. A aplicação móvel vai recolher informação detalhada, através de uma monitorização constante, transmitindo dados como localização, produtividade e realização de tarefas a uma aplicação web que centraliza os dados dos diversos *smartphones*. Estes dados são processados e transformados em métricas relevantes que o utilizador pode verificar através de relatórios constantes, detalhados e intuitivos com o objectivo de aumentar a performance da organização e reduzir custos. Este produto pretende adaptar-se a qualquer empresa que necessite desta gestão de recursos, independentemente do serviço e a integração com diferentes fontes de dados externas.

1.2 Objectivos

O trabalho desenvolvido pelo estagiário está integrado nas actividades de desenvolvimento de uma plataforma altamente escalável para tratamento de informação proveniente de dispositivos móveis. Os objectivos planeados são:

- Desenho e implementação de uma *framework* para o *front-end* da aplicação capaz de interpretar meta-dados, de gerar interfaces web tais como listas, detalhes, grelhas e mapas, e realizar todas as operações envolvidas, que permitam manipulação, visualização e comparação da informação produzida.

- Desenho e implementação de uma plataforma capaz de realizar integração de dados provenientes de múltiplas fontes de informação, com a finalidade de melhorar os recursos de uma organização.

1.3 Estrutura do documento

Capítulo I – Introdução: secção que apresenta uma vista geral do estágio, incluindo contextualização, motivação, objectivos que se pretendem atingir e estrutura do documento.

Capítulo II – Metodologia : metodologia usada durante o estágio tal como o planeamento.

Capítulo III – Estado da Arte : são analisadas e comparadas as soluções existentes no mercado.

Capítulo IV – Análise de Requisitos : Apresentados os requisitos para a realização do estágio, e o processo de levantamento dos mesmos.

Capítulo V – Arquitectura: Nesta secção vai ser detalhada a arquitectura do projecto. Conjuntamente será apresentado as tecnologias usadas durante o desenvolvimento do projecto bem como as decisões arquitecturais para atingir os objectivos.

Capítulo VI – Implementação : Neste capitulo é necessário separar por duas vertentes: a secção com a informação sobre a plataforma de *front end* e a secção com a integração de dados em que se explica o desenvolvimento como se encontra na arquitectura. Em ambos é detalhado o trabalho desenvolvido tal como os resultados obtidos.

Capítulo VII – Conclusão : Consiste numa reflexão sobre o trabalho desenvolvido, bem como o trabalho futuro a desenvolver e as contribuições para o produto.

Capítulo 2

Metodologia

2.1 Metodologias de Gestão de Processos

Durante o primeiro semestre foi seguida a metodologia *Waterfall*, um estilo mais clássico, para realização do planeamento, levantamento de requisitos, prototipagem e inicio da arquitectura.

Na fase de desenvolvimento o processo a ser usado é baseado na metodologia ágil *Scrum*. Em conjunto serão aplicados alguns princípios da metodologia *Kanban*, de modo a adicionar uma camada extra de visibilidade sobre o projecto, as *sprints* e as tarefas a serem efectuadas através de um *Kanban Board*. Este processo foi aplicado durante o segundo semestre.

2.1.1 Scrum

Scrum é uma metodologia no qual as equipas podem abordar problemas complexos, de forma produtiva resultando em produtos de alto valor[2]. Scrum é constituído por equipas, que tem associados papéis, eventos, artefactos e regras. O Scrum permite uma gestão flexível do projecto, baseado em iterações de curta duração. No nosso caso cada sprint ocupava 10 dias, centradas na equipa, para ter um controlo mais focado sobre o trabalho desenvolvido. Esta metodologia permite também realizar pequenas alterações ao planeamento que possam vir a surgir sem a necessidade de interromper todo o desenvolvimento. O processo está demonstrado na figura 2.1.

Os três papéis divididos no scrum são: *Product Owner*, *ScrumMaster* e membros da equipa.

O *Product Owner* é responsável pela comunicação da visão do produto à restante equipa. Representam também o interesse do cliente relativamente aos requisitos e à priorização e têm o papel de maior responsabilidade. Este papel é representado pelos professores Bruno Cabral e Jorge Granjal.

O *ScrumMaster* atua como um intermediário entre o *Product Owner* e a equipa. É responsável por garantir o cumprimento das *sprints* e da metodologia Scrum, ajudando sempre que possível os elementos da equipa nas dificuldades de desenvolvimento existente. Esta função é representada pelo elemento Rui Chicória.

A metodologia Scrum foi adaptada para o contexto deste projecto no qual se concentrou na organização dos conteúdos. A implementação iniciou com um backlog detalhado, que representa todas as *user stories* planeadas, com *sprints* projectadas para duas semanas, no período inicial de uma sprint era realizada uma reunião de modo a compreender quais as tarefas que fariam parte da sprint e do estado global do projecto. No inicio de cada dia são realizadas *daily scrum meetings*,semelhantes a *stand-up meeting*, com duração máxima de 15 minutos, ou seja reuniões de curta duração, para discussão das tarefas que devem ser implementadas durante esse dia, obstáculos e o ponto da situação relativo ao dia anterior, de forma natural e informal. No fim de cada sprint ocorre uma reunião para verificar as tarefas realizadas, se existiram desvios ao planeamento inicial da sprint e para compreender o estado do projecto.

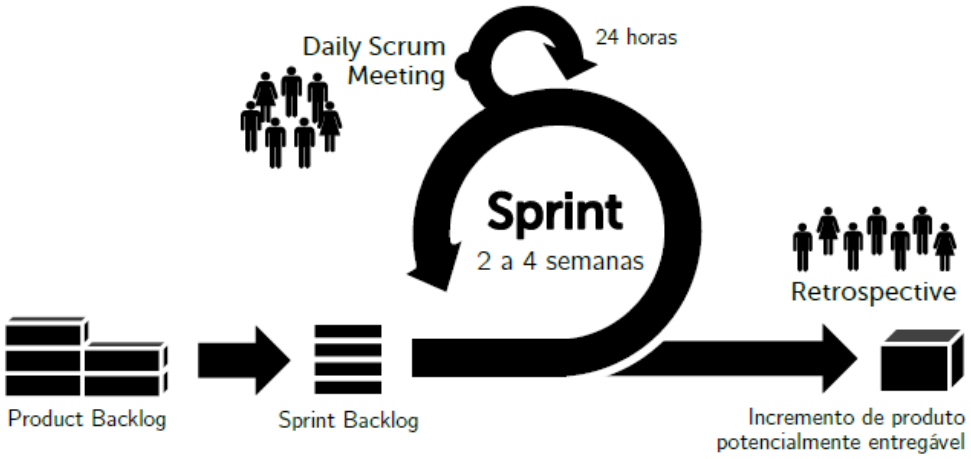


Figura 2.1: Representação da Metodologia Scrum.[3]

Planning Poker

No fim de construção do backlog, e com acesso a todas as *user stories*, é realizado o *planning poker*, uma técnica para estimar bastante utilizada em projetos ágeis. Em equipa, cada membro individualmente, atribui um determinado numero de pontos, *user points* para cada uma das user stories. O objectivo é o grupo chegar a um consenso sobre a estimativa de tempo de construção e contribuir com o seu conhecimento para cada uma das tarefas.

2.1.2 Kanban

Kanban[4], tal como o Scrum, é uma metodologia ágil usada no desenvolvimento de software. Durante o projecto esta metodologia será usada pela forte componente de visualização com a utilização de um *Kanban Board*, como é demonstrado na figura 2.2. Este método é vantajoso uma vez que a performance e a velocidade de um projecto é fortemente estabelecido pelo controlo detalhado de produção e pela percepção da equipa sobre os processos do projecto. Deste modo, o quadro permite facilmente a todos os membros da equipa conhecer o processo e as tarefas atribuídas a cada membro e assim conhecer o fluxo das diversas tarefas. A *board* está dividida nos seguintes fluxos :

- **Backlog**: Dividido entre o fluxo *Product* que contém todas as tarefas que estão por realizar e não se encontram na *sprint* da semana corrente, e o fluxo *Sprint* que inclui as tarefas que estão previstas ser iniciadas e implementadas durante a *sprint*.

- **In Progress**: Representa as tarefas que estão a ser realizadas durante a sprint no qual se encontram organizadas por dificuldade, em que quanto mais perto do topo maior a dificuldade. A distância do fim do fluxo representa o estado da tarefa, quanto mais próximo mais completa esta se encontra. Está presente também o fluxo *Closed* que contém todas as tarefas que se encontram concluídas e preparadas para serem validadas.

- **Staging**: Tarefas que já se encontram implementadas e que podem ser usadas para testes privados e demonstrações.

- **Production**: Tarefas que já se encontram completas e testadas que podem ser incluídas no produto final.

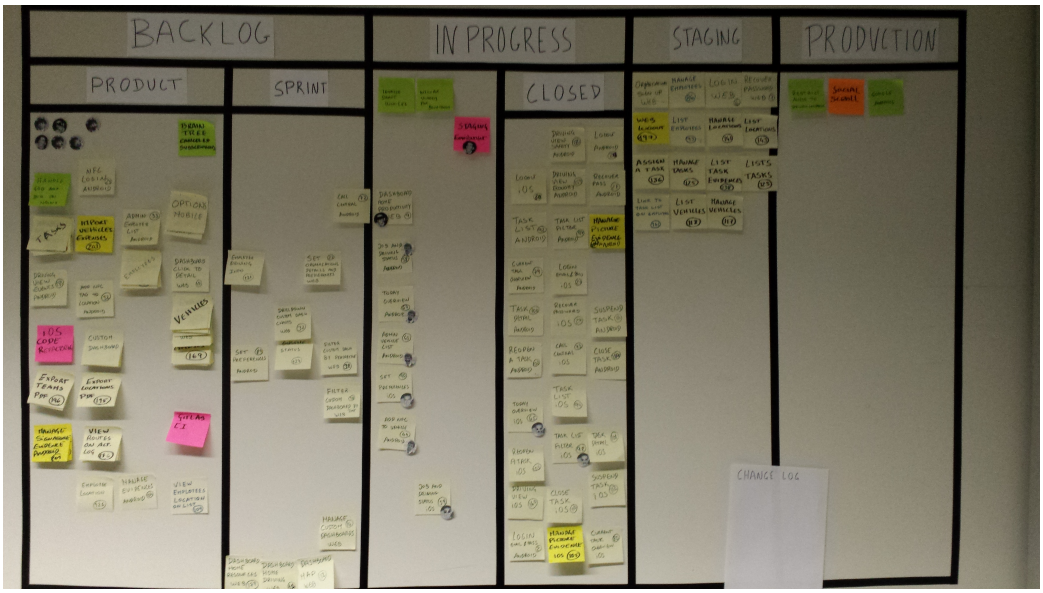


Figura 2.2: Kanban Board usado pela equipa

2.1.3 Extreme Programming

Pair Programming

A técnica de *Pair Programming* consiste em duas pessoas a trabalhar em conjunto apenas com um computador. Ambos se sentam lado a lado em frente ao monitor e concentram-se no código que está a ser escrito por um deles. Tem por objectivo aumentar a qualidade do software sem ter impacto na entrega do produto. O aumento de qualidade no desenvolvimento do software reduz alguns erros e defeitos que poderiam ser obtidos no futuro[5].

Esta técnica foi aplicada no inicio do desenvolvimento, para o estagiário enquadrar-se no projecto, nas tecnologias e linguagens de desenvolvimento.

Test Driven Development

O Test Driven Development (TDD) é uma técnica de desenvolvimento de software orientado a testes, caracterizados por serem escritos antes da implementação. O Test Driven Development é um processo cíclico em que inicialmente é escrito um teste automatizado para uma funcionalidade ou componente a ser implementada. Após a conclusão do teste, é escrito o código necessário, e no fim do ciclo caso seja necessário é efectuada uma optimização do código e reescrito.

Uma vez que os testes são automatizados e são escritos antes da implementação, no decorrer da mesma os testes podem ser executados diversas vezes. Esta técnica faz com que o programador pense no comportamento da funcionalidade em vez da implementação.

2.2 Controlo de versões de Software

De modo a fazer uma gestão eficiente do código desenvolvido pelos elementos da empresa e garantir o controlo de versões foi utilizado o Git[6], através do gestor de repositórios GitLab[7]. Este contém toda a informação necessária para guardar e gerir as revisões e o histórico de um projecto. Sempre que algum dos elementos faz um *commit* no Git, este guarda os meta-dados da mensagem e o autor do *commit* através de *snapshots*. Através de *branchs*, é possivel definir e controlar as versões do código realizado. Neste caso, existe uma *branch* para cada funcionalidade/tarefa a ser implementada, podendo o utilizador mudar o contexto, ou seja mudar de *branch*. Quando concluída uma funcionalidade, ocorre o processo de integração com a *branch* de desenvolvimento), e é realizado um *Merge*. Para garantir que todos os conflitos são resolvidos, e como existe mais do que uma pessoa a trabalhar na aplicação web, se necessário, os programadores envolvidos reúnem-se para resolver os conflitos manualmente.

Para garantir as diversas fases do produto estão em uso 3 diferentes ambientes. Deste modo cada um dos ambientes possuía configurações específicas. O ambiente de desenvolvimento permite aos membros da equipa desenvolver os módulos de software necessários e executar testes unitários. O ambiente de *staging* serve para demonstrar o produto numa versão estável para um determinado cliente e executar testes privados. Este ambiente possui as mesmas configurações que o ambiente de produção, para garantir que todos os componentes se encontram estáveis quando entrarem em produção. O ambiente de produção consiste na versão que é lançada ao publico, com todos os testes já executados [8].

2.3 Análise de riscos

A análise de riscos permite, a qualquer projecto de desenvolvimento, identificar, avaliar e compreender riscos, que podem condicionar o sucesso do estágio, e desta forma minimizar o impacto dos mesmos.

Os riscos são classificados segundo o seu nível de impacto (1-3-5 no qual

1 indica "sem impacto"e 5 "crítico") e a sua probabilidade de acontecer(1-5 no qual 1 representa "improvável"e 5 "muito alto"). Estes são a base para avaliar a magnitude, para verificar a vulnerabilidade a potenciais perdas.

Risco Nº1	Impacto:2	Probabilidade:4
Titulo	*Atraso no planeamento* : Subestimação do tempo definido para cada tarefa devido à falta de experiência do estagiário tarefas, que tem impacto sobre a qualidade de software	
Plano de Mitigação	Reavaliar o plano e dividir as maiores tarefas, em tarefas mais pequenas, garantir que as estimativas são definidas de acordo com o conhecimento técnico do estagiário em conjunto com a equipa.	

Risco Nº2	Impacto:3	Probabilidade:3
Titulo	*Conhecimento do tema do estágio e tecnologias* : O estagiário não tem conhecimento sobre ferramentas ETL, gestão operacional de tarefas, e algumas das tecnologias usadas.	
Plano de Mitigação	Estudo aprofundado das plataformas e das tecnologias, bem como comunicação com os restantes elementos da empresa e técnicas utilizadas como *pair programming*.	

Risco Nº3	Impacto:3	Probabilidade:3
Titulo	*Alteração de funcionalidades na aplicação* : Necessidade de alteração de funcionalidades e características relevantes para o produto	
Plano de Mitigação	Reestruturar alguns elementos do trabalho realizado para que tenham um comportamento mais genérico, para que seja mais simples a realização e integração de novos requisitos.	

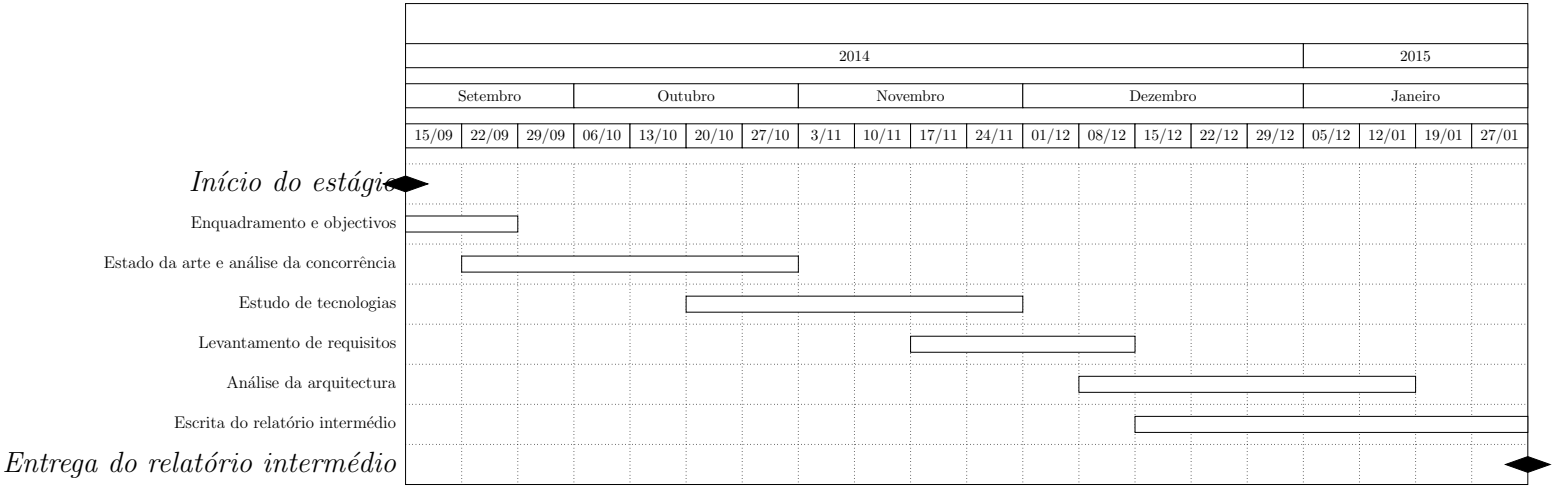
Dos riscos previstos, pode-se afirmar que o plano não foi devidamente calculado, com a dificuldade acrescida no planeamento das tarefas e complexidade que se traduziu em tempo de desenvolvimento superior ao disponível. Devido a esse imprevisto foi necessário reavaliar os requisitos existentes.

2.4 Planeamento

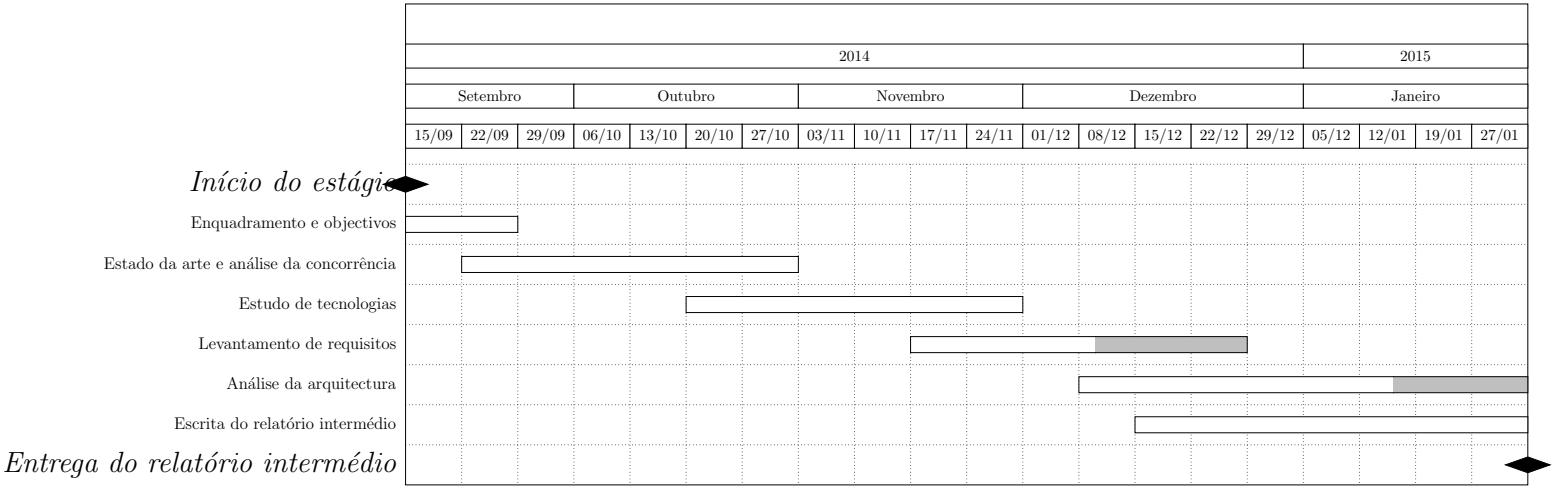
O planeamento do estágio relativamente ao primeiro semestre foi feito recorrendo a diagramas de *Gantt*. Relativamente ao primeiro semestre de trabalho estão associados dois diagramas. O primeiro revela o que foi inicialmente planeado e o segundo resulta do trabalho que foi realmente executado.

O trabalho efectuado durante o primeiro semestre consistiu na análise do estado de arte, definição dos requisitos, prototipagem e inicio da arquitectura. Durante esta fase, a abordagem às etapas foi sequencial, devido às características do trabalho a desenvolver no primeiro semestre.

Planeamento do 1ºsemestre

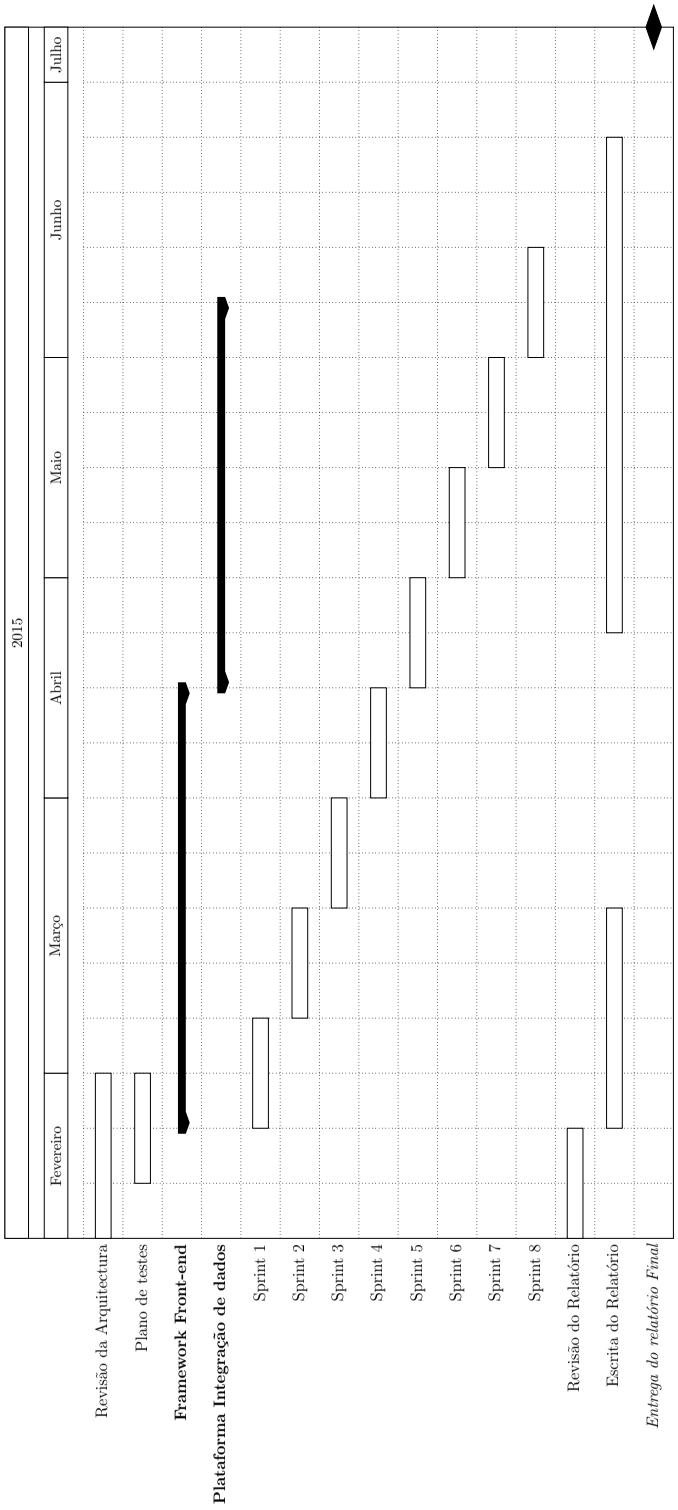


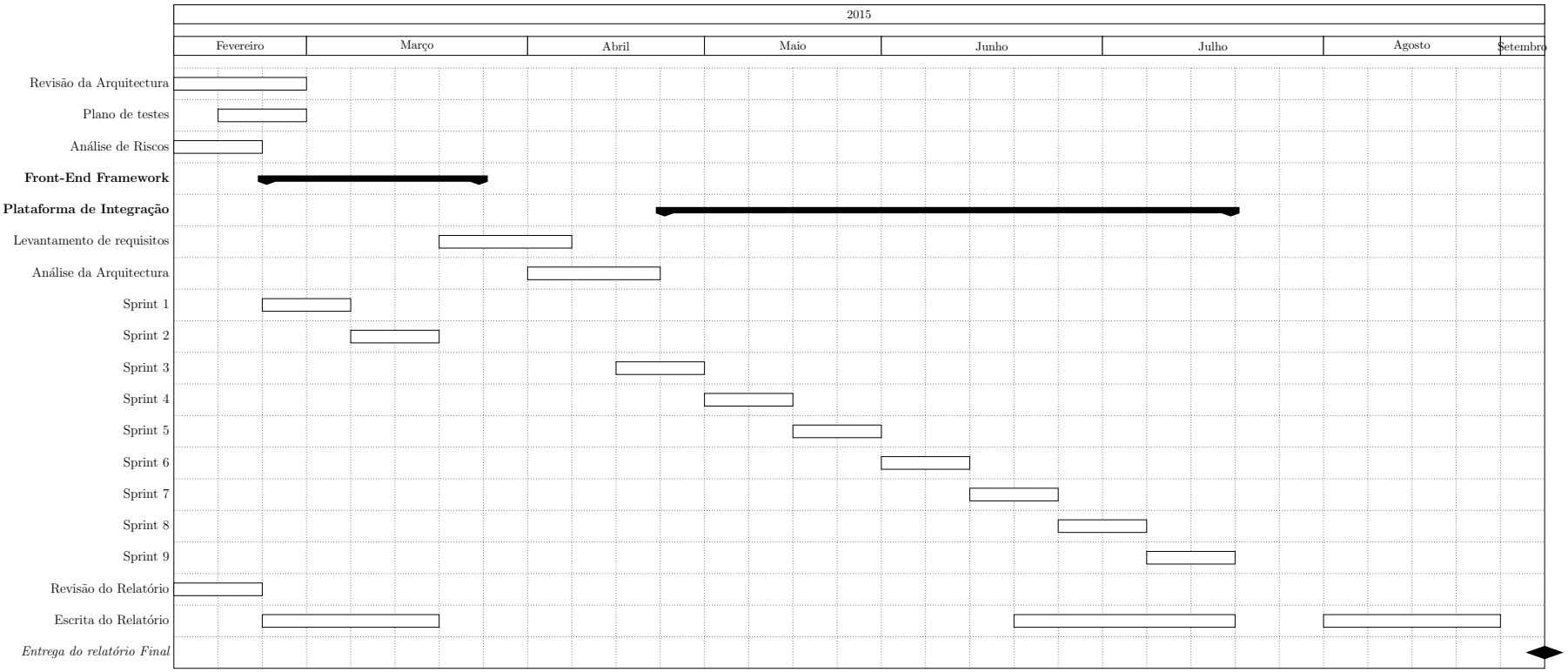
Trabalho realizado do 1ºsemestre



Durante o segundo semestre o planeamento foi baseado segundo uma metodologia ágil definida na sub-secção 2.1.1 recorrendo a sprints para a realização do trabalho definido através das *user stories*.

Devido a decisão colectiva, foi interrompido o processo de implementação da plataforma de *front-end*, que alterou o plano de trabalho e os requisitos.


2015
Fevereiro
Março
Abril
Maio
Junho
Julho
Revisão da Arquitectura
Plano de testes
Framework Front-end
Plataforma Integração de dados
Sprint 1
Sprint 2
Sprint 3
Sprint 4
Sprint 5
Sprint 6
Sprint 7
Sprint 8
Revisão do Relatório
Escrita do Relatório
Entrega do relatório Final


2015
Fevereiro
Março
Abril
Maio
Junho
Julho
Agosto
Setembro
Revisão da Arquitectura
Plano de testes
Análise de Riscos
Front-End Framework
Plataforma de Integração
Levantamento de requisitos
Análise da Arquitectura
Sprint 1
Sprint 2
Sprint 3
Sprint 4
Sprint 5
Sprint 6
Sprint 7
Sprint 8
Sprint 9
Revisão do Relatório
Escrita do Relatório
Entrega do relatório Final

Capítulo 3

Estado da Arte

Nesta secção são apresentados os resultados da pesquisa efectuada sobre o estado da arte, as soluções existentes para gestão telemática de frotas e soluções para gestão de tarefas operacionais. O estudo do estado da arte teve como objectivo compreender o mercado e entender as principais funcionalidades e requisitos que a solução pretendida deve possuir, a interacção com o utilizador, os dados que devem ser apresentados e a estrutura dos mesmos.

3.1 Gestão de Operações

3.1.1 Gestao de Tarefas Operacionais

A gestão operacional de tarefas concentra-se na gestão de processos para a produção e distribuição de serviços, através do planeamento das tarefas, controlo de parâmetros tais como custos e qualidade e uma gestão eficiente dos recursos. Os recursos podem ser humanos, na capacidade de controlar e planear as actividades, que permitem com uma gestão correcta, melhorar a performance dos funcionários e consequentemente da organização. Os recursos materiais concentram-se na manutenção e gestão de edifícios, de equipamentos, entre outros[9].

Os principais objectivos desta gestão são:

- **Serviço ao cliente**: Consiste em usar os recursos com o intuito de satisfazer o cliente através de redução de custos, tempo, entre outras.

- **Utilização dos recursos**: Manutenção e gestão dos recursos eficientemente de modo a obter o melhor desempenho de cada.

– **Planeamento de tarefas** Planear, controlar e analisar cada tarefa.

Soluções Comerciais

In4Tools

A In4Tools[10] é uma empresa que possui diversos módulos de gestão dentro do mesmo produto, 4Biz[11], incluindo gestão operacional e manutenção. Pretende proporcionar a optimização das actividades administrativas e operacionais. Os dados fornecidos são acedidos através de uma plataforma web, em tempo real, que contem alertas automáticos sobre a organização.

Esta solução permite criar tarefas e associar as mesmas a um utilizador, bem como verificar o seu estado, por exemplo, se as tarefas estão concluídas ou se já excederam o tempo limite de realização e criar localizações através de coordenadas *Global Positioning System* (GPS) para que se identifique mais facilmente as posições mais relevantes. É possível gerar relatórios das tarefas existentes e exportar posteriormente para o formato *Comma Separated Values* (CSV).

Jobber

O Jobber[12] é um software para gestão de negócio *web-based*, que permite fluxos de trabalho programados e não programados, gestão de tarefas e facturação. A informação sobre pagamentos encontra-se centralizada permitindo aos gestores um acesso mais detalhado aos relatórios financeiros. As principais funcionalidades deste produto são :

- Calendarização de tarefas a serem realizadas: com acompanhamento regular do progresso diário, podendo ser associados vários funcionários à mesma tarefa.

- Facturação: reduzindo o tempo de envio aos clientes, através de um sistema automático de envio.

- Relatórios personalizados: que permitem compreender o que deve ser melhorado, que os utilizadores podem criar e guardar.

- Informação em tempo real: que permite optimizar as rotas das tarefas não realizadas pela localização, diminuindo o tempo entre as tarefas a serem realizadas naquele dia.

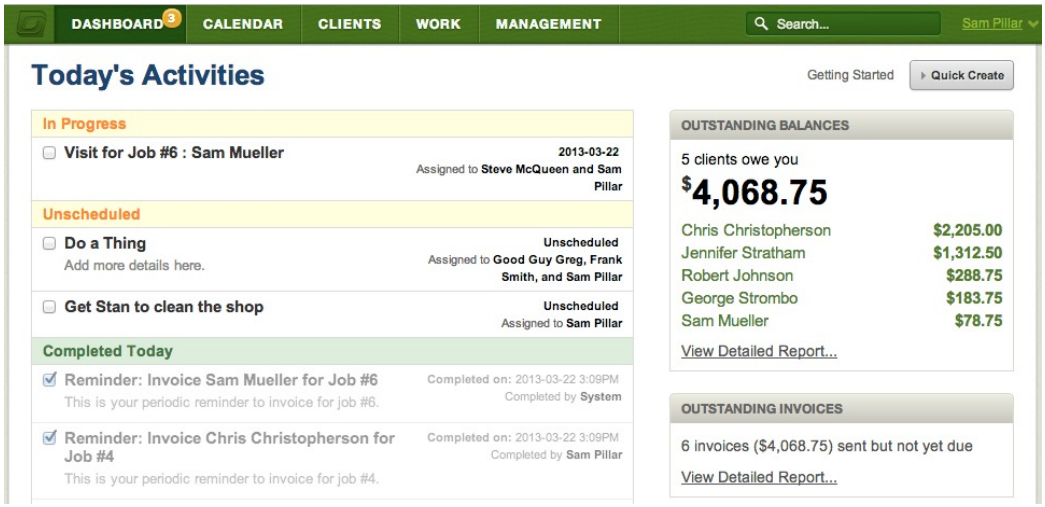


Figura 3.1: Dashboard da solução Jobber. Retirado de [13].

FieldAware

FieldAware[14] é um serviço criado em 2009, com o intuito de melhorar a gestão de operacionais inicialmente desenvolvido para mobile. A plataforma inclui relatórios e escalonamento automáticos, comunicação em tempo real entre o administrador e os operacionais com partilha de dados relevantes tais como localização, dados de tarefas, e planos realizados. Estes dados permitem criar e actualizar as rotas, consoante as tarefas, de modo constante e automático, permitindo optimizar as operações e serviços existentes, aumentar a produtividade, e reduzir custos de manutenção e consumo.

A solução criada é independente do negócio, flexivel, que permite a integração da plataforma com os serviços já existentes, através de uma *Open Application Programming Interface (API)* que permitem comunicar com outras aplicações Web e com uso de conectores criados *on demand* para cada tipo de negócio.

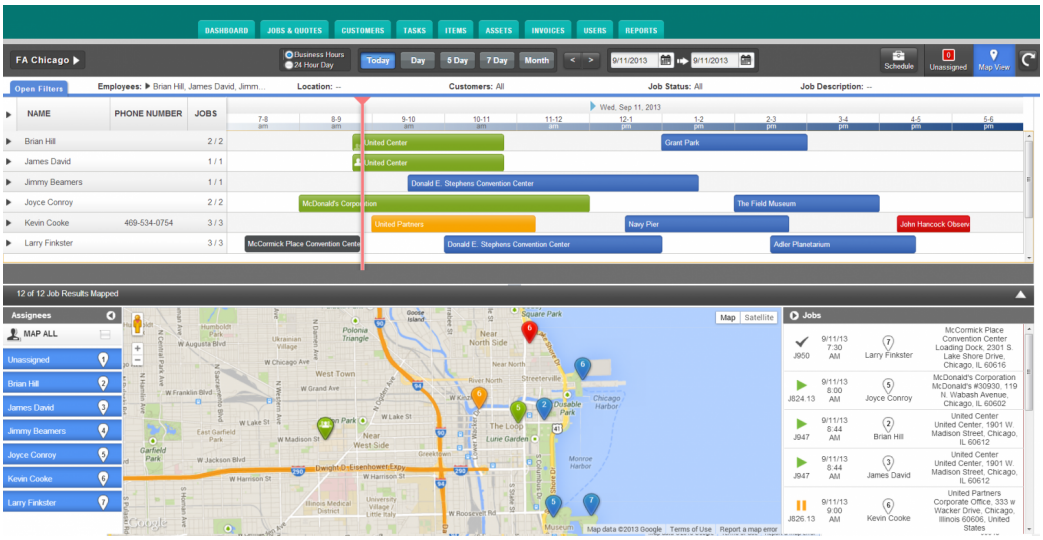


Figura 3.2: Dashboard da solução FieldAware.Retirado de [15].

Synchroteam

Synchroteam [16] é um produto que conecta ambas as vertentes, aplicação web e aplicação mobile, que tem como principal função planear de forma ágil e eficiente todas as tarefas a serem desempenhadas. Através da aplicação web é possível ter uma vista completa sobre as tarefas do dia, planear rotas automaticamente e alterar o plano actual, gerir e associar de modo claro e compreensível as tarefas, com notificações em tempo real, com personalização para cada tipo de negócio. Para este efeito, a aplicação já contém um conjunto de funcionalidades para integrar com diferentes serviços, com conectores previamente selecionados pelo utilizador. É possível também integração com *Web Services* já definidos, através de uma API bem documentada, com sistemas de informação já existentes na organização tais como *Customer Relationship Management* (CRM), *Enterprise Resource Planning* (ERP) ou outros serviços Web.

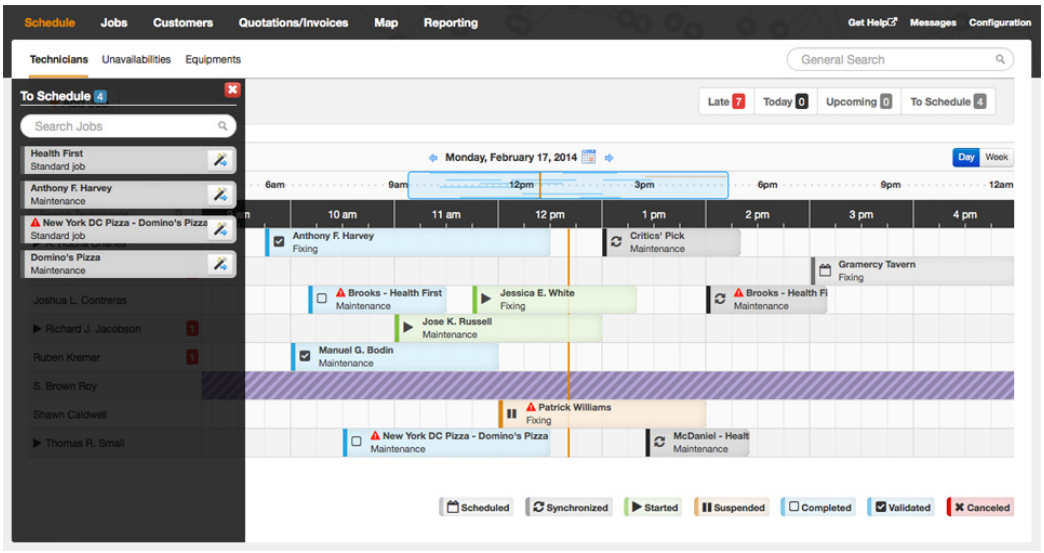


Figura 3.3: Calendarização de tarefas da plataforma Synchroteam. Retirado de [17].

3.1.2 Gestão de Frotas

Alguns dos objectivos desta gestão são:

- **Melhorar desempenho e lucros**: O acesso à eficiência dos veículos, à produtividade dos condutores e organização de rotas permitem ajudar na redução de custos de manutenção, tempo de condução e utilização dos funcionários [18].
- **Melhorar operações das frotas**: Com o uso de um sistema de GPS *tracking* os dados recolhidos sobre as operações realizadas não estão dependentes dos condutores, permitindo uma tomada de decisão mais eficiente e um aumento de tarefas realizadas por dia.
- **Redução de custos de combustível e emissões CO2**: Melhorar o comportamento dos condutores e reduzir eventos tais como acelerações ou travagens bruscas.
- **Manutenção de veículos e custos**: Monitorizar cada veículo e despesas associadas com uso de alertas garantindo uma manutenção eficiente e o aumento do ciclo de vida de um veículo.
- **Aumentar a segurança do condutor**: Através dos registos dos eventos que o condutor produz durante a condução, é possível treinar os

condutores para uma melhor condução. A localização GPS também pode ser usada para casos de emergência.

Soluções comerciais

Teletrac

A Teletrac FleetDirector, da empresa Teletrac[19],é um produto que permite automaticamente reconhecer a localização de veículos, gestão da frota e sistema de comunicação. Através do uso do dispositivo de localização GPS Prism[20] é possível obter informações do veículo em tempo real como dados de condução e localização, que através de uma solução *cloud based*, é possível obter comunicação constante e relatórios detalhados. A plataforma permite um controlo geral da frota, aumentar a produtividade e reduzir custos, através de equipamento móvel ou *browser*. O software encontra-se desenhado para interagir com frotas que possuam um vasto número de veículos.

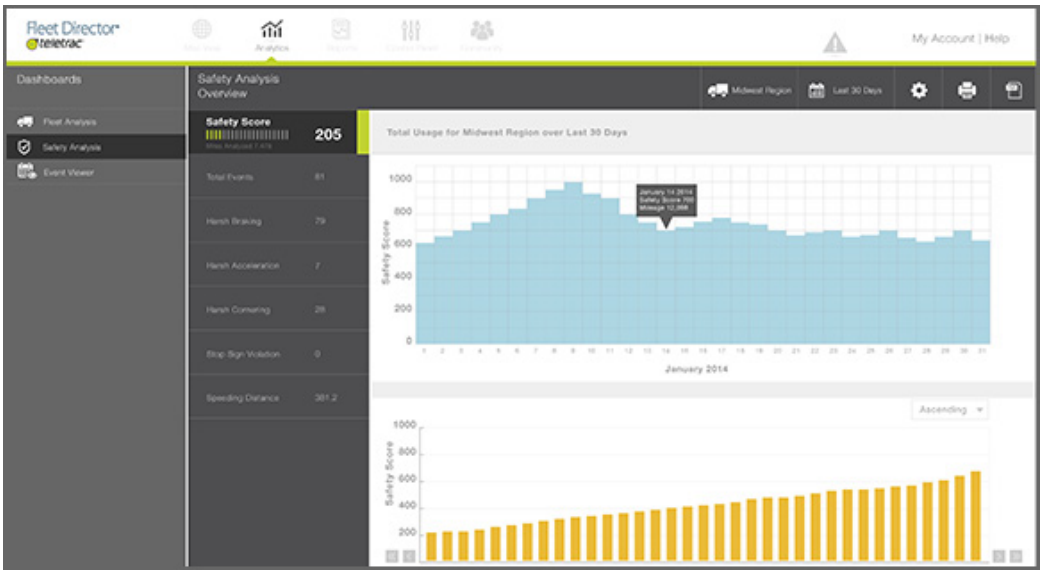


Figura 3.4: Solução Teletrac. Retirado de [21].

FleetMatics

A FleetMatics[22][23] é uma empresa que desenvolve soluções oferecidas como Software as a Service(SaaS). Fornece monitorização em tempo real da frota de veículos, dados históricos referentes de cada veículo como inicio e fim de viagem,eventos de condução como excesso de velocidade e travagens bruscas,

o estado do veículo entre outros. Em cada veiculo encontra-se instalado um equipamento receptor GPS que permite através de uma interface *web based*, obter informação centralizada da actividade da frota.

Através dos dados recolhidos, é criado um conjunto de métricas tais como eventos de condução, productividade, quilometragem efectuada em cada trajecto, disponibilizado através de relatórios e análises personalizadas dependendo da necessidade da empresa.

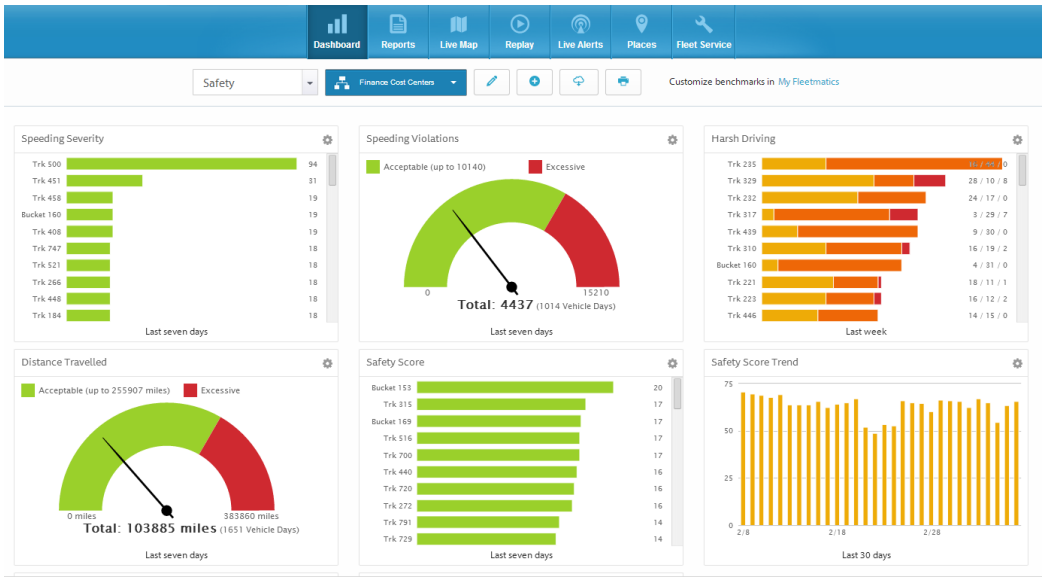


Figura 3.5: Dashboard da solução FleetMatics. Retirado de [24].

Inosat

A Inosat é uma empresa portuguesa a actuar no mercado da gestão e localização de frotas via GPS focado principalmente no sector empresarial B2B. Inofrota, um dos serviços mais relevantes na gestão de frotas para a Inosat, que contém quatro produtos, *Base*, *Pro*, *Light* e *Web* para se adequar à estrutura empresarial do cliente.

O sistema encontra-se baseado em equipamentos com a tecnologia GPS, instalados em cada veiculo, para que seja possível gerir em tempo real, com o objectivo de rentabilizar e optimizar os recursos da empresa. A transmissão de dados, entre as viaturas da frota e a empresa, é efectuado por e *Global System for Mobile Communications* (GSM) / *General Packet Radio Service* (GPRS), obtendo assim uma actualização de dados constantes e imediatos. A actualização de dados informativos pode ser alterada para que as actuali-

zações ocorram apenas em determinados momentos para reduzir os custos de transmissão. O sistema é acedido através de uma aplicação Web, com a finalidade de ser acedido por vários utilizadores de diferentes modos. O Sistema Inofrota pretende aumentar a produtividade através da detecção dos tempos de paragem indevidos da frota, prevê a rota mais eficiente através de optimização de rotas, reduzindo custos e rentabilização de tempo despendido. É possível detectar eventos de condução, como travagens e acelerações bruscas, para melhorar a segurança.

A empresa tem acesso a relatórios detalhados para analisar a performance da frota e assim controlar e reduzir significativamente os custos desta.

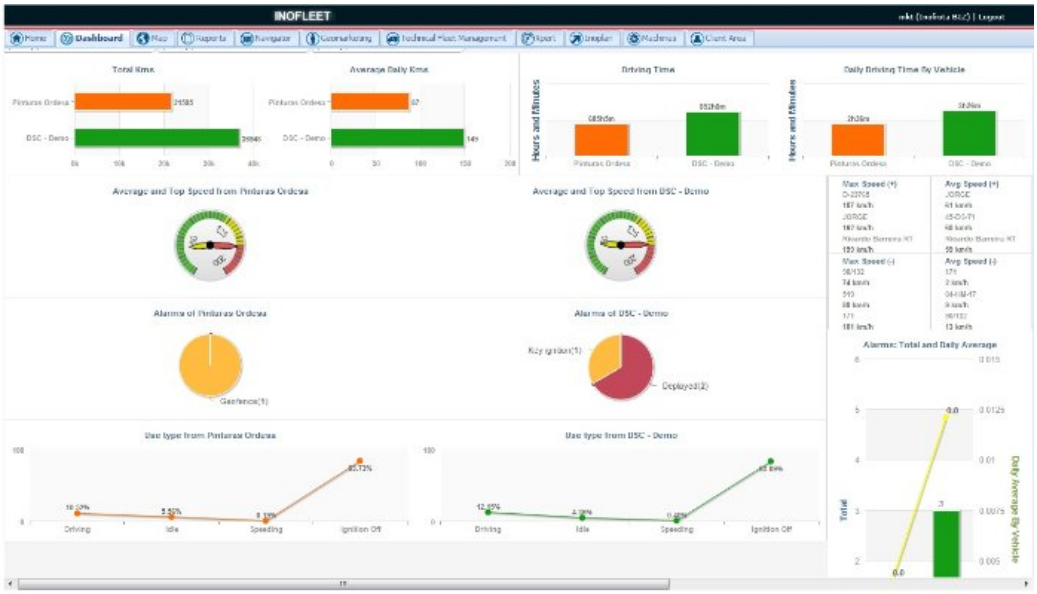


Figura 3.6: Dashboard da aplicação Inofrota. Retirado de [25].

Masternaut

A Masternaut[26] é uma empresa com foco principal no mercado Europeu, dedicada a desenvolver soluções telemáticas orientada para o apoio à gestão de frotas e operacional, apresentadas como *Software as a Service* (SaaS), com mais de 10,000 clientes em 32 países para diversos sectores de negócio. A Masternaut Connect é o produto principal da empresa e funciona como uma plataforma adaptável capaz de incluir diferentes módulos e aplicações, independentes entre si para que o cliente possa escolher as funcionalidades mais relevantes para a sua empresa sem que seja necessário adquirir a totalidade do produto. Estas aplicações permitem uma gestão constante da frota em

tempo real, monitorização e recolha de informação detalhada dos veículos, redução de gastos de manutenção e despesas, aumento de produtividade, controlar e melhorar o comportamento do condutor, redução de emissões CO2 entre outros.

Através de um equipamento *onboard* colocado em cada veiculo, capaz de localizar a posição constante dos veículos através de GPS e de transmissão de informação através de tecnologia GPRS, é possível às empresas criarem relatórios detalhados relativos à produtividade global e individual, controlar e comparar condutores, estado actual das viaturas, despesas e manutenções a ser realizadas no futuro, detectar eventos realizados pelo condutor como acelerações excessivas e travagens abruptas, informação que o condutor também pode visualizar, aumentando a segurança e reduzindo possíveis riscos.

Em Portugal a Masternaut[27] em conjunto com outros serviços facilita a adaptação e apoio para integrar com os serviços da Masternaut. A Masternaut Portugal, caso os serviços adaptados não se encontrem suficientes, desenvolve soluções dependendo do sector de actividade ou especificas para o cliente.

Figura 3.7: Dashboard da solução Masternaut. Retirado de [28].

RTA Fleet

O software *RTA Fleet Management*[29] foca-se principalmente na manutenção de uma frota, independente do negócio e tamanho da empresa. Através do uso de alertas, o utilizador recebe informação sobre manutenção de veículos, controlo e gestão de funcionários. Uma das principais vantagens deste sistema é realizar um controlo rigoroso sobre as despesas inerentes a uma frota e manutenção que permite aumentar o tempo de uso dos veículos. É possível adicionar módulos internos, como relatórios sobre o condutor ou um módulo que mantém um registo de dados relacionados com o acidente tais como local, hora, documentos do seguro e despesas. O objectivo é aumentar a eficiência através de disponibilização agrupada e correcta da informação relevante da frota.

3.1.3 Comparativo

Para realizar a análise das diversas soluções foram usados alguns dos seguintes critérios:

- **Características:** as principais funcionalidades e vantagens de cada produto tais como monitorização em tempo real, análise de informação e gestão de tarefas.
- **Custos:** associados à aquisição e manutenção de cada produto.

Como existem diversos produtos para gestão de frotas e gestão operacional, e para tentar encontrar os mais importantes e relevantes no mercado, foram também usados motores de procura de software [30][31].

Uma comparação entre as soluções estudadas encontra-se na tabela 3.1.

Tabela 3.1: Comparativo entre soluções de gestão estudadas.

	Teletrac	FleetMatics	Inosat	Masternaut	RTA Fleet	4biz	Jobber	FieldAware	Synchroteam
Aplicação *Web-based*	✓	✓	✓	✓	✗	✗	✓	✓	✓
Registo de viagens	✓	✓	✓	✓	✗	-	-	✗	✗
Localização GPS dos veículos	✓	✓	✓	-	✓	-	✗	✓	✓
Monitorização em tempo real	✓	✓	✓	-	✗	✓	-	✓	✓
Posição dos veículos no mapa	✓	✓	✓	✗	✗	-	-	✓	✓
Alertas proactivos	✓	✓	✓	✓	✗	✗	✓	-	✗
Eventos de condução	✓	✓	-	✓	✗	✗	✗	✗	-
Relatórios	✓	✓	✓	✓	✓	✓	✓	✓	✓
Manutenção de veículos	✓	✗	✓	✓	✓	✗	✓	✓	✗
Criação de Tarefas	✗	✓	✗	✓	✗	✓	✓	✓	✓
Calendarização de Tarefas	✗	✓	✗	✓	✗	✓	✓	✓	✓
Sistema de Facturação	✗	✗	✗	✗	✗	✓	✓	✓	✓
Integração de conteúdos de terceiros	✓	✗	✗	✗	✗	-	✗	✓	✓

Através da tabela 3.1 é possível concluir que não está presente no mercado uma solução que permita num só produto obter as funcionalidades de gestão de frotas conjuntamente com uma gestão operacional e tarefas. É de destacar o caso da Masternaut e FleetMatics que realizam uma gestão operacional através de integração com produtos internos ou externos à respectiva empresa. Este estudo não foi suficiente para compreender todas as necessidades da plataforma de integração de dados mas foi importante para perceber as necessidades dos clientes neste aspecto.

3.2 Integração de Dados

Para realizar a integração de dados, foi realizado um estudo sobre algumas ferramentas e sobre alguns conceitos que são necessários para o desenvolvimento e arquitectura do mesmo, apoiado inclusive pelos conceitos do processo Extract Transform Load (ETL).

3.2.1 Conceitos ETL

De seguida apresentam-se os conceitos inerentes à realização do módulo de integração de dados. Estes são importantes para compreender as funcionalidades relativamente ao acesso e manipulação dos dados.

Aquisição de dados e Data Profiling

Este procedimento inclui distribuir os dados fisicamente, para iniciar o ciclo de vida do processo e consequentemente analisar sistematicamente o conteúdo dos dados e a sua qualidade, para conhecer os dados antes de serem rectificados.

Limpeza dos dados

Para melhorar a qualidade dos dados, uma importante contribuição é a limpeza dos mesmos. É um processo iterativo, fundamentado pelas decisões de negócio e por padrões estabelecidos que cumprem os requisitos de qualidade. O objectivo é corrigir os valores que se encontram incorrectos ou preencher dados que não estão presentes ou incompletos.

Mapeamento dos dados

Conjunto de processos que consistem em criar uniformidade e executar a correspondência dos dados, para eliminar duplicados e criar uma versão única

que é compreendida pelo sistema. Também inclui exposição de erros ou excepções para, caso relevantes, sejam feitas as modificações necessárias.

Transformação dos dados

Significa modificar os dados para existir compatibilidade e obedecer aos padrões esperados pelo destino dos dados. Um exemplo é estabelecer um conjunto de valores para um determinado campo ou fundir diferentes campos a um único elemento.

Escrita dos Dados

Este processo consiste em escrever o resultado obtido das operações anteriores para a base de dados alvo.

3.2.2 Sistemas de Integração de Dados

De seguida apresentam-se as soluções estudadas nas quais foi considerado a popularidade das mesmas, as funcionalidades, e as vantagens em relação a outras concorrentes, bem como a relevância das mesmas para o estágio.

Kettle

O sistema de integração de dados da empresa Pentaho, Kettle [32], é responsável por extracção, transformação e carregamento dos dados, que pode ser usado para migração entre base de dados ou aplicações, carregamento de informação para a base de dados, ou integrar com outras aplicações. Uma das principais características deste sistema é a possibilidade de incorporar sistemas externos, tais como base de dados ou *web services*, com *plugins* já existentes.

Informatica Powercenter

O *Powercenter* [33] é um produto multi plataforma, *cloud based* que permite a conexão com aplicações empresariais e base de dados. Este produto permite planear e monitorizar a obtenção e qualidade dos dados. Apesar de não ser possível ao cliente desenvolver novos módulos, o sistema é vendido como *Software as a Service* (SaaS), já que contém um conjunto vasto de *Add-Ons* que também são desenvolvidos *on-demand* para cada cliente.

Adeptia

A solução Adeptia [34] consiste em três diferentes componentes. Possui uma versão web-based que permite de forma fácil realizar mapeamentos e validação de dados personalizados para cada organização e a capacidade de gerir facilmente o destino e origem dos dados. O segundo componente funciona como um repositório do serviço e permite guardar as regras e mapeamentos realizados. Por fim o terceiro componente é uma ferramenta que se encontra na máquina do cliente, e permite executar as transacções anteriormente criadas através da primeira componente e com a informação que se encontra no repositório. É possível integrar com diferentes fontes de dados, tais como *web-services*, base de dados ou ficheiros.
Foi a partir desta solução que foi retirado o conceito de *File Adapter* que consiste num ficheiro que permite realizar o mapeamento entre o destino e a origem dos dados.

3.2.3 Comparativo

A análise e comparação destas ferramentas serve para compreender as principais funcionalidades e características, tal como a modularidade das mesmas e os diferentes conectores integrados em cada.

Tabela 3.2: Comparativo entre as ferramentas de integração estudada.

Soluções Comerciais	Kettle	Powercenter	Adeptia
Comercial	✓	✗	✓
Open Source Comercial	✗	✓	✗
Aplicação Web-based	✗	✗	✓
Suporte de integração na *cloud*	✓	✓	✓
Monitorização e *triggers* em tempo real	✗	✓	✓
Alertas e notificações	✓	✓	✓
Performance com grande volume de dados	✓	✓	✓
Fácil união de Add-Ons Externos	✗	✗	✗
Validação dos dados	✓	✓	✓
Conectores para Web Services previamente definidos	✓	-	✓
Conectores para bases de dados	✓	✓	✓
Integração com *flat files*	✓	✓	✓
Mapeamento dos Dados	✓	✗	✓

Pelo comparativo da tabela 3.2, é possível verificar semelhanças na forma como os dados são extraídos e tratados. O estudo realizado permitiu compreender quais as funcionalidades que devem ser estudadas e se possível presentes no produto desenvolvido.

3.3 Conclusões

Depois de revistas as soluções que se enquadram no âmbito deste projecto, compreende-se que existe uma procura constante na optimização dos processos e redução de custos financeiros. Assim é cada vez mais comum encontrar aplicações que procuram unir a tecnologia ao controlo e gestão de frotas. As soluções existentes na gestão de frotas não permitem a execução de tarefas do dia-a-dia sem a necessidade de custos adicionais através de integração com outros serviços ou produtos. Com este estudo foi possível adquirir competências relativas ao produto *DrivianPro*, tais como entender qual a informação e como esta deve ser representada, funcionalidades essenciais, e compreender

os mecanismos de integração que algumas das plataformas utilizam para realizar recepção de dados de fontes externas. Esta plataforma para a empresa é de extrema importância pois normalmente a integração de dados é realizada manualmente pelas organizações, com necessidade de ferramentas externas para conectar com os serviço e sempre que são adicionados novos serviços é necessário refazer a plataforma e os mecanismos envolvidos. Este estudo permitiu compreender quais são as características importantes, a nível funcional e visual, a estrutura e o fluxo de processos necessários para desenvolver a plataforma.

Capítulo 4

Análise de Requisitos

A especificação das funcionalidades do projecto foram realizadas com recurso a *user stories*. São usadas para organizar requisitos e consistem em breves descrições representando as funcionalidades expostas na perspectiva do utilizador. Inicialmente o *Product Owner* forneceu à restante equipa alguns requisitos fundamentais que a aplicação deveria possuir. É consequentemente debatido cada um destes requisitos por todos os elementos e realizado o levantamento das *user stories*. A realização da prototipagem é uma fase de grande valor para a construção de novas *user stories* e alteração de existentes. Neste capitulo, serão analisados os requisitos funcionais e não funcionais que o sistema deve incluir, os actores do sistema intervenientes nas plataformas e a prototipagem de alta fidelidade que serviu de apoio na construção dos requisitos.

4.1 Actores

Os actores do sistema dividem-se entre utilizadores da aplicação móvel que fornecem os dados sobre localização e eventos de condução, utilizadores da aplicação web que representam os administradores da organização, e gestores da plataforma de integração. Esta informação encontra-se representados na tabela 4.1.

Os actores do sistema representados pelos utilizadores da aplicação móvel não fazem parte do desenvolvimento do estagiário mas enquadram-se no âmbito do projecto.

Actor	**Representa**	**Função**
Operacional da Empresa	Utilizador do aplicativo móvel	Regista as viagens com informação representativa do veículo, tarefas e eventos de condução
Administrador	Utilizador da aplicação Web	Gera os relatórios relativos à organização, extrai métricas importantes para a tomada de decisão e tem conhecimento sobre toda a organização.
Gestor da plataforma de integração	Equipa exterior à organização	Equipa de desenvolvimento que desenvolve os módulos de conectorização, por parte da empresa, que realiza a integração necessária para ser usado pelo cliente.

Tabela 4.1: Actores do Sistema

4.2 Prototipagem

A utilização da prototipagem é uma estratégia bastante usada já que permite estruturar, planear e desenhar a interface de utilização de uma aplicação, que deve possuir uma interface intuitiva e de alta usabilidade. A prototipagem foi bastante importante para a construção das *user stories*.

Esta técnica é produzida antes da fase de design e desenvolvimento e normalmente utilizada para inicialmente compreender e melhorar os requisitos, fornecendo ao cliente uma perspectiva diferente e um conhecimento visual sobre o comportamento da aplicação. O desenvolvimento desta técnica permitiu delinear inicialmente os requisitos essenciais ao produto, sendo necessário uma segunda avaliação e reconstrução dos protótipos inicialmente gerados de modo a obter uma melhor fiabilidade entre os protótipos e os requisitos estabelecidos.

Na figura 4.1 encontra-se o fluxo pelo qual foi realizada a técnica de prototipagem representando as etapas necessárias.

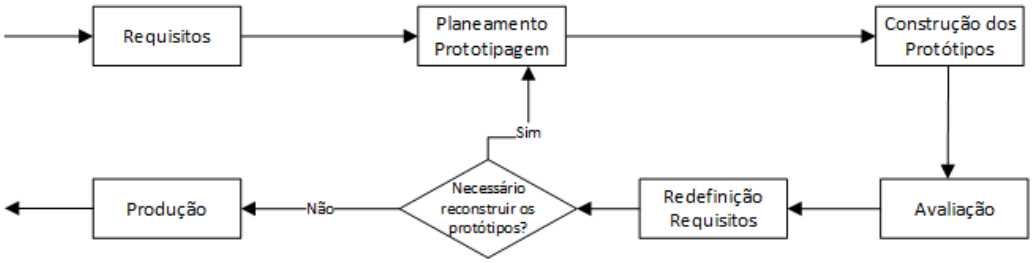


Figura 4.1: Diagrama do Modelo de Prototipagem

Para a realização deste procedimento, foram desenvolvidos protótipos de alta fidelidade, foram abordadas ambas as vertentes do estágio, a *framework* capaz de automatizar o desenvolvimento do *front-end* da aplicação e a plataforma para realizar a integração dos dados.

Front-end Framework

O protótipo representado na figura 4.2 representa a informação que deve ser apresentada ao utilizador relativamente ao detalhe de uma tarefa. É possível editar ou remover a tarefa, ver a rota, informação de produtividade, informação geral entre outros dados e métricas. O protótipo que se encontra na figura 4.3 representa o detalhe de um veículo. Através de uma comparação entre ambas é perceptível a necessidade de uma implementação por módulos que permita gerar interfaces.

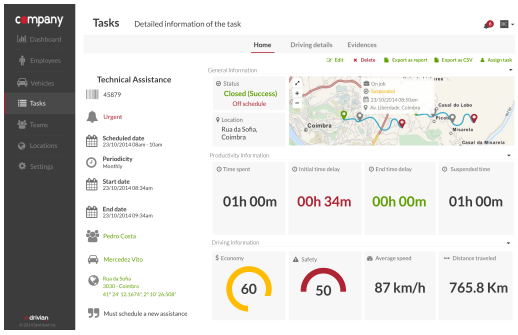


Figura 4.2: Detalhe de uma tarefa

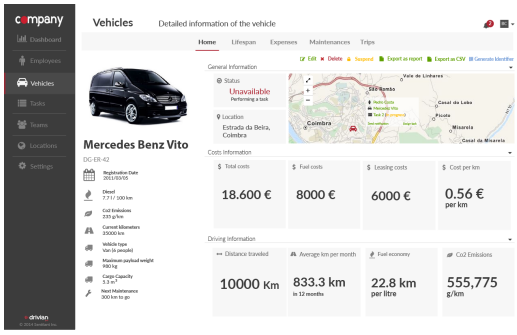


Figura 4.3: Detalhe de um veículo

O mesmo caso acontece aos protótipos relativos à listagem como o exemplo nas figuras 4.4 e 4.5.

Como é possivel denotar, com o recurso a uma *framework* para realizar algumas das funcionalidades do *front-end* é possível reduzir tempo de

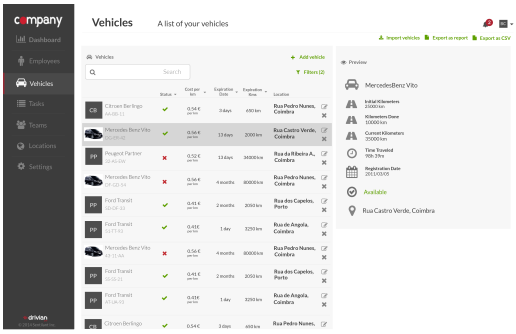


Figura 4.4: Catalogação de veículos

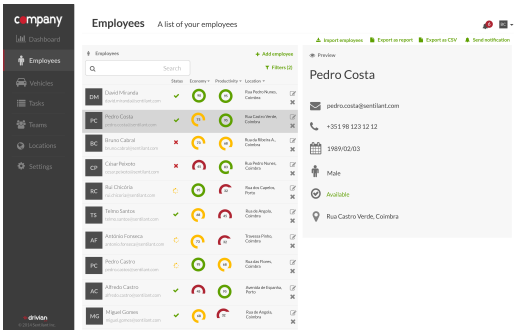


Figura 4.5: Catalogação de funcionários

desenvolvimento futuro.

Integração dos dados

Nas figuras seguintes, são apresentadas algumas das funcionalidades representativas da plataforma que realiza a integração de dados. O principal objectivo desta ferramenta é a obtenção de dados de diversas fontes, tal como a obtenção através de uma base de dados ou através de serviços já usados pela organização.

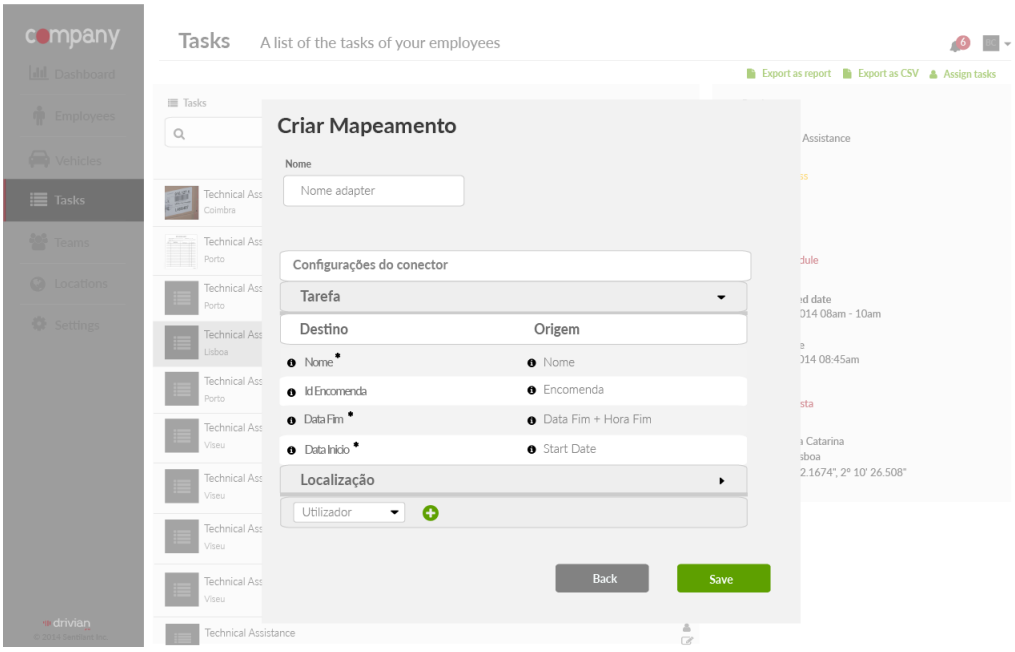


Figura 4.6: Mapeamento dos dados

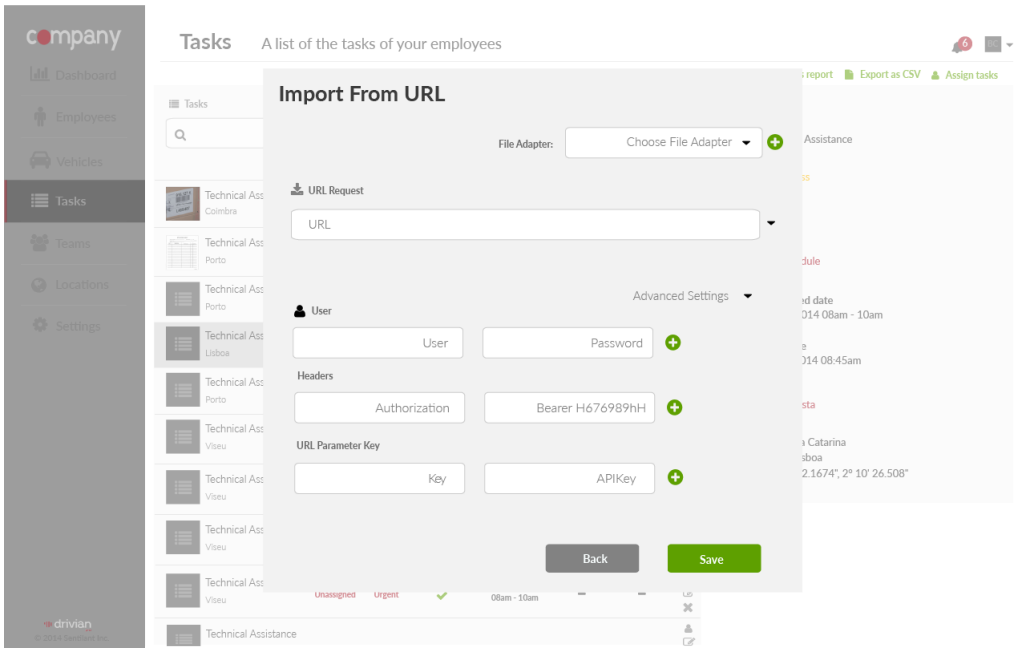


Figura 4.7: Importar dados provenientes de uma REST API configurada previamente.

Na figura 4.6 está representado a construção do mapeamento que é necessário para realizar a integração de dados, independente do tipo de importação que seja realizada. Na figura 4.7 está representado a obtenção de dados através de uma API REST, e as configurações que necessitam de ser realizadas para establecer conecção com a mesma.

4.3 Requisitos funcionais

Os requisitos funcionais foram construidos com recurso a *user stories*[35]. O uso destas para a definição dos requisitos tem a utilidade de fornecer a persectiva do utilizador final já que reflectem as necessidades e a interacção com a aplicação. Através das *user stories* é possivel extrair funcionalidades e acções, assim como a prototipagem, retirando tempo numa fase inicial de desenvolvimento. São de simples representação e sem necessidade de conhecimentos técnicos o que permitem uma fácil leitura. São também usados para a construção de testes funcionais e de integração.

O formato mais utilizado pela sua definição[35] é a seguinte:

- Enquanto **<actor do sistema>**
- Quero **<acção>**
- de forma a **<proposta de valor>**

As *user stories* pretendem responder a três questões na definição de funcionalidades:

- O que se pretende?
- Quem o pretende?
- Porquê?

Relativamente à primeira fase do estágio encontra-se no anexo A as *user stories* que funcionaram de base para o desenvolvimento da ferramenta.

De seguida encontram-se listados, sucintamente, os requisitos funcionais relativos à integração de dados. Estes requisitos, tais como os descritos anteriormente, seguem a técnica mencionado na figura 4.1.

Para prioritização de cada uma das *stories* foi usado o método MosCoW [36].é uma técnica de priorização de requisitos pela utilização de palavras com significado.Esta divide os requisitos em quatro diferentes categorias:

Categoria	Representação
Must	O requisito deverá estar contido no serviço final.
Should	Requisito que se possível deve estar na solução final.
Could	Requisito desejável mas não necessário.
Won't	Requisito decidido que não seria implementado na versão corrente mas possivelmente importante na continuidade do projecto.

Tabela 4.2: Escala de MoSCoW

ID : 1	Priority: Must
Como Utilizador Web **Quero** Importar múltiplos ficheiros com diferentes formatos tais como xml, csv ou json para que seja possível armazenar novos recursos na base de dados.	

ID : 2	Priority: Low
Como Utilizador Web **Quero** Importar um ficheiro através de um Url para que seja possível armazenar novos recursos.	

ID : 3	Priority: Must
Como Utilizador Web **Quero** Obter um ficheiro modelo, para preenchimento com dados úteis da organização.	

ID : 4	Priority: Could
Como Gestor da plataforma de integração **Quero** Criar e armazenar uma conexão para diferentes bases de dados (sqlite, mssql server, postgres, mysql) para fornecer ao utilizador uma conexão sem a necessidade de repetir as configurações.	

ID : 5	Priority: Won't
Como Utilizador Web **Quero** Criar e armazenar uma conexão para diferentes bases de dados para fornecer ao utilizador uma conexão sem a necessidade de repetir as configurações.	

ID : 6	Priority: Should
Como Utilizador Web **Quero** usar uma conexão existente para um modelo de dados especifico para que seja possível adicionar novos recursos através de conectores definidos.	

ID : 7	Priority: Could
Como Utilizador Web **Quero** adicionar um link e as configurações necessárias para aceder a diferentes API's REST para que seja possível obter dados(xml/json) e adicionar novos recursos.	

ID : 8	Priority: Could
Como Utilizador Web **Quero** obter autorização através de um protocolo de autenticação, para realizar um novo pedido e adquirir novos recursos.	

ID : 9	Priority: Won't
Como Utilizador Web **Quero** visualizar e utilizar um *token* válido de uma conexão anterior para uma REST API, para não repetir o processo de autenticação.	

ID : 10	Priority: Should
Como Gestor da plataforma de integração **Quero** Criar um ficheiro que realize o mapeamento entre os dados pretendidos e os dados que se encontram no modelo que desejo importar para que seja possível utilizar aquando a importação dos dados para diferentes modelos e origens.	

Os requisitos deste projeto são bastante dinâmicos e sofreram algumas influências por parte dos clientes envolvidos, proporcionando aos mesmos alterações recorrentes. Foram definidos vários requisitos que, apesar de interessantes e importantes para o futuro do projecto, têm menos relevância numa fase inicial do projeto, e, como tal, tem interesse que estejam documentados e planeados, mas o desenvolvimento dos mesmos não foi possível

ou não estava planeado no estágio. Esta é a principal razão da escolha do método de MoSCoW para a priorização dos requisitos já que o seu esquema tem em conta requisitos Could, que só devem ser implementados se não afetarem o resto do projeto, e Won't que devem ficar documentados, mas que só serão respondidos fora do contexto do estágio.

4.4 Atributos de Qualidade do Sistema

Antes de se prosseguir para o próximo capítulo, onde será apresentada a visão da arquitectura, serão mencionados nesta secção de que forma se pretendem cumprir certos atributos de qualidade e objectivos arquitecturais importantes que devem estar presentes para a construção do sistema. Os mesmos encontram-se descritos no anexo B. É ainda importante referir alguns requisitos que devem ser evidenciados e que servem de complemento. Requisitos não funcionais tais como usabilidade, que passam pela perceção da interface pelo utilizador ou facilidade de aprendizagem. O tipo de problemas associado pode ser ultrapassado pelos protótipos desenvolvidos e a utilização do produto. Os atributos de qualidade modularidade e modificabilidade devem também estar presentes durante o desenvolvimento, para que a inclusão de novas funcionalidades na aplicação, (relativamente à *framework* de *front-end* ou a um novo método de introdução de dados através da plataforma) sejam de simples e rápido desenvolvimento.

Capítulo 5

Arquitectura

Neste capítulo vai ser descrita a arquitectura do sistema existente, os objectivos, as tecnologias usadas, e as componentes nas quais o estagiário vai abordar.

A arquitectura do sistema de ambas as vertentes foi concebida de forma a ser o mais modular e *loosely coupled*. Ou seja permite flexibilidade e reutilização de código, de modo a adaptar-se às necessidades da empresa e dos seus clientes, permitindo que a adição e modificação de determinadas funcionalidades durante o desenvolvimento e posterior ao mesmo.

5.1 Descrição geral da arquitectura

A arquitectura presente na figura 5.1 enquadra-se no produto DrivianPro e representa a visão global da plataforma. A verde encontram-se os módulos nos quais existiu contribuição directa do estagiário.

Drivian Pro Web Client:

Representa a lógica aplicacional do lado do cliente, que realiza a comunicação directa com o administrador da empresa e que irá permitir ao mesmo realizar todas as funcionalidades de gestão e visualização. Neste componente o estagiário foi responsável por desenvolver uma *framework* de *front-end* que permita construir dinamicamente interfaces semelhantes entre os diferentes módulos, neste caso permitindo criar, de forma simplificada, listas que representam um conjunto de valores dependente do conteúdo que está a ser visualizado.

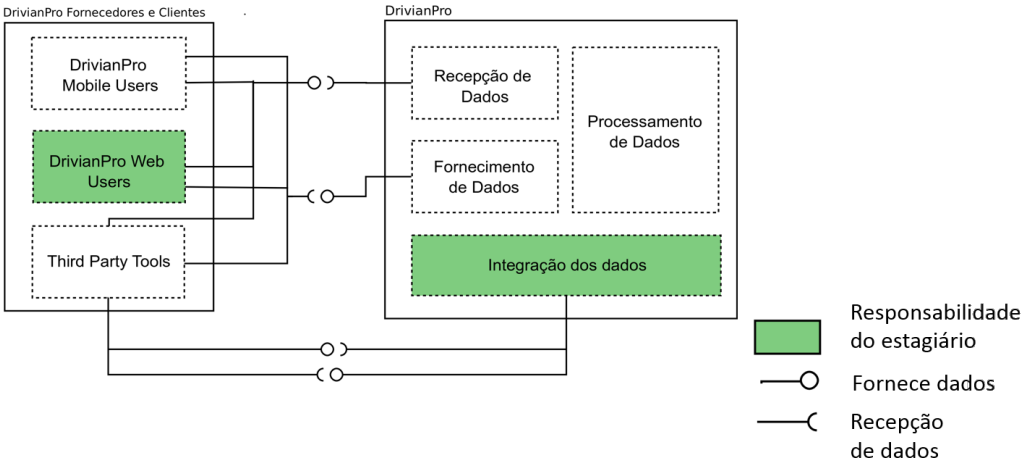


Figura 5.1: Perspectiva de alto nível

Drivian Pro BackEnd:

O *Backend* vai comunicar com ambas as vertentes, *frontEnd* e *Mobile*, e é responsável pela parte operacional e de *Business Intelligence* (BI) de todo o sistema. Neste módulo o estagiário realizou algumas das funcionalidades para realizar integração dos dados provenientes de diferentes plataformas ou fontes de dados.

Drivian Pro Mobile:

A vertente *Mobile* dedica-se na recolha, tratamento e transformação dos dados sensoriais provenientes de um *smartphone*, para consequentemente comunicar os dados para a componente de *Backend* da aplicação.

5.2 Visão Tecnológica

De seguida apresenta-se a visão tecnológica do sistema. Esta análise foi de extrema importância para o enquadramento no produto tal como detectar como seriam retratados alguns dos atributos de qualidade referidos anteriormente na secção 4.4.

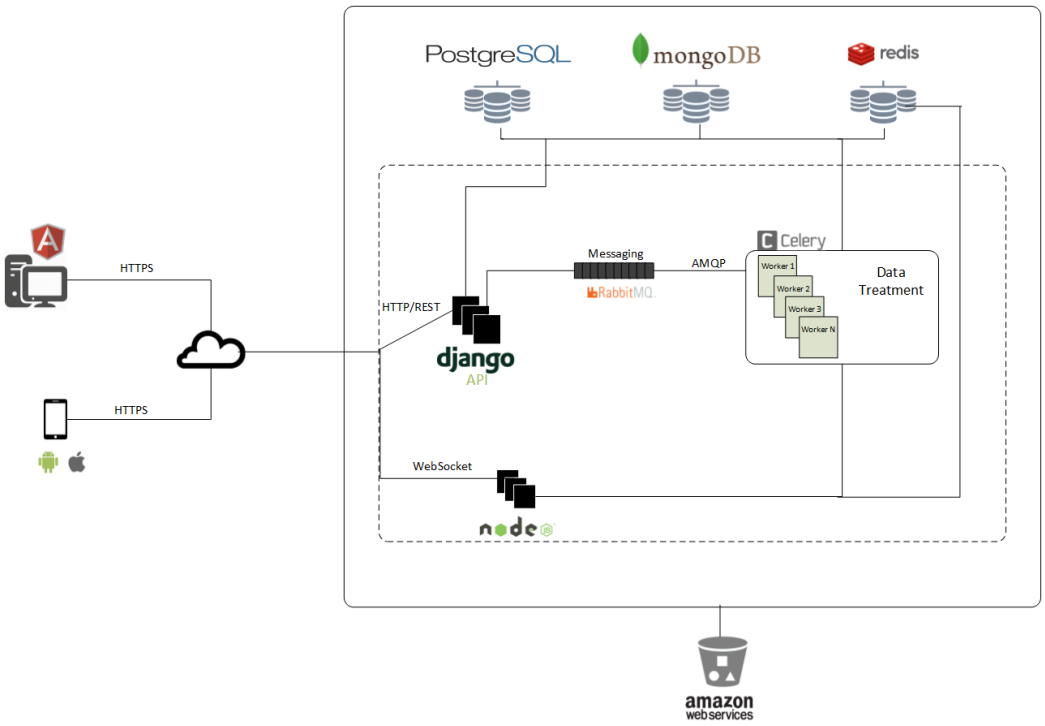


Figura 5.2: Visão Tecnológica do Sistema.

Cliente Web: Aplicação web, acedida através de um *browser*. Utilizada para visualizar os dados obtidos previamente analisados. No desenvolvimento desta aplicação, foi utilizado o *AngularJS* permitindo que a renderização seja realizada do lado do cliente, diminuindo o processamento do lado do servidor.

Cliente Mobile: Aplicação desenvolvida para os sistemas operacionais *android* e *iOS*. Têm como principais finalidades avaliar o modo de condução de cada utilizador, analisar e enviar dados sensoriais para o sistema usando serviços *REST* e recolher dados de cada viagem em tempo real enviando os dados através de *Websockets*.

API Server: Esta componente permite comunicar com ambas as aplicações, Web e Mobile e fornecer os recursos necessários às mesmas. É desenvolvido na linguagem Python com o recurso a uma *framework*, Django[1], que permite modularidade e flexibilidade nas aplicações construidas para o projecto, que em conjunto com a *framework*, Django REST Framework (DRF)

[1]https://www.djangoproject.com/

[2], permite simples construção de Web APIs. É neste módulo que são procedidos os pedidos provenientes da aplicação Web.

Celery Workers: Uma das principais vantagens no uso do Celery é a capacidade de criar *workers* que realizam o processamento de diferentes tarefas de forma assíncrona e independente, já que pode realizar o processo de diferentes mensagens em diferentes servidores, permitindo assim escalar sempre que necessário a aplicação. A escolha da execução das tarefas é dependente da ordem, tempo e a prioridade. Com o uso deste mecanismo foi possível criar um novo módulo para realizar o tratamento dos dados provenientes de cada mensagem.

PostgreSQL[3]: Base de dados relacional que permite armazenar os dados provenientes do sistema. Garante confiabilidade, replicação e integridade dos dados, e possui uma enorme comunidade. No âmbito do estágio este mecanismo foi utilizado para armazenar os dados relativos às importações realizadas.

MongoDB: Base de dados não relacional orientada a documentos. Permite flexibilidade, pois não exige a utilização de um esquema pré-definido com capacidade de suportar diferentes formatos de conteúdo. No âmbito do estágio é usado principalmente para armazenamento de documentos no formato *JavaScript Object Notation* (JSON).

Redis: Redis utiliza apenas a memória, não persistente e escalável, distribuído e rápido armazenamento chave-valor que garante performance, com mecanismos de publicação-subscrição.

RabbitMQ[4]: Solução de distribuição de *messaging* que implementa o protocolo *Advanced Message Queueing Protocol* (AMQP), que têm como principais características robustez, facilidade de uso, e suporte, que permite a possibilidade de definir as acções que uma determinada mensagem desempenha dentro do *broker*, com uma forte componente de escalabilidade pois fornece uma plataforma comum para enviar e receber mensagens para que estas sejam armazenadas com segurança de forma assíncrona até ao momento do envio e filas com alta disponibilidade que podem ser distribuídas por várias máquinas num agrupamento, assegurando que, no caso de uma falha, as

[2]http://www.django-rest-framework.org/
[3]http://www.postgresql.org/
[4]https://www.rabbitmq.com/

mensagens estão seguras.

Amazon S3: Ficheiros que foram alojados num sistema externo, como a Amazon S3 [5] (Amazon Simple Storage Service) que pertence aos AWS (Amazon Web Services). Mais concretamente, os ficheiros criados que funcionam como modelo são gravados em buckets criados explicitamente na Amazon S3.

5.3 Contribuições para a arquitectura geral

Durante o percurso do estágio o aluno esteve envolvido em duas áreas de foco distintas. Numa fase inicial o desenvolvimento de uma *framework* que têm como principal função implementar a camada de lógica e visualização do lado do cliente. Posteriormente a implementação de uma plataforma como conector de dados, que permita ligação e extração de dados de diversas fontes, que comunique os dados tratados e preparados com o servidor para serem posteriormente armazenados.

5.3.1 Front-End Framework

:

Nesta secção será realizada uma reflexão sobre a arquitectura da plataforma a ser implementada.

Controladores: Permitem adicionar funcionalidades extra ao *scope* da *view* em que se encontra, e fazem a comunicação com as directivas. São usados para controlar o fluxo de informação na aplicação.

Directivas: Funções que manipulam e adicionam comportamento a elementos do DOM. É possível usar múltiplas vezes uma directiva, e o *AngularJS* já possui de base algumas, tais como *ng-repeat* ou *ng-model*. Durante o trabalho do estagiário estas são extremamente importantes devido a ser possível construir *custom directives* que permitem ao programador originar as próprias directivas, que podem ser usadas em diversos contextos por qualquer controlador e manter o código mais organizado, sustentável e reutilizável.

Views: Possuem o código *HyperText Markup Language* (HTML) e transformam os dados que são apresentados.

[5] http://aws.amazon.com/s3/ - Amazon S3 fornece um serviço web simples, altamente escalavel, seguro e estavel que pode ser usado para guardar e retirar dados em qualquer altura e em qualquer lugar

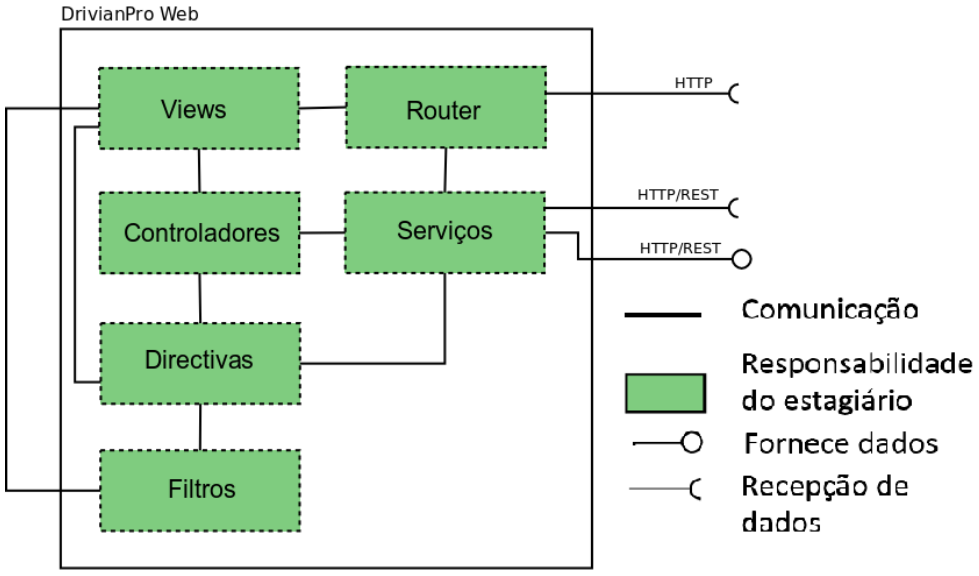


Figura 5.3: Drivian Pro

Serviços: Permite manter informação durante o tempo de vida da aplicação e comunica com os controladores de maneira consistente, são objectos *singleton* existe só uma instância do mesmo objecto.

Router: Biblioteca que permite organizar a interface como uma máquina de estados.

Para este modelo ser mais facilmente executado, foi usada uma *framework* desenvolvida em *Javascript*, *AngularJS*[6], para desenvolvimento de *front-end web*, *client-side*, que permite velocidade, segurança e reutilização de código, extensível e modular, usada para escrever aplicações web dinâmicas. Implementa a *design pattern Model View Controller* (MVC) permitindo que exista uma separação entre lógica aplicacional e as *views*. Permite também a reutilização de componentes ao longo do desenvolvimento de toda a aplicação, através da criação de novas directivas. Com a utilização de uma *framework* como o angular para processamento de código do lado do cliente, traduz-se numa diminuição da carga do lado do servidor, uma vez que este não necessita de realizar todo o processamento e posteriormente enviar para o cliente.

[6]https://angularjs.org

5.3.2 Plataforma para Integração de dados

Nesta secção é especificado o módulo de integração de dados no qual é referido a recepção, a análise e a validação dos dados.

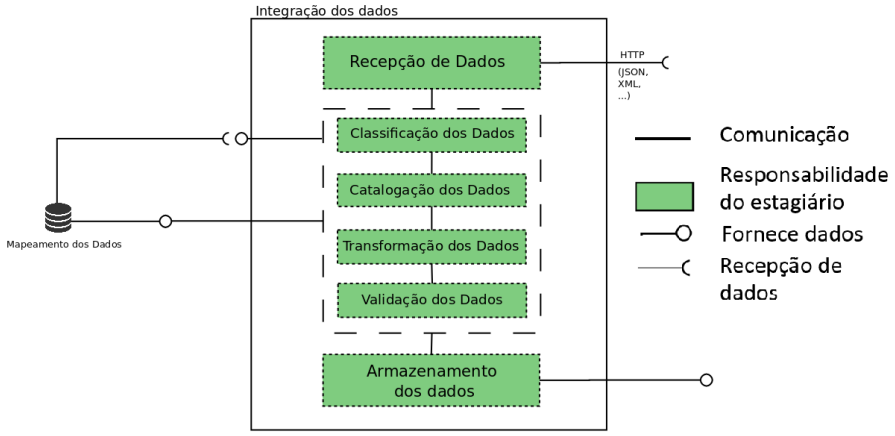


Figura 5.4: Tratamento dos Dados

Para a realização do tratamento e análise dos dados, o processo é dividido em quatro fases diferentes tal como descrito na figura 5.4.

Classificação dos Dados: Neste módulo os dados são inicialmente recolhidos e convertidos para um formato único, neste caso *JSON*, devido a representar objectos de forma simplificada. Esta abordagem permite a consistência das etapas seguintes já que o formato do ficheiro vai ser idêntico para os seguintes componentes.
Catalogação dos Dados: Através da base de dados, é possível obter um documento no qual se encontra informação sobre o mapeamento a ser realizado. Com esta informação é analisado os dados provenientes de uma fonte externa e mapeados para um novo documento com todos os objectos devidamente identificados.

Transformação dos Dados: Depois dos dados devidamente identificados, é feito uma análise e correcção dos objectos criados dependentes do modelo de dados e do tipo de dados que cada representa.

Validação dos Dados: Tem a principal função de validar os dados através de análise a erros sintáticos ou semânticos e se estes detectam algum erro aquando a inserção. É durante esta fase que é construido um *log* dos erros,

ou seja um registo que possui informação sobre o conteúdo do erro e a sua localização.

Armazenamento dos Dados: A inserção dos dados é realizada aquando o processamento das etapas anteriores e actualiza ou insere os dados se não existir qualquer erro decorrido na validação dos dados. Depois desta etapa o utilizador recebe uma notificação por *e-mail* através de *Simple Mail Transfer Protocol* (SMTP) com o resultado do armazenamento dos dados em conjunto, se necessário, os *logs* com os erros.

Mapeamento dos Dados

O processo que realiza o mapeamento dos dados implica a tradução de conteúdos num formato não perceptível ao sistema para dados compreendidos pelo mesmo. Estes dados são guardados numa base de dados não relacional, MongoDB, que permite guardar documentos que contém pares chave-valor, nos quais se mapeia a base de dados da organização com os valores obtidos de uma fonte de dados externa.

Através da figura 5.5 é possível compreender a inserção de um ficheiro que vai ser usado para mapeamento.

Depois de introduzir todos os dados para realizar o mapeamento, a aplicação web trata de validar através do conteúdo, e cria o ficheiro. Através de um POST feito à API, o ficheiro é devidamente enviado para o sistema que valida o mesmo e insere no MongoDB, que devolve posteriormente ao utilizador o resultado da operação.

5.4 Decisões arquitecturais

Seguem-se algumas das decisões que devem ser consideradas durante a implementação do módulo.

MVC

É um padrão de software dividido em três componentes principais. O *Model* é usado para definir e representar a lógica da estrutura de dados numa aplicação. A *View* gere a forma da apresentação da informação enquanto que o *Controller* interpreta e analisa os dados de entrada e comunica a mensagem a ser enviada.

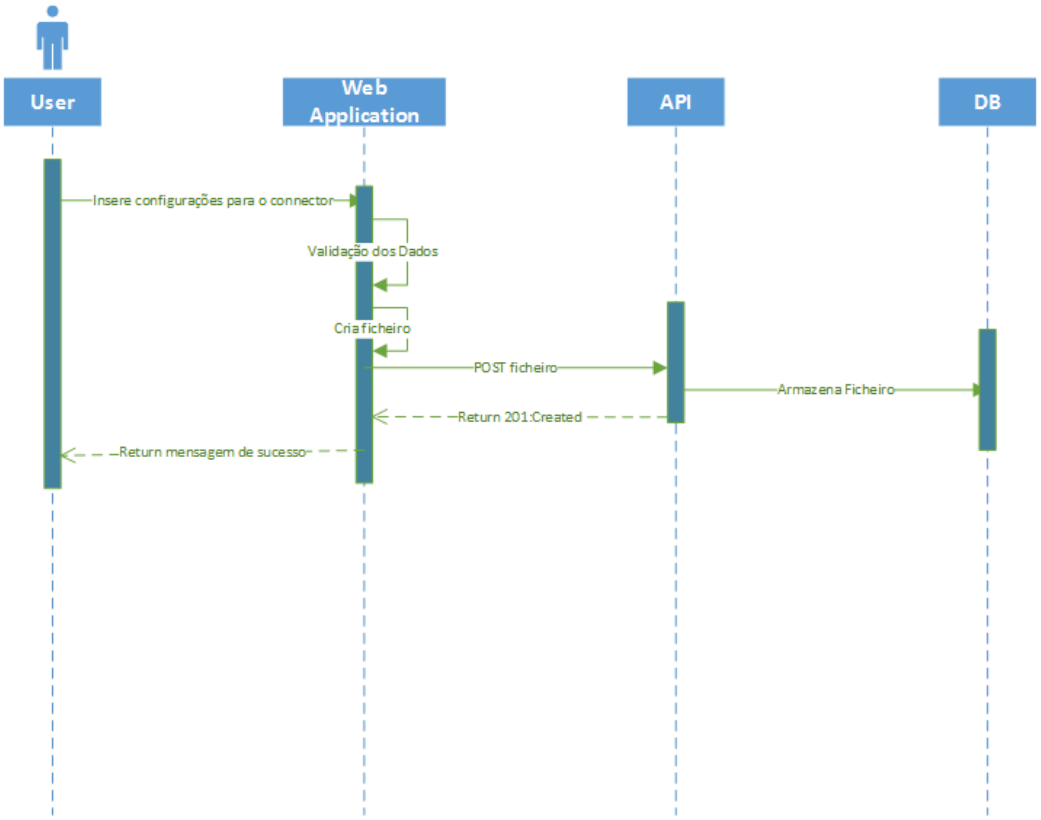


Figura 5.5: Inserção de Ficheiro para Mapeamento

Client side rendering

O uso de Angular significa que o javascript que está a correr no *browser* produz HTML e manipula o *Document Object Model* (DOM). Uma das grandes vantagens é que a informação é actualizada automaticamente com os dados que tem do lado do cliente, em vez de esperar o pedido ao servidor como é feito com o server-side rendering. Com esta técnica é possível aumentar a performance do servidor, pois o tráfico entre cliente e servidor é reduzido, e com optimizações bem definidas como *caching* que permite aumentar a performance do servidor. Algumas das desvantagens passam por renderização da página que é mais lenta devido ao trabalho extra realizado na manipulação do DOM e compatibilidade com *legacy browsers* se usada uma framework como o *AngularJS*.

REST

REST é um mecanismo que define um conjunto de restrições que, quando aplicadas à arquitetura de um sistema distribuído, permitem características importantes tais como baixo acoplamento e escalabilidade horizontal [37] com vantagens tais como menor transmissão de pacotes e flexibilidade comparativamente com outros mecanismos.

Messaging

A utilização de uma fila de mensagens para gerir os pedidos possibilita a execução de tarefas assincronamente e interdependência dos serviços que elimina o tempo de espera até ao final da execução da tarefa. Este tipo de mecanismos simplifica o trabalho de dividir entre diferentes máquinas, aumentando assim a escalabilidade do sistema e permitem no caso de indisponibilidade do sistema, garantir que todos os pedidos são armazenados e executados assim que o sistema se encontra disponível. Para garantir que todos os pedidos são executados e como existem tarefas que possuem muitos dados é necessário introduzir um novo *worker* para realizar o processamento dos dados e garantir a escalabilidade do produto.

Capítulo 6

Implementação

Nesta secção são explicados os detalhes sobre a implementação do projecto. Esta divide-se entre a implementação da *framework*, onde são apresentados os resultados do seu desenvolvimento e a plataforma de integração de dados através de fontes de dados externas.

6.1 Front end Framework

6.1.1 Desenvolvimento

A estrutura de desenvolvimento deste módulo foi baseado na arquitectura do AngularJS como explicado no capítulo anterior. Neste caso, o que acontece é que os controladores possuem informação estática sobre a informação que deve ser transmitida às directivas e todos os métodos que devem ser usados posteriormente. Consequentemente as directivas personalizadas permitem manipular o DOM e os elementos correspondentes. Ou seja, aquilo que vai ser visualizado pelo utilizador, neste caso as vistas, podem ser construídas de forma dinâmica já que vão ter o seu conteúdo manipulado pelas directivas e realizam comunicação bidirecional com os controladores já instanciados.

Com o uso da técnica *Two-Way Data Binding*, permitida pelo AngularJS, a vista é uma projecção do modelo, que quando este altera, a vista reflecte as alterações. O mesmo efeito acontece caso seja a vista a ser alterada.

As directivas criadas, que se compreendem como componentes que são reutilizados, permitem modularidade no desenvolvimento. Como estas são genéricas e independentes, sempre que necessário criar um novo módulo, neste caso uma nova listagem de dados, apenas é necessário construir a camada lógica, ou seja, os controladores.

6.1.2 API Requests

Cada módulo, que neste caso se traduz em cada listagem de dados, possui um serviço associado, que permite realizar os pedidos para obtenção e envio dos dados, e contém todas as chamadas necessárias que permitem executar as funções REST. Os métodos usados nos pedidos para a API permitiram obtenção de um ou mais recursos, GET, envio de um novo recurso, POST, actualização de um recurso, PATCH, ou eliminar um recurso, DELETE. O servidor processa os pedidos e para cada acção realiza uma resposta.

Relativamente a autenticação, é enviado, associado a cada pedido, um *access token*, criado no momento do *login* do utilizador, que através do protocolo usado, permite garantir a autenticidade dos dados.

6.1.3 Resultados

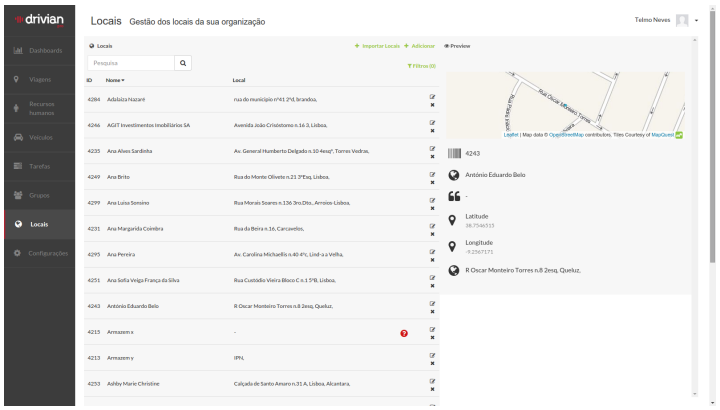
Nesta secção pretende-se apresentar os resultados do trabalho desenvolvido na *framework*.

Como é possível visionar na figura 6.1 as diferentes listagens de dados actualmente desenvolvidas, relativamente a características visuais e funcionais, são bastante semelhantes. As funcionalidades existentes são bastante similares entre módulos, e é possível definir nos controladores caso uma funcionalidade não seja pretendida, permanecendo em todos os outros módulos.

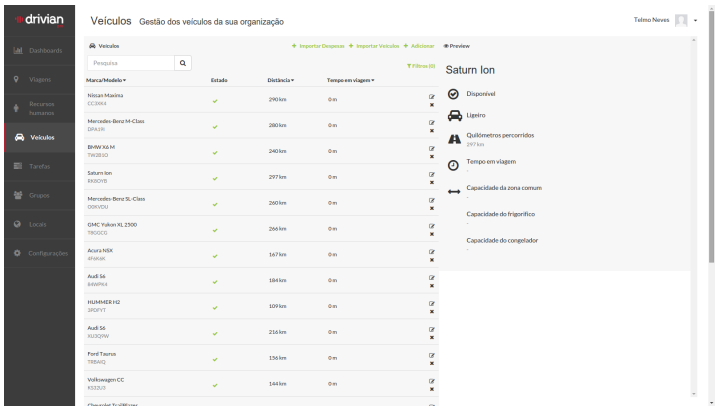
Através da figura 6.2 são detalhadas as funcionalidades presentes.

Uma das funcionalidades presentes é a listagem dos dados, que permite visualizar todas as tarefas existentes para aquela organização. Cada lista é composta por um *header* onde é visualizado o significado de cada campo presente em cada tarefa. Cada campo possui atributos diferentes, como texto ou ícones, e se definido possui ordenamento dos dados. Cada tarefa possui também um *preview* que permite obter informações mais detalhadas. Estas funcionalidades estão presentes em todos as listagens de dados devido à modularidade das directivas criadas. No desenvolvimento deste módulo foi necessário não só realizar a *framework* mas também os *templates* necessários e a dificuldade acrescentada pelo mesmo.

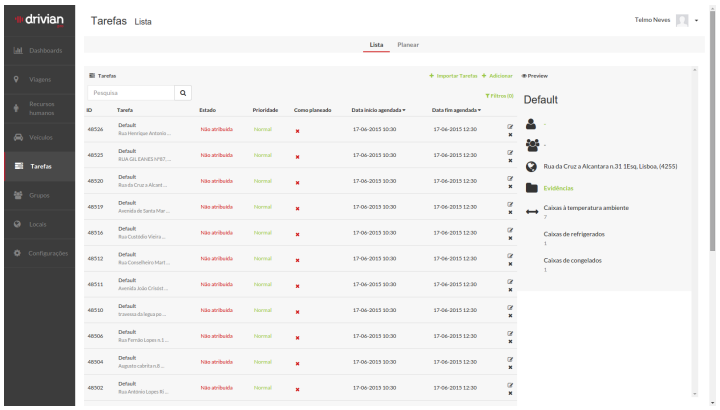
Quando o utilizador actualiza a página, é feito um pedido ao servidor para obtenção dos dados, neste caso um pedido GET, com parâmetros adicionais, tais como pesquisa, ordem ou filtros, caso existam, que permitem reconstruir a página e apresentar uma nova lista com a informação obtida do servidor. Se o utilizador fizer *scroll* sobre a página e caso existam mais recursos, neste caso tarefas como se encontra na figura 6.2, é feito um novo pedido GET, com as mesmas características do pedido anterior à excepção da página, que permite



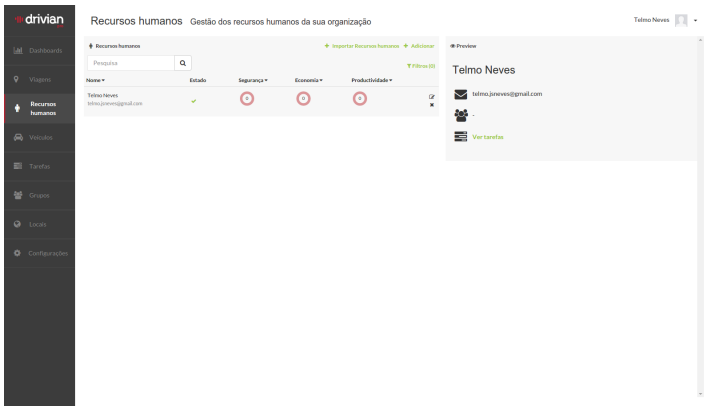
(a) Lista de Locais



(b) Lista de Veículos



(c) Lista de Tarefas.



(d) Lista de Utilizadores.

Figura 6.1: Diferentes Listagens.

ir buscar os dados da página seguinte. A lista é actualizada automaticamente, permanecendo os dados dos restantes pedidos. Sempre que existe um destes pedidos, é obtida a informação de cada recurso, ficando este guardado e que permite construir o *preview* respectivo.

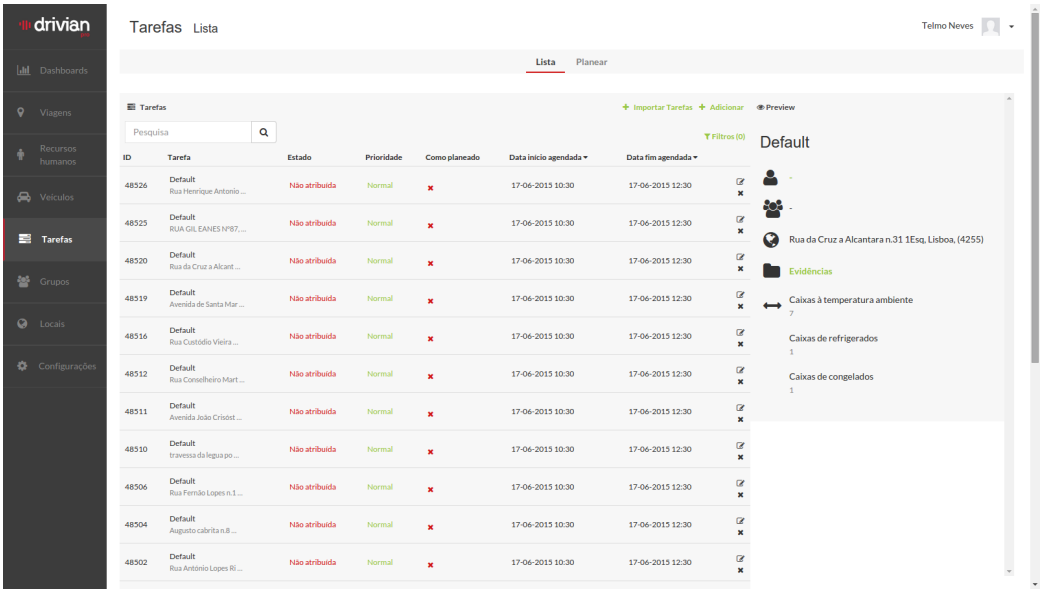


Figura 6.2: Listagem de Tarefas.

6.2 Plataforma de Integração de Dados

Na secção seguinte apresenta-se o desenvolvimento da plataforma de integração de dados.

6.2.1 Integração de Dados

Como referido, o módulo desenvolvido para efectuar o tratamento e mapeamento dos dados está presente num *worker* que se encontra desenvolvido com recurso à ferramenta *Celery*. Deste modo foi possível separar a componente lógica e tratar os pedidos provenientes da fila de mensagens paralelamente com a restante aplicação.

Relativamente à validação dos erros obtidos, estes são verificados de dois diferentes métodos. O primeiro tende a validar algumas informações através de análise dos campos e se estes comprometem a estrutura de dados. Consequentemente são avaliados os erros que são obtidos aquando o armazenamento dos dados na base de dados na qual todas as alterações são realizadas, ou nenhuma.

Para notificar o utilizador, é enviado um email ao mesmo que contém o resultado em conjunto com o *log* de erros gerado anteriormente.

6.2.2 Leitura e Integração de ficheiros

Como havia sido descrito na arquitectura do sistema, foi feita a integração com documentos. Esta integração inclui a leitura de ficheiros de diversas fontes tais como XML, CSV e XLS. Para realizar a leitura destes formatos, foi inicialmente utilizada uma biblioteca [1] que permitia ler e converter diferentes formatos de forma automática. Como esta biblioteca carecia de algumas funcionalidades, e como seria desejável ter maior controlo, foi desenvolvida uma nova ferramenta, que possibilitaria a leitura dos diferentes ficheiros e conversão dos dados em um formato único. Este ficheiro é fornecido pela aplicação ou criado pelo utilizador, que modifica o mesmo e envia para o sistema. O envio do mesmo é feito através de um pedido POST à API, dependente do modelo onde deve ser injectado a informação.

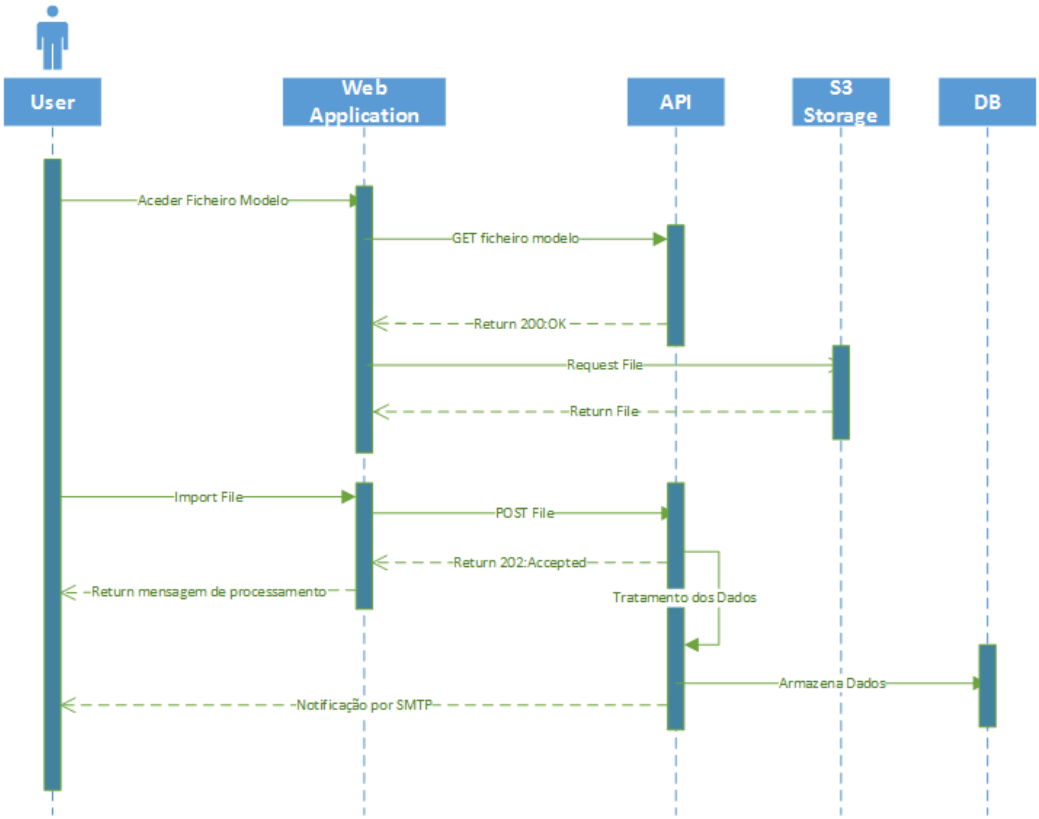


Figura 6.3: Inserção dos dados.

[1] http://okfnlabs.org/dataconverters/ - Biblioteca desenvolvida em python que permite conversão de ficheiros.

No diagrama 6.3 são mostradas duas funcionalidades distintas, o método de obtenção de um ficheiro modelo, e o método de envio de um ficheiro para ser analisado e adicionar novos dados. No primeiro método, o utilizador obtém através da visualização na página web de uma opção para obter o ficheiro modelo. Através de um pedido GET, realizado à API e que se correctamente executado responde com o estado 200 e o url associado. A aplicação web acede ao link e como resposta é retornado um ficheiro ao qual o utilizador pode aceder. No segundo método, o utilizador decide importar um ficheiro para adicionar novos dados à base de dados. Neste caso o utilizador seleciona um ficheiro, que se encontra localmente na sua máquina, em que a aplicação web envia um pedido com o ficheiro associado. Se o ficheiro for obtido com sucesso é retornado um código representativo de ficheiro aceite e o utilizador é notificado de tal. É realizado do lado do servidor o tratamento dos dados obtidos do conteúdo do ficheiro, tal como explicado na secção 5.3.2, realizando posteriormente o armazenamento dos dados na base de dados. Após esta etapa o utilizador é notificado por email do processo e se os dados foram realizados com sucesso.

6.2.3 Integração com outros Sistemas Externos

Como havia sido referido anteriormente, foi considerado outras fontes de dados, que seriam relevantes para o produto. De seguida explica-se como este processo se encontra estruturado.

Conectores para Base de Dados

Relativamente ao diagrama encontrado na figura 6.4 existem duas funcionalidades, a configuração de uma base de dados e consequentemente a conexão para armazenamento de dados. Para a primeira fase o utilizador ou um gestor da plataforma, através da página web configuram uma conexão com uma das base de dados específica para aquela configuração. A aplicação web trata de examinar os dados, para confirmar a validade dos dados, realizando um pedido POST com os dados. Os dados são enviados para uma aplicação *standalone* para garantir a modularidade da mesma. Esta aplicação cria o conector realizando posteriormente a conexão retornando o estado da mesma. Se a conexão for realizada com sucesso é guardado o registo da mesma e o utilizador é notificado. A segunda fase do processo compreende a necessidade de existir uma conexão previamente criada. Inicialmente é obtido, através de um pedido GET, a configuração a uma das base de dados que o utilizador selecionou. O utilizador recebe a confirmação dos dados e envia os dados

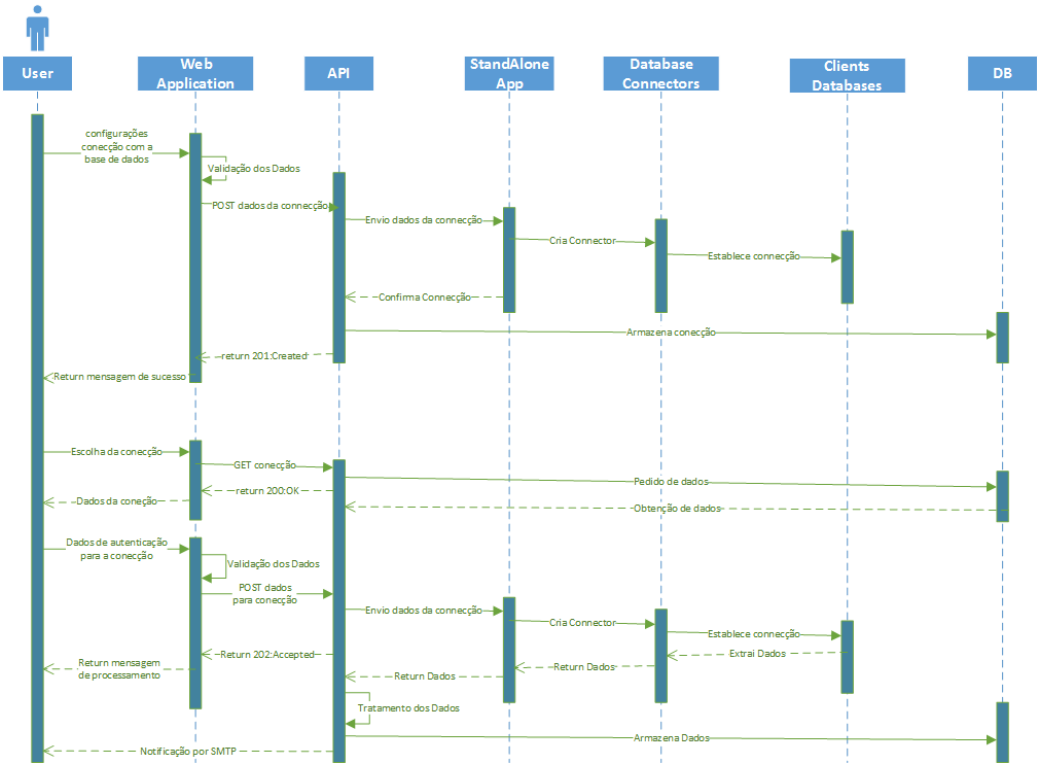


Figura 6.4: Configuração e uso de uma conexão com uma base de dados.

para realizar a autenticação, enviando os dados para a aplicação *standalone* para estabelecer a comunicação com a base de dados. A aplicação extrai os dados e retorna os mesmos. Durante este processo o utilizador é notificado que o processamento dos dados vai ser realizado. Depois de obter os dados é realizado o tratamento dos dados e armazenados os mesmos na base de dados. o utilizador é notificado por email do sucesso ou insucesso deste procedimento.

Integração de Dados através do consumo de serviços REST

Neste anexo encontra-se estruturado o trabalho desenvolvido referente à importação de dados através de uma API REST.

Através do diagrama 6.5 é possivel entender o fluxo existente, e retirar três diferentes funcionalidades com diferentes tipos de autenticação, para o efeito de obtenção de dados:

HTTP Basic Authentication, em que é fornecido apenas o nome do utilizador e a password.
API key em que o pedido enviado tem em conjunto uma chave que permite realizar o accesso.
oAuth 2.0 O acesso à API para obter os dados só é permitido quando o *user agent* já realizou autenticação e possui um *access token*.

Para qualquer um dos métodos referidos, o tratamento de dados têm um comportamento semelhante e à excepção de obtenção de dados através do protocolo *oAuth 2.0* todos os outros métodos são bastante simples, necessitando apenas de uma chamada à API na qual se pretende obter os dados.

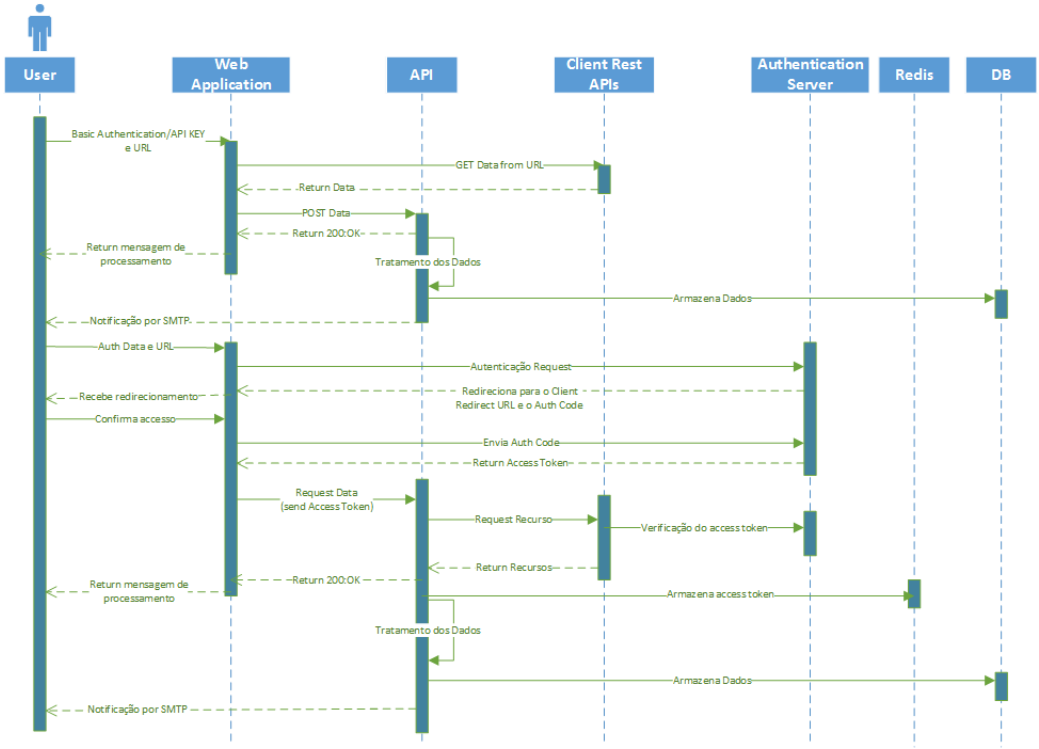


Figura 6.5: inserção de dados através da obtenção de dados localizados em uma API REST.

Autorização e Autenticação oAuth 2.0

Nesta secção encontra-se o estudo realizado sobre o protocolo oAuth 2.0.

No diagrama 6.6 encontra-se a interacção genérica entre si. Inicialmente a aplicação pede autorização ao utilizador para aceder aos recursos. A aplicação, depois de concedida a autorização pelo utilizador, realiza um pedido para o servidor que fornece a autorização com a autorização obtida na fase anterior e autenticação da aplicação, devolvendo posteriormente um *access token*, que se válido, o servidor de recursos envia os recursos para a aplicação. Este modelo é definido por quatro papéis, o *utilizador*, ou *resource owner*, que autoriza a aplicação para aceder à sua conta, o *Servidor de Autorização* e o *Servidor de Recursos*, que se encontram por norma na mesma máquina, e que respectivamente verificam a identidade e fornecem os recursos. A aplicação é o cliente que pretende aceder à conta do utilizador e obter os recursos.

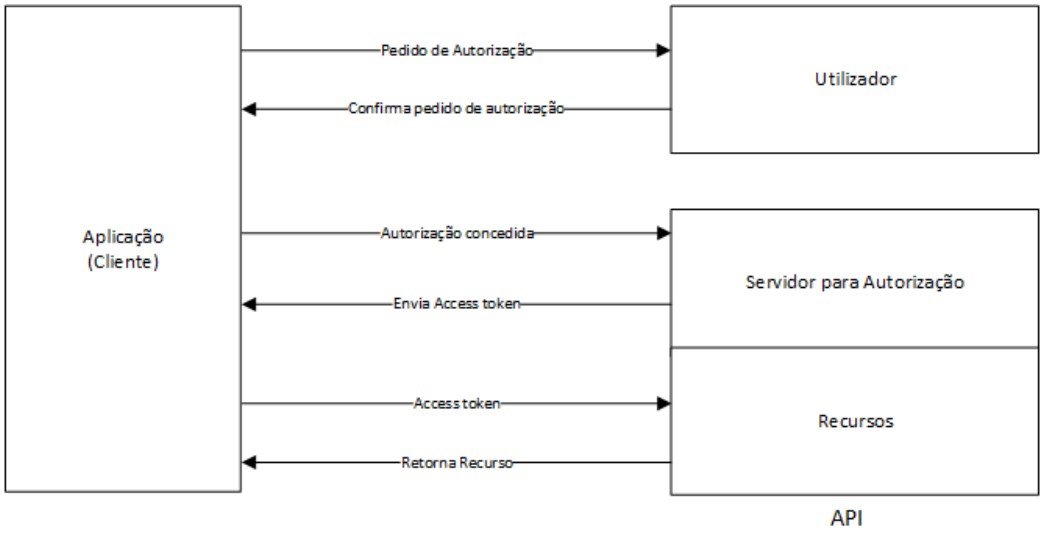


Figura 6.6: Abstração do método de Autenticação.

De seguida apresentam-se como algumas aplicações solicitam e obtém a autorização e o *access token*, dependente do tipo de autenticação concedido suportado pela API.

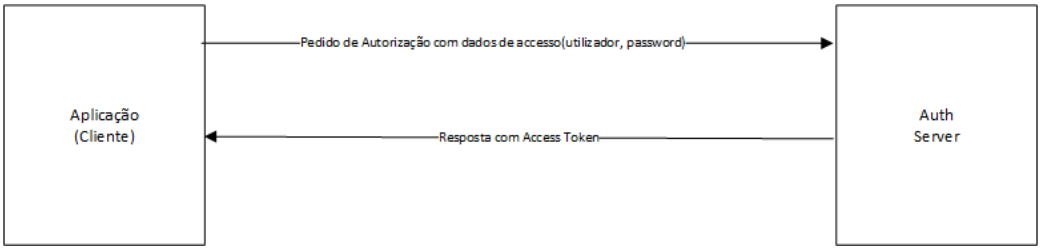


Figura 6.7: Tipo de permissão : Credenciais

O método encontrado na figura 6.2.3 é o mais simples de implementar, e normalmente usado com aplicações de confiança, tais como as possuídas pelo próprio serviço.

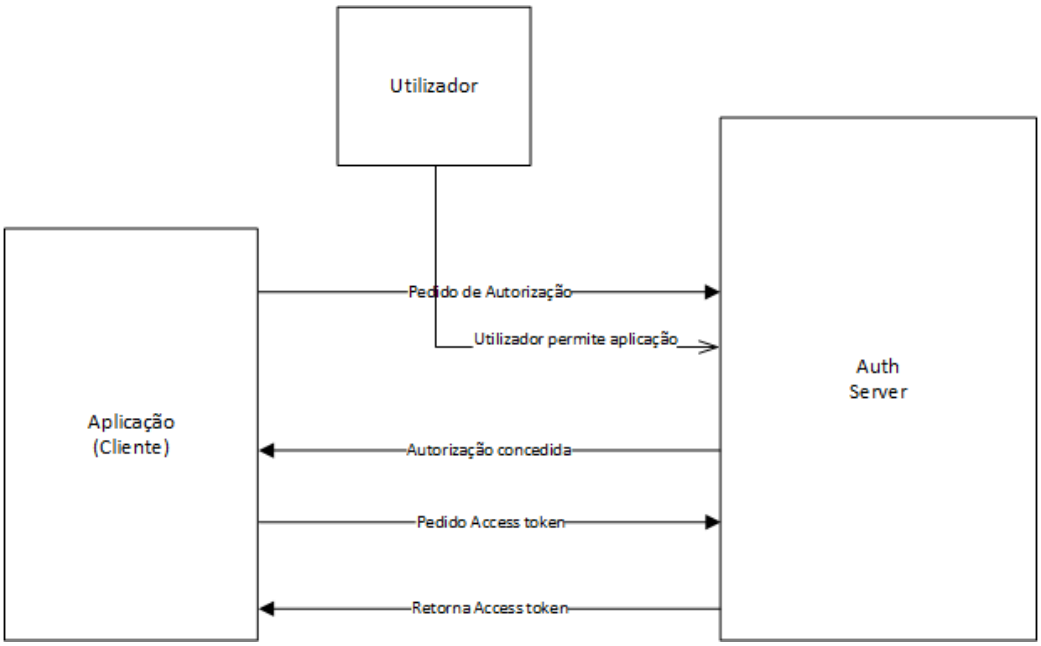


Figura 6.8: Tipo de permissão : Código de Autorização

A abordagem encontrada na figura 6.8 é usualmente seguida por aplicações *server-side*, que possui um fluxo baseado em redirecionamento, ou seja que a aplicação é capaz de interagir com o *user-agent* que recebe os códigos de autorização da API.

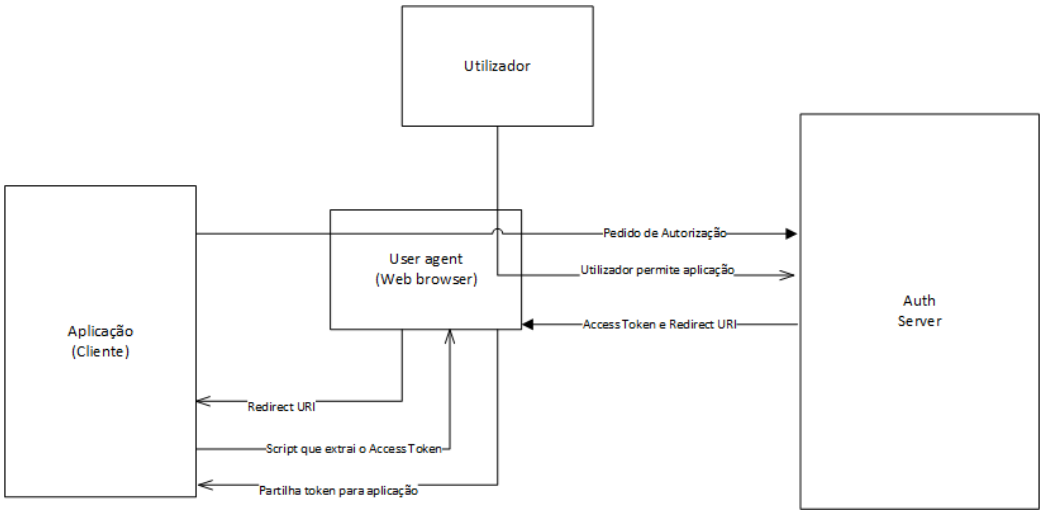


Figura 6.9: Tipo de permissão : Implícito

O método encontrado na figura 6.9 é normalmente utilizado por aplicações Web ou mobile, na qual não é possível garantir a confidencialidade da chave do cliente. Esta abordagem não suporta *refresh tokens*.

Este módulo tal como a obtenção de dados através de uma fonte de dados fornecida por uma REST API, encontram-se parcialmente implementados, já que apesar de desenvolvidos, não se encontram devidamente integrados no produto devido a decisões de negócio e planeamento.

Em ambos os módulos, e independentemente da fonte de dados que seja utilizada, o processo de tratamento dos dados é executado identicamente, já que os dados são convertidos para um formato único que permite facilmente introduzir novos módulos a esta componente.

6.2.4 Resultados

Posteriormente apresenta-se os resultados relativamente à plataforma de integração de dados.

Através da figura 6.10 é possível visionar as principais funcionalidades que se encontram actualmente na plataforma.

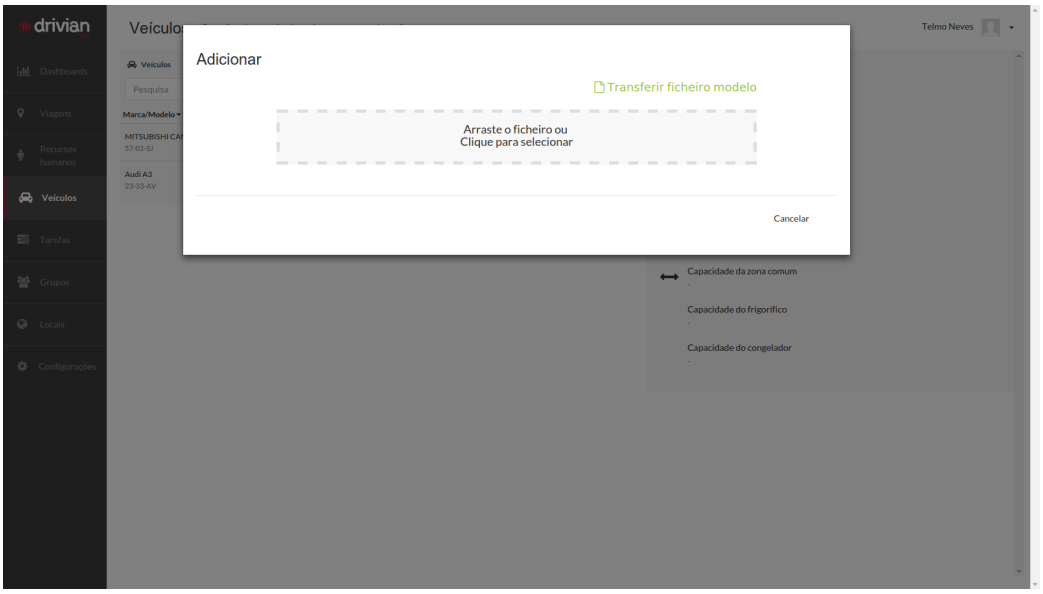


Figura 6.10: Inserção do ficheiro.

Inicialmente o utilizador tem a hipótese de extrair um ficheiro que funciona como modelo para a inserção dos dados. Este documento encontra-se num *bucket* na Amazon e quando clicado é feito automaticamente o download do ficheiro para a máquina do utilizador. Este ficheiro, no formato xls, é composto por um *header* no qual cada coluna tem informação associada sobre os dados que devem ser preenchidos. Depois do preenchimento deste ficheiro, ou se o utilizador já possuir um ficheiro com as mesmas características, pode realizar a importação do mesmo. Depois do ficheiro ser enviado o utilizador é notificado que o processamento de dados se encontra a ser realizado.

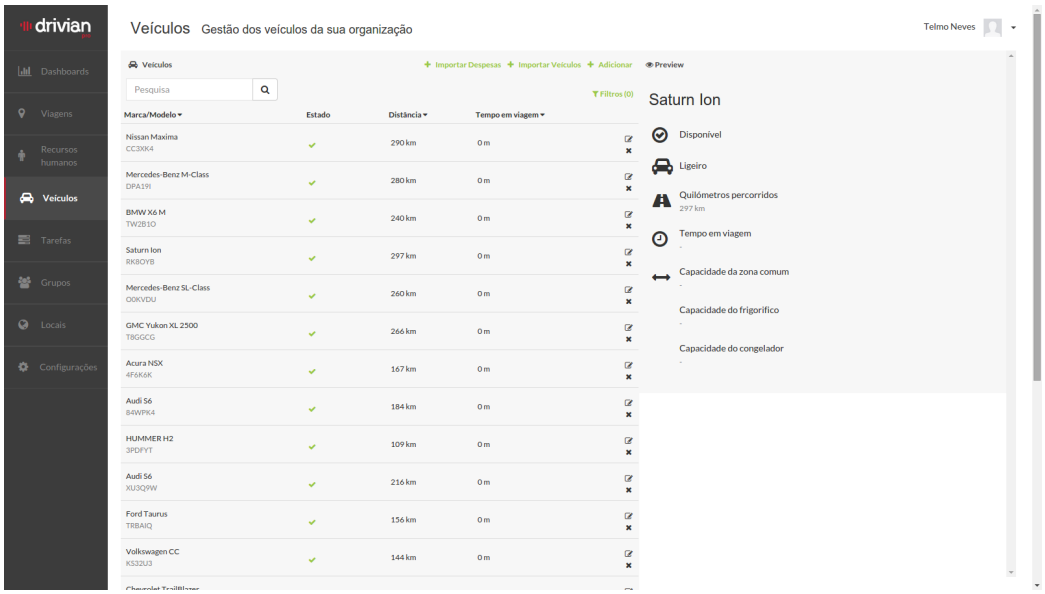


Figura 6.11: Listagem de dados depois da importação de dados.

Quando este processo termina o utilizador é notificado por e-mail e se o processamento de dados ocorrer como o pretendido, quando actualiza a página Web encontra os dados actualizados com o ficheiro submetido, tal como se encontra na figura 6.11. A partir deste momento os dados podem ser acedidos e manipulados pelas organizações, não só para explorar as funcionalidades de visualização, mas também usar nas restantes funcionalidades do produto.

6.3 Testes

Relativamente à validação do trabalho, foram desenvolvidos testes funcionais baseados nas *user stories* para avaliar o comportamento da aplicação. Estes foram testados durante o desenvolvimento de ambas as ferramentas, inicialmente pelo estagiário, e posteriormente pela equipa de desenvolvimento e *stakeholders*.

Relativamente à usabilidade do produto, foram utilizados os protótipos para enquadrar com a equipa de desenvolvimento e descobrir o que era mais importante a nível visual e funcional. Estes permitiram enquadrar com o produto e manter o mesmo padrão de visualização.

Neste momento as funcionalidades implementadas encontram-se no produto e o mesmo já esteve presente em demos, nos quais foi apresentado as

funcionalidades do mesmo e encontra-se actualmente no ambiente de produção no qual já existem clientes a usar o produto.

Capítulo 7

Conclusões

O trabalho desenvolvido ao longo do ano lectivo resultou num produto que acrescenta valor à empresa e ao produto DrivianPro. Deriva-se do facto que o trabalho desenvolvido se encontra no produto e que permite facilitar o trabalho e melhorar o desempenho das organizações que utilizam o produto. Devido à flexibilidade do produto e do trabalho desenvolvido, é possível incorporar algumas das funcionalidades em futuros produtos da empresa.

7.1 Contribuições

Como referido anteriormente o estagiário foi responsável, inicialmente por desenvolver uma *framework front-end* que permite a visualização e manipulação de dados através da criação de interfaces Web. Numa segunda iteração o estagiário esteve envolvido numa plataforma para integração de dados, capaz de conectar a diferentes fontes de dados, e integrar os mesmos na plataforma. Em ambas as plataformas o estagiário teve de implementar e realizar a arquitectura, em conjunto com a equipa, para compreender e realizar o trabalho da melhor maneira possível. O trabalho desenvolvido pelo estagiário encontra-se actualmente no produto.

Apesar de não ter sido possível implementar todas as funcionalidades desejadas, o trabalho desenvolvido, bem como o estudo e a arquitectura, beneficiam o trabalho futuro a ser implementado.

7.2 Balanço

A realização deste estágio foi de extrema importância para o estagiário, não só relativamente a conhecimentos de Engenharia Informática e Engenharia de Software, mas também o uso de metodologias ágeis e o trabalho em equipa, que são constantemente usadas em ambiente profissional e nunca antes aprofundadas pelo estagiário. O ambiente no qual o estagiário esteve presente, uma Startup, permitiu ganhar conhecimentos técnicos e uma excelente forma de aprendizagem de como estas funcionam e como devem estar estruturadas. Apesar de existirem algumas dificuldades, que levaram a que algumas funcionalidades não estivessem devidamente implementadas e integradas no produto, serviram como aprendizagem e experiência para o estagiário.Contudo é importante referir que as funcionalidades desenvolvidas se encontram actualmente no produto e usado por clientes.

7.3 Trabalho Futuro

Relativamente ao trabalho a desenvolver no futuro, e como referido anteriormente, é importante acrescentar funcionalidades relativamente a integração de novos serviços, tais como *web services*, para permitir comunicar com CRMs e ERPs. Apesar de necessário realizar algumas actualizações, como o sistema de análise, tratamento e transformação de dados se encontra realizado, o acrescento de novas funcionalidades é mais simplificado e desta forma o desenvolvimento efectuado pode tornar-se mais útil dentro da empresa e para os seus clientes. É importante desenvolver as funcionalidades de *front end* para que os gestores possam alterar e personalizar os ficheiros que realizam o mapeamento tal como se encontra na figura4.6. O desenvolvimento e arquitectura deste módulo têm de estar preparados para compreender as necessidades de cada organização.

Apêndice A

User Stories

Apêndice B

Atributos de Qualidade do Sistema

Bibliografia

[1] drivian. drivian. http://goo.gl/FVh6jX. Acedido em 21 de Janeiro de 2015.

[2] K Schwaber and Jeff Sutherland. The scrum guide. *Scrum. org, October*, 2(July):17, 2011.

[3] Rui Chicória. Real-time architecture and workflow for the One.Stop.Transport platform.

[4] Henrik Kniberg. *Lean from the Trenches: Managing Large-Scale Projects with Kanban*. Pragmatic Bookshelf, 2011.

[5] a Cockburn and L Williams. The costs and benefits of pair programming. *eXtreme Programming and Flexible Processes in Software Engineering–XP2000*, pages 33–42, 2000.

[6] Git. Git. http://goo.gl/z8R0Sj. Acedido em 17 de Janeiro de 2015.

[7] GitLab. GitLab. http://goo.gl/GQ6VHT. Acedido em 17 de Janeiro de 2015.

[8] Vincent Driessen. A successful Git branching model. http://goo.gl/P8XWWe. Acedido em 17 de Janeiro de 2015.

[9] Wj Stevenson and M Hojati. *Operations management*. 2007.

[10] In4Tools. In4Tools. http://goo.gl/NyKYol. Acedido em 9 de Janeiro de 2015.

[11] In4Tools. 4Biz. http://goo.gl/DVeICI. Acedido em 9 de Janeiro de 2015.

[12] Jobber. Jobber. http://goo.gl/uyBmZ8. Acedido em 11 de Janeiro de 2015.

[13] jobber. blog.getjobber. https://goo.gl/VPyiGO. Acedido em 28 de Agosto de 2015.

[14] FieldAware. FieldAware. http://goo.gl/Mmlqhf. Acedido em 15 de Agosto de 2015.

[15] FieldAware. FieldAware - scheduling and dispatch. http://goo.gl/Ap2EzP. Acedido em 28 de Agosto de 2015.

[16] Synchroteam. Synchroteam. http://goo.gl/9iB8k3. Acedido em 15 de Agosto de 2015.

[17] Synchroteam. Synchroteam - blog. http://goo.gl/GnazBK. Acedido em 28 de Agosto de 2015.

[18] fieldtechnologies. study gps fleet tracking saves 5484 per driver. http://goo.gl/6piaF2. Acedido em 25 de Janeiro de 2015.

[19] Teletrac. Teletrac. http://goo.gl/5aDW2t. Acedido em 8 de Janeiro de 2015.

[20] Teletrac. Prism. http://goo.gl/IgD6uv. Acedido em 8 de Janeiro de 2015.

[21] Teletrac. Teletrac - Driver Safety Behaviour. http://goo.gl/KKdwNv. Acedido em 28 de Agosto de 2015.

[22] FleetMatics. FleetMatics. http://goo.gl/unVo4z. Acedido em 9 de Janeiro de 2015.

[23] Automotive Fleet. Technology Fuels Productivity for Fleets. http://goo.gl/bblM7s. Acedido em 9 de Janeiro de 2015.

[24] thinkinghighways. small fleets switch on to benefits of telematics. http://goo.gl/D7yKGE. Acedido em 28 de Agosto de 2015.

[25] Tiago Borges. Your fleet intelligence at the distance of a click. http://goo.gl/fWvB7i. Acedido em 28 de Agosto de 2015.

[26] Masternaut. Masternaut. http://goo.gl/JUEWLl. Acedido em 10 de Janeiro de 2015.

[27] Masternaut. Masternaut. http://goo.gl/EDToxl. Acedido em 11 de Janeiro de 2015.

[28] vehicletrackingtech. Masternaut Tax & Expense. `http://goo.gl/KqwLjT`. Acedido em 28 de Agosto de 2015.

[29] rtafleet. RTA fleet. `http://goo.gl/h8HgYT`. Acedido em 17 de Janeiro de 2015.

[30] capterra. capterra. `http://goo.gl/C5V8HD`. Acedido em 20 de Janeiro de 2015.

[31] getapp. getapp. `http://goo.gl/N6XGCR`. Acedido em 20 de Janeiro de 2015.

[32] Pentaho. Kettle. `http://goo.gl/WImqdG`. Acedido em 15 de Agosto de 2015.

[33] Informatica. Informatica PowerCenter. `https://goo.gl/LFsF3l`. Acedido em 15 de Agosto de 2015.

[34] Adeptia. Adeptia Integration Suite. `https://goo.gl/jqRm9e`. Acedido em 15 de Agosto de 2015.

[35] Mike Cohn. *User Stories Applied: For Agile Software Development*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2004.

[36] Haughey D. MoSCoW Method. `https://goo.gl/rSb6BN`. Acedido em 15 de Agosto de 2015.

[37] Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine.

[38] IMTT. Eco Condução. `http://goo.gl/FQbjbq`. Acedido em 11 de Janeiro de 2015.

[39] Alexandre Coutinho. "Eco condução pode poupar mais de 800 milhões"Expresso Maio de 2010. `http://goo.gl/FL9lny`. Acedido em 11 de Janeiro de 2015.

[40] Frotcom. Frotcom. `http://goo.gl/tXeOWu`. Acedido em 9 de Janeiro de 2015.

[41] TomTom. TomTom. `http://goo.gl/ZHijYi`. Acedido em 9 de Janeiro de 2015.

[42] techempower. Web Framework Benchmarks. `http://goo.gl/DbwgXs`. Acedido em 19 de Janeiro de 2015.