

Universidade do Minho

Escola de Engenharia

Departamento de Informática

Tarcísio Malheiro

**Serviço Safe-return-home baseado
em Análise de Big Data em Tempo Real**

December 2016



Universidade do Minho

Escola de Engenharia

Departamento de Informática

Tarcísio Malheiro

**Serviço Safe-return-home baseado
em Análise de Big Data em Tempo Real**

Master dissertation

Master Degree in Computing Engineering

Dissertation supervised by

Pedro Rangel Henriques

Alda Gançarski

December 2016

ABSTRACT

In our society there are various forms of crimes that occur quite often. To diminish crime, a number of safe-return systems are being developed for vulnerable people. However these systems are focused mainly in response to situations in which the danger has occurred, with no effort to predict a possible crime. These predictions of danger situations can be done with the gathering of information concerning the surroundings of the user, in real time. A safe-return-home system should allow the users of the service, to be notified in a timely fashion of possible dangerous situations that occur near the user's location, in real time. These predictions are made through the analysis of various types of data about the surroundings of the user. The data collection and analysis should include information regarding news about crimes, accidents and disasters, and the user's location. Data analysis then allows to predict dangers, so that they can be visually recognized on the user's smartphone, making it possible to respond actively to these dangers, in real time.

RESUMO

Na nossa sociedade existem várias formas de crimes que ocorrem com bastante frequência. No combate ao crime, vários sistemas safe-return estão a ser desenvolvidos para pessoas vulneráveis. Contudo estes sistemas focam-se principalmente na resposta a situações em que o perigo já aconteceu, não é feita nenhuma previsão de um possível crime. Estas previsões de situações de perigo podem ser feitas com a informação recolhida, relativa às proximidades do utilizador, em tempo real.

Um sistema de safe-return-home deve permitir que os utilizadores do serviço sejam notificados, atempadamente, de possíveis situações perigosas que ocorram nas proximidades da localização do utilizador, em tempo real. Estas previsões são feitas através da análise de vários tipos de dados relativos ao que se encontra em redor do utilizador. A recolha e análise de dados deverá incluir informação relativa a notícias relacionadas com crimes, acidentes e desastres, incluindo também a localização do utilizador. A análise dos dados permite então fazer uma previsão de perigos, de forma que possam ser visualmente reconhecidos no smartphone do utilizador, sendo assim possível reagir de forma ativa a esses perigos, em tempo real.

CONTEÚDO

1	INTRODUÇÃO	1
1.1	Motivação	3
1.1.1	Idosos	3
1.1.2	Crianças	3
1.2	Objetivo	4
1.3	Hipótese de Investigação	4
1.4	Estrutura do Documento	4
2	SERVIÇO SAFE-RETURN-HOME	5
2.1	Trabalho Relacionado	5
2.1.1	Comparação com as aplicações atuais	7
2.2	Fontes de Informação	8
3	SERVIÇO SAFE RETURN HOME: ABORDAGEM PROPOSTA	9
3.1	Funcionamento	11
3.1.1	Servidor	11
3.1.2	Aplicação Web	11
3.1.3	Aplicação Móvel	11
4	EXTRAÇÃO DE INFORMAÇÃO	13
4.1	Técnicas de recolha de informação	13
4.1.1	Web crawling	13
4.1.2	Web scraping	14
4.2	Fontes consultadas	17
4.2.1	Estradas de Portugal	18
4.2.2	Instituto Português do Mar e da Atmosfera	21
4.2.3	OpenWeatherMap	25
4.2.4	Cidades de Portugal	26
4.3	Normalização da informação	29
4.3.1	Base de dados	29
5	SISTEMA WEB	33
5.1	Node.js	33
5.2	StrongLoop LoopBack	34
5.2.1	Configurações	36
5.3	GoogleMaps	38
5.4	Interface Web	39

Conteúdo

5.4.1	Resultado do Cliente em AngularJS	39
6	APLICAÇÃO ANDROID	44
6.1	Android Studio	44
6.2	GoogleMaps	45
6.3	Integração da informação extraída	45
6.4	Aplicação	47
7	CONCLUSÃO E TRABALHO FUTURO	51
7.1	Trabalho Futuro	51

LISTA DE FIGURAS

Figura 1	Arquitetura do Sistema Safe-return-home	10
Figura 2	Filtro do Sistema Safe-return-home	10
Figura 3	As ocorrências retiradas da página do estradas.pt	19
Figura 4	Os Alertas Meteorológicos disponíveis no site do IPMA	22
Figura 5	Os Avisos de Sismos disponíveis no site do IPMA	23
Figura 6	As cidades de Portugal	26
Figura 7	As vilas de Portugal	27
Figura 8	Modelo da recolha e filtragem da informação	28
Figura 9	Diagrama da Base de Dados	30
Figura 10	Parte da arquitectura do sistema SRH	32
Figura 11	Hierarquia de Herança do Modelo (StrongLoop, 2016)	35
Figura 12	Exemplo do Autocomplete	38
Figura 13	Barra de navegação	39
Figura 14	Separador com a tabela de ocorrências	40
Figura 15	Separador com a tabela de temperaturas	40
Figura 16	Separador com a tabela de sismos	41
Figura 17	Separador com o mapa	42
Figura 18	Continuação do separador com o mapa	42
Figura 19	Exemplo de um percurso escolhido	43
Figura 20	Direções do percurso escolhido	43
Figura 21	Login do utilizador	48
Figura 22	Ecrã de boas vindas	48
Figura 23	Ecrã com o Mapa	49
Figura 24	Percurso Braga-Barcelos	49
Figura 25	Temperatura de Braga	50
Figura 26	Diálogo com o contador	50

INTRODUÇÃO

Hoje em dia numa sociedade moderna ocorrem, constantemente, diversas formas de crime. Todos os dias existem notícias de todos os tipos de crimes, levando a que qualquer habitante residente numa cidade, grande ou pequena, seja mais cuidadoso. A existência de vigilantes ou a tomada de decisões preventivas de senso comum conseguem prevenir alguns destes crimes. Para as pessoas que sejam mais vulneráveis, como crianças, estão a ser desenvolvidos sistemas de segurança para o dia-a-dia, tentando assim diminuir a quantidade de crimes.

Com o objetivo de fazer um regresso a casa em segurança surgem os serviços safe-return-home, ajudando assim qualquer pessoa a chegar a casa em segurança. Os sistemas atuais de safe-return-home são ineficientes, estão apenas focados na resposta a situações de socorro ou de um crime que já foi cometido, não fazem nenhum tipo de previsão da probabilidade da ocorrência de um crime em tempo real.

Para preencher a necessidade de algo, o objectivo desta tese é o desenvolvimento de um serviço de safe-return-home que permita aos utilizadores serem avisados de possíveis situações perigosas na sua redondeza em tempo real. Para que estes avisos possam surgir é necessário que vários dados do tipo *Big Data* sejam recolhidos e devidamente analisados acerca do ambiente que rodeia o utilizador. Um exemplo desse tipo de dados é a Web Semântica considerada uma fonte de informação pública por excelência, devido à riqueza de conceitos e relações que nos traz. Os dados recolhidos e analisados deverão incluir informações relativas à localização do utilizador, a notícias em tempo real do tipo crime, desastres e acidentes, locais de abrigo e notícias recolhidas na Web. Fazendo a análise de estes dados, será possível a visualização de situações potencialmente perigosas relativas à posição do utilizador.

BIG DATA

Hoje em dia, com a informação que circula pela internet e com os avanços tecnológicos, todas as pessoas com um smartphone podem aceder e facilmente partilhar ou passar informação de utilizador para utilizador e de dispositivo para dispositivo. Como esta enorme quantidade de dados a ser produzida no momento não pode ser processada ou analisada utilizando ferramentas tradicionais, os sistemas recorrem ao uso de *Big Data*.

Big Data é um termo amplamente usado na atualidade para nomear conjuntos de dados muito grandes ou complexos, que as aplicações de processamento de dados tradicionais ainda não conseguem processar (Wikipedia, 2016a). Num cenário em que informação seja complicada de processar por causa do volume, variedade e velocidade de informação, trata-se de um problema de *Big Data*.

O termo *Big Data* é usado muitas vezes em *predictive analytics*, que consiste na extração de informação de datasets com a finalidade de determinar padrões e prever resultados e tendências, ou em outros métodos para extrair valor de dados. A análise de *Big Data* pode levar a melhores tomadas de decisão, e melhores decisões podem resultar numa melhor eficácia operacional, redução de custos e riscos.

WEB SEMÂNTICA

A Web Semântica é uma extensão da *World Wide Web* atual, que permitirá aos computadores e humanos trabalharem em cooperação. A Web Semântica tem como foco principal a integração, intercâmbio e compreensão semântica da informação, tanto na ótica dos humanos, como na ótica das máquinas, através de *Metadata* (dados sobre outros dados), de ontologias e de agentes de software. A ideia da Web Semântica pode ser resumida como a solução da forma como os computadores conseguem entender o conteúdo da *Web*, começando por organizar e estruturar a informação e em seguida adicionar semântica à informação da *Web*, de tal forma que os agentes de software possam compreendê-la.

A informação na Web Semântica é útil em vários domínios, contudo a quantidade de dados é tão grande que a sua exploração pode ser complicada para um utilizador que não esteja familiarizado com os modelos de dados *RDF*, *OWL* e *XML*.

- XML (*eXtensible Markup Language*) é uma linguagem de marcação definida num formato para descrever dados estruturados que facilita declarações mais precisas do conteúdo.
- RDF (*Resource Description Framework*) é a linguagem de modelação dos dados para a Web Semântica. Toda a informação na Web Semântica é guardada e representada no modelo de dados RDF.

1.1. Motivação

- OWL (*Ontology Web Language*) é uma linguagem de representação de conhecimento da Web Semântica. Uma ontologia OWL permite que um utilizador defina conceitos agregáveis de forma que estes possam ser reutilizados o máximo possível. Desta forma cada conceito é cuidadosamente definido, podendo ser selecionado e usado em várias combinações com outros conceitos necessários para diferentes aplicações.

1.1 MOTIVAÇÃO

Dentro da nossa sociedade existem formas de crimes que têm como público-alvo pessoas vulneráveis, como por exemplo crianças, pessoas fisicamente fracas ou idosos. Existe uma real preocupação com a segurança destas pessoas mais vulneráveis, seja por parte delas ou de quem as rodeia, levando a que tomem medidas que diminuam a sua ansiedade quando fazem algum tipo de percurso. Em algumas situações recorre-se a serviços de *safe-return* mas estes focam-se apenas em responder a situações de perigos que já aconteceram.

1.1.1 Idosos

Para pessoas idosas existem múltiplos serviços de auxílio, nomeadamente em casos de demência, variando de comunidades de ajuda voluntária a serviços pagos. Com o número elevado de idosos que sofrem de Alzheimer, autismo ou síndrome de Down, a probabilidade destes mesmos se perderem ou desorientarem é grande, por isso são desenvolvidos programas de apoio e dispositivos de localização como braceletes.

Destacam-se as iniciativas do MedicAlert+ ([MedicAlert+](#), 2016) e o Lifesaver ([Lifesaver](#), 2016) de um trabalho de pesquisa já elaborado sobre programas de localização de idosos ([Petonito et al.](#), 2013).

Atualmente existe uma vasta variedade de organizações e programas dedicados ao *Safe Return*, como instituições sem fins lucrativos, Websites de voluntariado, governamentais e comerciais ¹.

1.1.2 Crianças

Para pais preocupados com a segurança dos seus filhos, existem várias aplicações e serviços de localização para monitorizarem as suas deslocações. Estas aplicações podem ser dispositivos com GPS como relógios para serem usados pelas crianças permitindo aos seus responsáveis saber a localização exata dos seus filhos², ou ainda sistemas para toda a

¹ <https://www.wolfson.ox.ac.uk/~ben/SafeReturn/Links.html>

² <http://www.safewise.com/blog/10-wearable-safety-gps-devices-kids>

1.2. Objetivo

família como o ZoeMob (Zoemob, 2016) que permite localização dos membros familiares, comunicação directa e também a possibilidade de agendar atividades e compromissos.

1.2 OBJETIVO

O principal foco desta dissertação será o estudo dos métodos de recolha, análise e visualização de *Big Data* com a finalidade destes dados serem utilizados na prevenção de situações perigosas durante o percurso de uma pessoa vulnerável. Começando pela pesquisa bibliográfica, que é necessária para servir de suporte para uma proposta de um serviço de safe-return-home e culminando no desenvolvimento de uma aplicação Web e uma aplicação Android.

1.3 HIPÓTESE DE INVESTIGAÇÃO

Pretende-se mostrar com este trabalho que é possível recolher e tratar informação de interesse público disponível na Web para aumentar a segurança dos cidadãos na sua vida quotidiana e apenas recorrendo à atual tecnologia móvel hoje comum.

1.4 ESTRUTURA DO DOCUMENTO

Nos capítulos que se seguem é apresentado o trabalho relacionado e o serviço de safe-return-home. No próximo capítulo (Capítulo 2) são expostos os trabalhos relacionados, como sistemas focados na segurança pessoal e sistemas de localização. No Capítulo 3 é descrita a abordagem escolhida para o desenvolvimento de um serviço de safe-return-home. No Capítulo 4 são descritos os métodos e tecnologias usados para a extração e normalização de informação para a construção de uma base de dados consistente e coerente. Em seguida são apresentados o servidor e clientes (Web e Android) do sistema. No Capítulo 5 foi utilizada uma plataforma que permite o desenvolvimento de um servidor e um cliente que comunicam via API para exposição da informação recolhida. No Capítulo 6 é apresentada a aplicação Android que vai comunicar via API REST com o servidor para que os utilizadores autenticados possam usufruir das funcionalidades do sistema.

SERVIÇO SAFE-RETURN-HOME

Neste capítulo serão referenciados vários trabalhos na área de aplicações de segurança pessoal, que fazem uso de serviços de localização para auxiliar o utilizador da aplicação. Na primeira secção será retratado o trabalho relacionado com o serviço de safe-return-home. Na secção seguinte encontram-se as fontes de informação que serão usadas no serviço de safe-return-home proposto.

2.1 TRABALHO RELACIONADO

Real Time Safe-return-home Service Based on Big Data Analytics

Na Coreia do Sul foi proposto um sistema de safe-return-home (Lee et al., 2014), que notifica os utilizadores de potenciais situações perigosas em seu redor em tempo real. Neste sistema, é feita a recolha e tratamento de vários tipos de dados em Big Data como a localização dos utilizadores, as localizações de cameras CCTV (Closed-Circuit Television) e dados de redes sociais. Com a análise dos dados recolhidos é feita a previsão de potenciais perigos nas proximidades. A previsão é então visualizada no smartphone e são feitas sugestões ao utilizador contra os perigos previstos, em tempo real (Lee et al., 2014).

Kitestring

O *Kitestring* representa um serviço web-based que se mantém em comunicação com o utilizador e envia alertas para contactos previamente seleccionados, caso o utilizador deixe de responder ao serviço, trabalhando assim a partir da inatividade detectada pela aplicação.

O *Kitestring* faz check-ins via SMS, tornando este serviço operacional em qualquer telemóvel, sendo necessário que o utilizador visite o website para se registar neste serviço gratuito. Este serviço depois de ativado pode ser configurado de acordo com as preferências do utilizador. É possível definir palavras ou códigos, definindo uma palavra de *Check-in* para poder desativar o serviço, tal como um código *Duress* para se usar numa situação em que algo correu mal antes do tempo de check-in. Com o envio de uma SMS com estas pala-

2.1. Trabalho Relacionado

vras para o Kitestring é terminada a viagem que estava a ser monitorizada ou é contactado o contacto de emergência de imediato (Kitestring, 2016). O KiteString está disponível em vários países mas a sua interface é apenas em Inglês.

bSafe

bSafe é uma aplicação de segurança pessoal para smartphones e contém várias funcionalidades, pode servir para alertar diretamente amigos que o utilizador chegou ao seu destino. Nos casos em que o utilizador necessite de ser ajudado, o *bSafe* começa a capturar vídeo e a localização GPS, estas informações são recolhidas e podem ser partilhadas com a polícia. Permite também usar um temporizador que, se não for desligado pelo utilizador, envia uma mensagem SOS automaticamente.

O bSafe permite criar uma rede de amigos ilimitada aumentando assim a segurança do utilizador. Esta amizade é mútua fazendo com que o utilizador que adicionou um amigo vai também ajudar na segurança deste novo amigo. Um utilizador deste serviço pode simular uma chamada falsa através da predefinição da hora em que espera receber a chamada. O bSafe também permite que um utilizador convide um amigo para o supervisionar na sua caminhada (bSafe, 2016).

Bugle

Sendo uma aplicação para *iPhone*, o *Bugle* é direcionado para desportistas que não querem levar o seu telemóvel enquanto fazem desporto no exterior. Os utilizadores definem quanto tempo vão demorar a voltar a casa e onde vão estar enquanto estão fora, se não chegarem a tempo de desligar o temporizador, o contactos de emergência definidos vão ser alertados via SMS ou email que o utilizador pode estar numa situação de perigo (Bugle, 2016). O Bugle é uma aplicação que não faz uso dos serviços de localização GPS, o utilizador predefine o local onde vai praticar a sua atividade e em caso de emergência o utilizador da aplicação será procurado na área da atividade.

SafeTrek

A *SafeTrek* é uma aplicação de emergência, serve para alertar a polícia numa situação perigosa, o utilizador mantém o dedo no botão Safe no ecrã, se retirar o dedo será pedido um código de 4 dígitos, caso não seja introduzido a policia será notificada. A aplicação SafeTrek foi originalmente criada com o intuito de ajudar mulheres a sentirem-se mais seguras quando regressam a casa sozinhas. Quando usado o SafeTrek, os utilizadores mantêm o dedo no ecrã e seguram até se sentirem seguros, isto vai possibilitar que o podem segurar o

2.1. Trabalho Relacionado

smartphone no bolso enquanto caminham e estarem assegurados que serão ajudados caso sejam confrontados com alguma situação de perigo. Esta aplicação está disponível apenas nos Estados Unidos da América ([SafeTrek, 2016](#)).

Samsung Galaxy Safety Assistance

Os Samsung Galaxy S5, S6 e S7 e os telemóveis Note possuem múltiplas funcionalidades, sendo uma delas a “Safety Assistance”. Esta funcionalidade interna permite, numa situação de emergência, notificar contactos predefinidos que o utilizador pode estar em perigo. Esta assistência de segurança pode ser configurada através das definições do telemóvel. Uma vez ativada, é possível enviar alertas SOS pressionando o botão de bloqueio três vezes. A informação enviada no alerta vai incluir a localização do telemóvel, fotos tiradas pelas cameras frontal e traseira e uma breve gravação de voz. A Samsung Safety Assistance não contacta nenhuma autoridade ou serviço de emergência ([AndroidCentral, 2016](#)).

Circle of 6

O Circle of 6 é uma aplicação grátis para dispositivos iOS e Android que permite criar um círculo de seis amigos ou pessoas de confiança para uma chamada rápida quando o utilizador se encontra em perigo e necessita de ajuda, numa má situação com outra pessoa, ou na necessidade de alertar as autoridades. Com um simples toque a aplicação liga aos amigos ou serviços de emergência com o pedido de ajuda do utilizador juntamente com a sua localização. A ideia desta aplicação é para alguém que precise de ajuda e não seja totalmente necessário um telefonema para o 112. Por exemplo, pedidos de boleias que com um simples toque é pedido a um amigo para o ir buscar com o envio de uma mensagem com a sua localização. Um utilizador pode escolher o ícone com um carro para fazer um pedido de boleia, o ícone de um telefone para fazer um pedido de chamada ou o ícone de Chat para pedir a um amigo que lhe envie uma mensagem. Em todas estas opções a aplicação partilha a localização do utilizador ([Circleof6, 2016](#)).

2.1.1 *Comparação com as aplicações atuais*

A aplicação de safe-return-home que vai ser desenvolvida vai proporcionar funcionalidades que as atuais aplicações não possuem. A construção deste serviço com o público alvo a ser Portugal Continental vai orientar o desenvolvimento desta aplicação a recorrer a fontes de informação relacionadas a localidades portuguesas.

A principal diferença do serviço safe-return-home (SRH) com as aplicações atuais vai ser a recolha prévia de informação que vai permitir que os utilizadores sejam avisados de zonas

2.2. Fontes de Informação

por onde não é aconselhável passar. Várias funcionalidades vão coincidir com as aplicações atuais, nomeadamente na resposta a uma situação de emergência onde o utilizador pode precisar de ajuda. Este auxílio pode ser manifestado com um contacto de emergência ficar em alerta e tentar resolver a situação ou em casos mais sérios recorrer-se às Autoridades através do envio da última localização registada do utilizador. As diferenças nas funcionalidades entre o SRH e outras aplicações podem ser verificadas na Tabela 1.

	SRH	Outras Aplicações
Direcionada a Portugal	Sim	Não
Identificação de situações perigosas	Sim	Não
Planeamento de percurso	Sim	Não
Contactos de emergência	Sim	Sim
Partilha da localização do utilizador	Sim	Sim
Contactar a polícia	Sim	Sim

Tabela 1: Comparação do sistema com as aplicações existentes

2.2 FONTES DE INFORMAÇÃO

Para o serviço de safe-return-home funcionar, este vai precisar de fazer a recolha de dados que identifiquem situações perigosas. Estes dados poderão ser recolhidos de diversas fontes de informação obtendo assim o máximo de informação útil possível para o sistema poder posteriormente analisar. As fontes poderão variar desde redes sociais, dados públicos até repositórios online. Com a grande variedade de formatos dos dados provenientes de n fontes, existirá a necessidade de usar várias técnicas de extração de informação destas fontes, desde técnicas como *Web Scrapping* até *Data Mining*.

As fontes de informação que deverão ser usadas nesta abordagem a um sistema de safe-return-home, serão provenientes de várias páginas Web com conteúdo diverso. A recolha de informação será feita:

- em notícias online;
- na Web semântica;
- em dados do tipo *linked data* em Portugal;
- no site da Proteção Civil;

Para cada tipo de informação será necessário aplicar diferentes técnicas para extração da informação (ver Cap. 4) que irão permitir fazer uma previsão de situações perigosas em qualquer zona do percurso de um utilizador do serviço safe-return-home.

SERVIÇO SAFE RETURN HOME: ABORDAGEM PROPOSTA

Sendo o serviço de safe-return-home um sistema de segurança pessoal que alerta, o utilizador, de potenciais perigos em seu redor em tempo real, são necessários vários processos para o funcionamento do sistema. Os alertas são baseados em previsões em tempo real feitas através da análise de vários tipos de dados relativos ao que se encontra em redor do utilizador. A recolha e análise de dados deverá incluir informação relativa a notícias relacionadas com crimes, acidentes e desastres, incluindo também a localização do utilizador. A análise dos dados permite então fazer uma previsão de perigos, de forma que possam ser visualmente reconhecidos no smartphone do utilizador, sendo assim possível reagir de forma ativa a esses perigos, em tempo real.

O sistema recebe informação sobre situações perigosas ou a ter em atenção, provenientes de diversas fontes ou famílias de fontes. Toda esta nova informação é recolhida, filtrada e processada e posteriormente é guardada na base de dados. É desenvolvido um servidor que trata de responder a pedidos e autenticações. Na interação, o servidor vai receber pedidos de utilizadores via aplicação Web ou Android, sendo necessária a identificação do utilizador. A identificação dos utilizadores é efetuada pelo o servidor através da verificação das credenciais na base de dados. Para a resposta a um pedido, o servidor acede à base de dados para obter informação necessária, sendo então gerado um output.

A arquitetura do serviço de safe-return-home proposto pode ser visualizada na Figura 1.

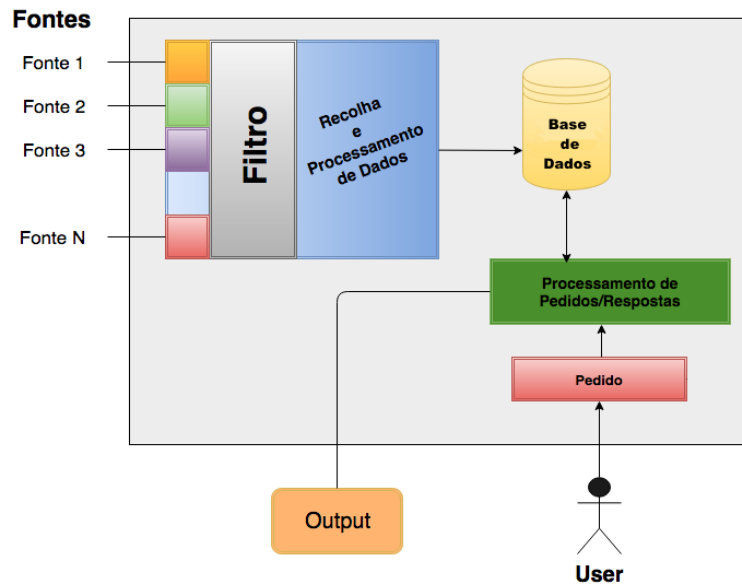


Figura 1: Arquitetura do Sistema Safe-return-home

Na fase de recolha e processamento de dados do sistema de safe-return-home, a informação passa por um filtro, apresentado na Figura 2. Após a recolha de informação de fontes diferentes, os dados são filtrados para que seja obtida apenas informação relevante para o sistema. De seguida todos os dados são normalizados, para permitir um armazenamento consistente e um acesso eficiente, quando estes estiverem guardados na base de dados. Por fim, é feita a consolidação dos dados, de forma a unificar a informação que foi recolhida de fontes diversas, formando assim uma base de dados consistente para que seja possível fazer a previsão de situações de perigo.

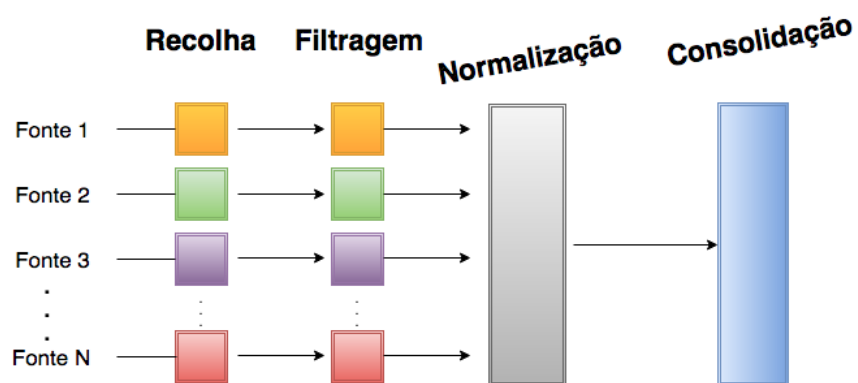


Figura 2: Filtro do Sistema Safe-return-home

3.1. Funcionamento

3.1 FUNCIONAMENTO

3.1.1 *Servidor*

Para que sejam atendidos os pedidos feitos pelos utilizadores através de ambas aplicações vai ser desenvolvido um servidor apto a aceder à informação presente na base de dados e fornecer as respostas com eficiência.

3.1.2 *Aplicação Web*

Na aplicação Web será possível consultar toda a informação presente na base de dados e planear uma viagem em segurança. As funcionalidades do sistema serão apresentadas após o registo e login do utilizador no site. O servidor vai verificar a autenticação do utilizador no sistema. É apresentada ao utilizador toda a informação presente na base de dados permitindo que o utilizador fique informado de situações possivelmente perigosas. Na funcionalidade de realizar um percurso seguro, o utilizador vai definir uma origem e um destino. Em seguida é apresentado num mapa o percurso juntamente com marcadores com as situações de potencial perigo para o utilizador. O percurso pode ser alterado pelo utilizador de forma a evitar os marcadores no mapa.

3.1.3 *Aplicação Móvel*

Para iniciar o processo de funcionamento do serviço safe-return-home na aplicação móvel, um utilizador deverá fazer o login no sistema e definir um percurso. O servidor vai calcular a distância da posição que o utilizador definiu como origem até ao seu destino e definir um alarme com um contador de acordo com o tempo estimado para o utilizador chegar ao seu destino. À medida que o utilizador se desloca para o seu destino, a sua localização será recolhida periodicamente para que o servidor registe todo o trajeto efetuado. Caso aconteça algo inesperado como, por exemplo, a expiração do tempo previamente estimado pelo sistema antes do utilizador chegar ao seu destino, a aplicação vai considerar que possivelmente terá de resolver uma situação de emergência e começa então um procedimento sequencial de três passos. Numa primeira situação o contador decrescente expirou e é apresentada ao utilizador uma mensagem, onde é feito um pedido para reiniciar o contador, caso o utilizador esteja em segurança. Na ausência de resposta do utilizador da primeira mensagem enviada, a aplicação vai, então informar a possibilidade de ter ocorrido uma situação de perigo a uma pessoa designada de responsável pela segurança do utilizador, sendo este o segundo passo do procedimento. Neste ponto, caso não haja nenhuma resposta por parte do utilizador após o segundo passo, o sistema considera que se

3.1. Funcionamento

trata de uma situação de emergência e o utilizador necessita de ajuda, por isso é enviada a informação do utilizador para a estação de polícia mais próxima.

EXTRAÇÃO DE INFORMAÇÃO

Neste capítulo serão descritos os métodos e tecnologias usados para a extração e normalização de informação para a construção de uma base de dados consistente e coerente. Na secção inicial (Sec 4.1) serão apresentadas as técnicas de extração de informação da internet. Na secção seguinte (Sec 4.2) encontram-se as fontes de dados consultadas para a recolha da informação com recurso às técnicas referidas anteriormente. Na secção final (Sec 4.3) será descrita a normalização da informação que vai ser incluída na base de dados.

4.1 TÉCNICAS DE RECOLHA DE INFORMAÇÃO

Na primeira fase de construção do serviço de SRH (safe-return-home) é feita a extração de informação. Os dados alvo para este processo estão disponíveis na Internet, levando à necessidade de recorrer a tecnologias que consigam retirar informação relevante. O facto de estes dados se encontrarem em HTML leva à construção de uma aplicação que recorra ao uso de técnicas de extração de conteúdo HTML como Web crawling ou Web scraping. Como o Web crawling (Sec 4.1.1) e o Web scraping (Sec 4.1.2) estão muito relacionados é necessário compreender como funcionam ambos. Após uma análise comparativa, optamos por utilizar o Web scraping, para a extração da informação relativa a atividades climáticas e distúrbios nas estradas de Portugal para a construção deste serviço SRH.

4.1.1 *Web crawling*

Um Web Crawler é por definição um programa que navega pela *World Wide Web* de uma forma metódica e automatizada. Web crawlers são também conhecidos por indexadores automáticos, internet bots, Web spiders e Web robots. O processo executado por um Web Crawler é chamado de Web crawling ou spidering. Para muitos sites, em particular os motores de busca, o uso de crawlers é essencial para manter as suas bases de dados atualizadas. Os Web crawlers são principalmente utilizados para criar uma cópia de todas as páginas visitadas para um pós-processamento por um motor de busca que irá indexar as páginas

4.1. Técnicas de recolha de informação

copiadas para providenciar procuras mais rápidas. Crawlers também podem ser usados para tarefas de manutenção automatizadas de um Web site, como verificar os endereços ou validar o código *HTML* (Wikipedia, 2016b).

4.1.2 *Web scraping*

Web scraping, conhecido também como *Web harvesting* ou *Web data extraction*, é uma técnica para desenvolver software para a extração de informação de websites. Esta captura de informação é conseguida através de uma implementação direta de *Hypertext Transfer Protocol* (na qual é baseada a Web), ou a incorporação de um Web browser.

O Web scraping é associado ao *Web indexing*, onde é feita a indexação da informação na Web com o uso de um *bot* ou *Web crawler* sendo uma técnica universalmente adoptada pela grande maioria dos motores de busca. No entanto, Web scraping centra-se mais na transformação de dados não estruturados da web, normalmente encontram-se no formato *HTML*, em dados estruturados que podem ser guardados e analisados numa base de dados ou folha de cálculo. É usualmente relacionado com a *Web automation*, onde é simulada a navegação humana com o uso de software. Outros casos de utilização do *Web scraping* são a comparação de preços online, extração de contactos, monitorização meteorológica, deteção de modificações em websites (Wikipedia, 2016e).

Para realizar a extração de informação é necessário recorrer a ferramentas de Web scraping. As que de seguida se apresentam foram as escolhidas para o projeto, sendo Java a principal linguagem de programação. Estas ferramentas vão permitir a recolha de informação das fontes consultadas (Sec. 4.2), em que cada uma delas terá um processamento e tratamento particular que será posteriormente descrito.

Selenium

O Selenium é uma plataforma de teste de software open-source para aplicações Web. Fornece uma linguagem de teste específica ao domínio (Selenese) para a escrita de testes em várias das mais populares linguagens de programação, como Java, C#, Groovy, Perl, PHP, Python e Ruby. Estes testes podem então ser usados na maioria dos browsers modernos. O Selenium está disponível para as plataformas Windows, Linux e Macintosh. Como alternativa à escrita de testes em Selenese, os testes podem ser escritos em várias linguagens de programação. Neste caso, os testes vão comunicar com o Selenium pelo uso de métodos via API com o Selenium Client API. Com o lançamento do Selenium 2, foi apresentado um novo cliente API que tem como componente central o *WebDriver*.

O Selenium *WebDriver* aceita comandos, tanto em Selenese ou via API, e envia-os para um browser. É implementado através de um browser driver de um browser específico, que envia comandos para um browser e recolhe os resultados. O Selenium *WebDriver*

4.1. Técnicas de recolha de informação

usa o suporte nativo do browser para automação (que é diferente em cada browser). A maioria dos browser driver lançam e acedem a uma aplicação browser (Firefox ou Internet Explorer). Existe também um *HtmlUnit* browser driver, que simula um browser a usar *HtmlUnit* (Wikipedia, 2016f; Selenium, 2016). A sua API é orientada ao objeto o que facilita o seu uso. O WebDriver começa, de forma direta, uma instância de um browser, permitindo o seu controlo e manipulação. Sendo que com o WebDriver é possível fazer a extração do conteúdo de uma página HTML.

Na Listagem 4.1 é apresentado o código da função de exemplo do funcionamento do Selenium com o *getTitle()*. Nesta função, é criada uma nova instância do driver do Firefox (linha 3), em seguida é visitado o site da Google (linha 5) e também recolhido o elemento HTML através do seu nome (linha 7). No final é apresentado o título da página e o elemento (linha 9) e no final é terminado o uso do WebDriver (linha 11).

```
1 public static void getTitle() {
2
3     WebDriver driver = new FirefoxDriver();
4
5     driver.get("http://www.google.com");
6
7     WebElement element = driver.findElement(By.name("q"));
8
9     System.out.println("Page title is: " + driver.getTitle()+ " The Element is
10         : " + element.toString());
11
12     driver.quit();
13 }
```

Listing 4.1: Exemplo de recolha do título de uma página Web com o Selenium

HtmlUnit

O *HtmlUnit* pode ser descrito como um browser sem interface gráfica para programas escritos em Java. Tem a capacidade para modelar documentos HTML e fornece uma API que permite invocar páginas, preencher formulários, clicar em links, em suma permite simular um browser com todas as suas funcionalidades. O *HtmlUnit* emula partes do comportamento do browser incluindo aspetos de baixo nível como o TCP/IP e HTTP. Este browser sem cabeçalho (headless) consegue lidar com a segurança HTTPS, autenticação básica http, redirecionamento automático e outros cabeçalhos. Permite que o código de teste em Java examine as páginas retornadas como texto, um XML DOM, ou coleções de formulários, tabelas e links. *HtmlUnit* tem um bom suporte de Javascript e é capaz de trabalhar com bibliotecas de AJAX bastante complexas, simular o Google Chrome, Firefox ou Internet

4.1. Técnicas de recolha de informação

Explorer dependendo da configuração usada. A utilização do HtmlUnit é normalmente associada a realização de testes ou à extração de informação de Web sites (HtmlUnit, 2016; Wikipedia, 2016g).

Por exemplo, a função `getElements()` apresentada na Listagem 4.2 permite uma melhor percepção do que é possível fazer com o HtmlUnit. Esta função vai criar um `WebClient` (linha 2) que vai possibilitar a recolha de uma página de um determinado URL (linha 3). Uma vez recolhida a página HTML, é possível retirar a informação que o programador desejar. Neste exemplo o objetivo é obter um elemento HTML de tipo `div` através do seu `ID` (linha 4) e um elemento `anchor` através do seu nome (linha 5).

```
1 public void getElements() throws Exception {
2     try (final WebClient webClient = new WebClient()) {
3         final HtmlPage page = webClient.getPage("http://some_url");
4         final HtmlDivision div = page.getHtmlElementById("some_div_id");
5         final HtmlAnchor anchor = page.getAnchorByName("anchor_name");
6     }
7 }
```

Listing 4.2: Exemplo de recolha do título de uma página Web com HtmlUnit

Jsoup

O `jsoup` é uma biblioteca open-source em Java em que os métodos disponíveis são pensados para a extração e manipulação de dados armazenados em documentos HTML. Fornece uma API convenientemente direcionada para a extração e manipulação de dados, usando os métodos DOM, CSS e o estilo jQuery. O `jsoup` implementa a especificação de WHATWG HTML5, e faz o parsing do HTML para o mesmo DOM tal como nos browsers modernos (Jsoup, 2016; Wikipedia, 2016c).

Algumas das funcionalidades que o `jsoup` fornece são:

- fazer o scraping e o parsing do HTML partindo de um URL, ficheiro ou string;
- a procura e extração de informação com o uso de DOM transversal ou selecionadores de CSS;
- manipulação de elementos, atributos e texto do HTML;
- um output HTML limpo e organizado;

No exemplo da Listagem 4.3 que se segue, a função `jsoup` listada vai recolher a página da Wikipedia, faz o seu parsing para DOM e seleciona os elementos HTML do tipo `anchors`

4.2. Fontes consultadas

com o atributo *href*.

```
1 Document doc = Jsoup.connect("http://en.wikipedia.org/").get();
2 Elements links = doc.select("a[href]");
```

Listing 4.3: Exemplo de recolha do título de uma página Web com jsoup

Os elementos do jsoup suportam um selecionador CSS ou jQuery para encontrar os elementos correspondentes, o que permite queries bastante poderosas e robustas. O método *select* está disponível num *Document*, *Element*, ou em *Elements*. É contextual, por isso é possível filtrar através da seleção de um *element* específico, ou através de um encadeamento de sucessivos *select*. O *select* devolve uma lista de *Elements*, que fornece uma vasta quantidade de métodos para a extração e manipulação de resultados.

4.2 FONTES CONSULTADAS

Para iniciar a construção de um serviço SRH é preciso preencher uma base de dados com informação relevante. Para obter esta informação, começamos por um processo de seleção de websites onde é necessário estudar a possibilidade de extrair os dados desejados. As seguintes fontes de informação foram consultadas para a construção do serviço SRH:

- Estradas.pt
- IPMA
- OpenWeather

Uma vasta lista de fontes foram consideradas e acabaram por não ser utilizadas. Começando pelos websites de notícias onde foram procuradas notícias relacionadas com acidentes e bloqueios de estradas. Esta procura foi feita com o uso de Web scraping onde para cada um dos sites era feita uma pesquisa recursiva de URL's. Assim, procura-se em cada página do website as notícias que se pretendem extrair, passando pela página inicial e pelos seus links internos fazendo uma navegação em profundidade no website. Com o auxílio de uma lista de palavras-chave são cruzadas as notícias com essa lista recolhendo apenas as notícias relevantes para o sistema. Nos seguintes websites, aquando consultados, não foi possível a recolha de informação relativa a acidentes ou bloqueio de estradas:

- Jornal de Notícias
- Correio da Manhã
- Correio do Minho

4.2. Fontes consultadas

- Diário de Notícias

A informação recolhida nos sites de notícias não foi considerada por conter informação com menor importância comparativamente às fontes escolhidas no final.

Com o intuito de obter as temperaturas nas localidades de Portugal foi visitado o site IPMA. Devido à ausência de uma API pública foi feito um pedido de acesso à API para uma possível recolha de temperaturas. Sem uma resposta a este pedido e devido à complexidade de fazer scraping do conteúdo dinamicamente gerado na página de temperaturas do IPMA, o site deixou de ser considerado para a recolha de temperaturas.

No site da Proteção Civil apenas havia informação em formato pdf que apelava à tomada de medidas de precaução na presença de chuva e ventos fortes. A falta de atividade e atualização de várias fontes levou ao descrédito das mesmas. A procura de informação na Web semântica e de dados do tipo *linked-data* não acrescentou material relevante às fontes de dados consultadas.

4.2.1 Estradas de Portugal

Estradas.pt é um portal mantido pelas Estradas de Portugal (EP) que oferece informações úteis sobre o trânsito em Portugal Continental. Informa das condições de trânsito, congestionamentos, obras, cortes de estrada, entre outros. Os registos aparecem associados a um mapa, onde existem vários sinais que indicam as ocorrências. Clicando nesses ícones, aparece então descrita a informação (Estradas.pt, 2016).

Uma das páginas deste portal contém uma tabela com as ocorrências. Nesta tabela encontra-se disponível o tipo, a estrada, o concelho, o distrito e a data de uma ocorrência. Os tipos de uma ocorrência são os seguintes:

- Acidentes
- Trabalhos
- Condicionamento
- Outros

Dos outros campos que são recolhidos, a *estrada* apresenta a informação da estrada e do quilómetro da ocorrência. O *Concelho* representa a localidade da estrada juntamente com seu o *Distrito* e a *Data* da ocorrência.

Na Figura 3 encontra-se o conteúdo da página das ocorrências do estradas.pt e é possível ver a lista de ocorrências que se pretende obter. Nesta página HTML a lista de ocorrências está dentro de uma tabela com paginação que é gerada dinamicamente. Sendo o objetivo de recolher todas as ocorrências, é necessário alterar o número de registos apresentados de

4.2. Fontes consultadas

forma a evitar a paginação, o que vai facilitar a recolha da tabela e de todos os registos. O elemento HTML *select* contém o valor 10 e restringe o número de registos por página da tabela. O Selenium (ver Sec. 4.1.2) vai permitir o uso de um WebDriver que consegue executar um Script que vai alterar esse número de registos por página para o valor desejado, possibilitando a recolha da tabela e de todos os seus registos.



Tipo	Estrada / km	Distrito	Concelho	Data
Outros	N239 ao km 45	Castelo Branco	Idanha-A-Nova	2016-11-06, 09:30 a 2016-11-06, 13:00
Condicionamento	A1 ao km 298	Porto	Vila Nova De Gaia	2016-10-01, 17:27
Trabalhos	N327 ao km 24	Aveiro	Oliveira De Azeméis	2016-10-01, 16:00 a 2016-10-01, 19:00
Trabalhos	N223 ao km 12	Aveiro	Santa Maria Da Feira	2016-10-01, 16:00 a 2016-10-01, 19:00
Trabalhos	N118 ao km 162	Portalegre	Gavião	2016-10-01, 01:00 a 2016-10-03, 00:59
Trabalhos	N244 ao km 58	Portalegre	Gavião	2016-10-01, 01:00 a 2016-10-03, 00:59
Condicionamento	IC5 ao km 152	Bragança	Vila Flor	2016-09-30, 12:12
Trabalhos	N333-3 ao km 9	Viseu	Oliveira De Frades	2016-09-30, 10:00

Figura 3: As ocorrências retiradas da página do estradas.pt

Na Listagem 4.4 é caracterizada a função *scrapeWebsite()*. O objetivo final da função é preencher uma lista com as ocorrências que estão na tabela do site, visível na Fig. 3. Um WebDriver é iniciado com a criação de um novo *FirefoxDriver*, para em seguida ser feita a recolha da página do Estradas.pt (linhas 6 e 7). É feita uma espera para confirmar a existência do elemento *select* que tem o *tagName* "example_length" no HTML da página (linha 9). Em seguida é executado um Script em Javascript que vai mudar o valor do elemento "example_length" para 1000 e é simulada a mudança do número de registos ficando assim a tabela com todos os registos (linhas 11-14). A string *tabela* vai ser atribuída à tabela que

4.2. Fontes consultadas

foi extraída (linha 16). A função *extrairOcorrencia* vai ser apresentada na próxima listagem (linha 20).

```
1 private final String url = "http://www.estradas.pt/Informacoes/Ocorrencias";
2 public List<Ocorrencia> lista = new ArrayList<>();
3
4 public List<Ocorrencia> scrapeWebsite() {
5
6     WebDriver driver = new FirefoxDriver();
7     driver.get(url);
8
9     (new WebDriverWait(driver, 5)).until(ExpectedConditions.
10         presenceOfElementLocated(By.name("example_length")));
11
12     JavascriptExecutor js = (JavascriptExecutor) driver;
13     js.executeScript(
14         "document.getElementsByName('example_length').item(0).children[0].
15             value = 1000;"
16         + "document.getElementsByName('example_length').item(0).dispatchEvent
17             (new Event('change'))");
18
19     String tabela = driver.findElement(By.id("example")).findElement(By.
20         xpath("tbody")).getAttribute("innerHTML");
21
22     driver.quit();
23
24     extrairOcorrencia(tabela);
25     return lista;
26 }
27 }
```

Listing 4.4: Recolha de Ocorrências

Na Listagem 4.5, a variável *lista* (linha 2 da Listagem 4.4) vai ser preenchida após a execução da função *extrairOcorrencia(String res)*, a string *res* vai conter o resultado da extração. Nesta função é feito o tratamento da informação recolhida de forma a preencher a classe *Ocorrência* que tem como variáveis os campos da tabela (Tipo, Estrada, Concelho, Distrito e Data). Com o uso do *jsoup* vai ser feito o parsing da string com a tabela para um documento DOM (linha 2). Em seguida são separadas as linhas da tabela na coleção *Elements* (linha 3) que são iteradas para retirar cada conteúdo das células da tabela (linhas 5-13). Assim é preenchido o *ArrayList* com ocorrências.

```
1 public void extrairOcorrencia(String res) {
2     Document doc = Jsoup.parse(xmlRecord, "", Parser.xmlParser());
```

4.2. Fontes consultadas

```
3     Elements elementsByTag = doc.getElementsByTag("tr");
4
5     for (Element element : elementsByTag) {
6         Elements elementsByTag1 = element.getElementsByTag("td");
7         lista.add(new Ocorrencia(
8             elementsByTag1.get(0).text(), // tipo
9             elementsByTag1.get(1).text(), // estrada
10            elementsByTag1.get(2).text(), // concelho
11            elementsByTag1.get(3).text(), // distrito
12            elementsByTag1.get(4).text() // data
13        ));
14    }
15 }
```

Listing 4.5: Parsing da string com a tabela de ocorrências

O facto do site Estradas de Portugal não ter uma API pública que facilitasse a recolha da informação que era pretendida, tal como o facto da página onde estava a tabela ser dinamicamente gerada, levou ao uso de uma ferramenta que possibilitasse a extração de conteúdo gerado dinamicamente. O Selenium forneceu esta opção, permitindo assim que o material recolhido fique guardado num *ArrayList* contendo elementos do tipo *Ocorrencia*.

4.2.2 Instituto Português do Mar e da Atmosfera

No site do Instituto Português do Mar e da Atmosfera (IPMA) é fornecida uma vasta variedade de informação. Com a tentativa falhada da recolha das temperaturas individuais das localidades de Portugal Continental, foram procuradas outras informações do website que fosse possível extrair com recurso a Web scraping devido à ausência de API. Foram escolhidos os alertas meteorológicos e os avisos de atividades sísmicas para o enriquecimento da base de dados do serviço SRH.

Alertas meteorológicos

Os dados relativos aos alertas meteorológicos que vão ser recolhidos podem ser vistos na Figura 4. No lado direito encontram-se os Distritos de Portugal e os seus alertas correspondentes. Apenas os alertas de Portugal Continental são recolhidos. Como este conteúdo não é gerado dinamicamente, podem ser recolhidos os valores desta coluna com o auxílio do jsoup.

Na Listagem 4.6 é apresentada a função *crawlAvisos()* onde é feita a extração da coluna com os distritos e respectivos alertas. O resultado desta função será uma lista de strings contendo queries para inserção na base de dados. O primeiro passo será recolher a página

4.2. Fontes consultadas

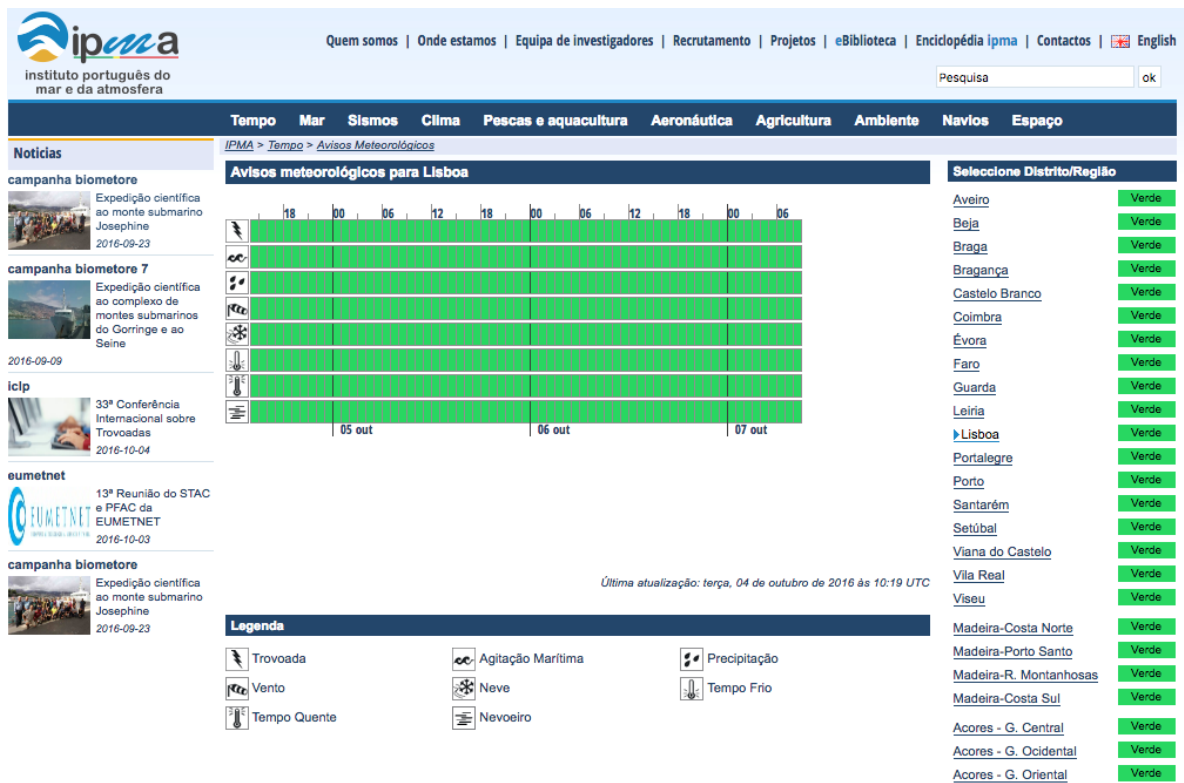


Figura 4: Os Alertas Meteorológicos disponíveis no site do IPMA

da Figura 4 para um documento DOM e selecionar os elementos HTML que se pretendem extrair (linhas 3 e 4). Para cada elemento é selecionada a string do elemento do tipo *anchor*, que representa o nome do distrito, com o *element.select("a").text()*. É feita uma query à base de dados para descobrir o *id* desse distrito, caso exista, para posteriormente ser incluído na string da query de inserção (linha 8). A tabela da base de dados que representa esta informação recolhida vai ter como campos o *id_distrito* e a *cor* do alerta (linha 10).

```

1 public List<String> crawlAvisos() {
2     try {
3         Document doc = Jsoup.connect(avisos).get();
4         Elements links = doc.select("div#RightCol_container > ul > li");
5         BD bd = new BD();
6         for (Element element : links) {
7             if (!element.select("span").isEmpty()) {
8                 int id_distrito = bd.executeQueryR("SELECT id FROM " + BD.
9                     DATABASE + ".distrito WHERE nome LIKE _utf8 '%" + element.
10                        select("a").text() + "%' COLLATE utf8_general_ci");
11                 if (id_distrito != -1) {

```

4.2. Fontes consultadas

```
10         avisoDistrito.add("INSERT INTO "+ BD.DATABASE+".Aviso (
11             id_distrito, cor)VALUES ('" + id_distrito + "',' +
12             element.select("span").text() + "')");
13     }
14     bd.closeConnection();
15 }
16 ...
17 return avisoDistrito;
18 }
```

Listing 4.6: Extração dos alertas meteorológicos

Avisos de atividades sísmicas

No website do IPMA também podem ser recolhidos os avisos relativos a atividades sísmicas em Portugal Continental. Na Figura 5 é apresentada a informação, tal como o aviso que se encontra expandido, que vai ser extraída via Web scraping. O processo é semelhante à recolha dos avisos meteorológicos diferindo no padrão para a seleção de elementos.

The screenshot shows the IPMA website interface. At the top, there is a navigation menu with links for 'Quem somos', 'Onde estamos', 'Equipa de investigadores', 'Recrutamento', 'Projetos', 'eBiblioteca', 'Enciclopédia ipma', 'Contactos', and 'English'. Below the navigation menu is a search bar with the text 'Pesquisa' and an 'ok' button. The main content area is divided into several sections. On the left, there are sections for 'Notícias', 'campanha biomotore 7', 'iclp', 'eumetnet', and another 'campanha biomotore'. The main content area displays a list of seismic alerts. The first alert is expanded, showing the following details:

- Aviso de Sismo Sentido no Arquipélago dos Açores 29-09-2016 11:58**
- 2016-09-29 11:58:00**
- O Instituto Português do Mar e da Atmosfera informa que no dia 29-09-2016 pelas 11:58 (hora local) foi registado nas estações da Rede Sísmica do Arquipélago dos Açores, um sismo de magnitude 2.7 (Richter) e cujo epicentro se localizou próximo de Ribeirinha (Faial).
- Este sismo, de acordo com a informação disponível até ao momento, não causou danos pessoais ou materiais e foi sentido com intensidade máxima V (escala de Mercalli modificada) na região de Ribeirinha, Faial.
- Se a situação o justificar serão emitidos novos comunicados.

Below this expanded alert, there are several other alerts listed, including:

- Aviso de Sismo no Continente 23-09-2016 07:34** (2016-09-23 07:34:00)
- Aviso de Sismo Sentido no Arquipélago dos Açores 22-09-2016 22:19** (2016-09-22 22:19:00)
- Aviso de Sismo no Continente 17-09-2016 20:24** (2016-09-17 20:24:00)
- Aviso de Sismo no Continente 11-09-2016 12:30** (2016-09-11 12:30:47)
- Aviso de Sismo Sentido no Continente 11-09-2016 12:30** (2016-09-11 12:30:00)

Figura 5: Os Avisos de Sismos disponíveis no site do IPMA

4.2. Fontes consultadas

Com a Listagem 4.7 da função `crawlSismos()` é possível notar as semelhanças com a extração dos avisos meteorológicos. É feita a recolha da página e selecionados os elementos sobre os quais é necessário aplicar novas funções (linhas 3 e 4). A função `removeIlhas(String str)` vai testar se o aviso recebido como argumento acontece no Continente ou nos Arquipélagos (linha 8). No caso da função devolver um valor verdadeiro será usada uma outra função auxiliar (`getCidadeFromText(String str)`) onde vai ser retirado o nome da localidade partindo do texto extraído (linha 9). A função `cleanText(String str)` é usada para tratar o texto da descrição do sismo (linha 11). Para finalizar é executada uma query para se obter o `id` da localidade do sismo (linha 12). Uma vez obtido o `id` da localidade vai ser gerada a string para inserção na base de dados na tabela `Sismo` com os campos (`id_cidade`, `descricao`) (linha 14).

```
1 public List<String> crawlSismos() {
2     try {
3         Document doc = Jsoup.connect(site).get();
4         Elements links = doc.select("div#CRcol_container > table >tbody > tr >
           td > table");
5         BD bd = new BD();
6         for (Element element : links) {
7             Elements texto = element.select("table > tbody > tr >td > div.
           divBlock");
8             if (removeIlhas(texto.text())){
9                 String cidade = getCidadeFromText(texto.text());
10                if (!cidade.equals("")) {
11                    String desc = cleanText(texto.text());
12                    int id_cidade = bd.executeQueryR("SELECT id FROM "+ BD.
           DATABASE+".cidade WHERE nome LIKE _utf8 '%" +cidade+"%'
           COLLATE utf8_general_ci");
13                    if (id_cidade!=-1) {
14                        sismos.add("INSERT INTO "+ BD.DATABASE+".Sismo (
           id_cidade, descricao)VALUES ('" + id_cidade + "','
           + desc + "')");
15                    }
16                }
17            }
18        }
19        bd.closeConnection();
20    }
21    ...
22    return sismos;
23 }
```

Listing 4.7: Extração dos avisos de atividades sísmicas

4.2. Fontes consultadas

4.2.3 OpenWeatherMap

O OpenWeatherMap é um serviço online que fornece dados meteorológicos, incluindo informação meteorológica atual, previsões e histórico para desenvolvimento de serviços Web ou aplicações móveis. Fornece uma API com endpoints do tipo JSON, XML e HTML. Os utilizadores podem solicitar informações sobre o tempo atual, previsões estendidas e mapas gráficos (inclui cobertura de nuvens, velocidade do vento, pressão e precipitação).

Este serviço foi escolhido devido à presença de API e à sua facilidade de uso. Para recolher as informações relativas às temperaturas das cidades de Portugal Continental foi usada a opção de criar uma bounding box. Após a recolha das coordenadas que limitam o território Português formando um rectângulo é possível fazer um pedido via API para a recolha das temperaturas dentro dos limites da bounding box.

Na Listagem 4.8 está a função *crawlWeather()* onde é feito o pedido à API do OpenWeatherMap e é recebida a resposta em JSON (linha 4). Esta resposta será submetida a um parsing feito na função *parseWeather(String str)* (linha 13). Aqui vai ser tratada a informação de forma a criar uma lista de temperaturas sendo a classe criada com o construtor *Temperatura(String nome, String descricao, String chuva, String temp, String humidade, String vento)*.

```
1 private final String url = "http://api.openweathermap.org/data/2.5/box/city?
   bbox=-10,36,-6,42,14&cluster=yes";
2 public List<Temperatura> lista = new ArrayList<>();
3
4 public List<Temperatura> crawlWeather(){
5     try {
6         String resultJson = Jsoup.connect(url).ignoreContentType(true).
           execute().body();
7         parseWeather(resultJson);
8     }
9     ...
10    return lista;
11 }
12
13 public void parseWeather(String stg){
14     Temperatura temp;
15     String nome, descricao, chuva, temps, humid, vento;
16     JSONArray list = new JSONObject(stg).getJSONArray("list");
17     for (int i = 0; i < list.length(); i++) {
18         JSONObject arr = list.getJSONObject(i);
19
20         nome = arr.getString("name");
21
22         descricao = arr.getJSONArray("weather").getJSONObject(0).get("
           description").toString();
```


4.2. Fontes consultadas

```

23     chuva = arr.getJSONObject("rain").toString();
24
25     temps = arr.getJSONObject("main").get("temp").toString();
26
27     humid = arr.getJSONObject("main").get("humidity").toString();
28
29     vento = arr.getJSONObject("wind").get("speed").toString();
30
31     temp = new Temperatura(nome, descricao, chuva, temps, humid, vento);
32     lista.add(temp);
33 }
34 }
35 }

```

Listing 4.8: Extração e parsing das temperaturas

4.2.4 Cidades de Portugal

Para uma extração mais completa era necessário garantir primeiramente fossem obtidos dados sobre as localidades de Portugal. Assim para cada informação extraída é possível verificar se está relacionada com Portugal Continental. Com o intuito de fazer uma recolha única de informação sobre as cidades, vilas e distritos recorreu-se à Wikipedia (ver Figura 6 e Figura 7). Mais uma vez foi usado Web scraping para a recolha de informação retirando a informação contida nas células das tabelas.

Lista de cidades em Portugal

Origem: Wikipédia, a enciclopédia livre.

Esta é uma [lista de cidades portuguesas](#). Desde 2013, existem em [Portugal](#) 159 povoações com a categoria de *cidade*:

Cidade		Localização:		Área	Pop.	Freguesias	Data de elevação, reconhecimento, ou 1.ª menção
		Concelho < NUTS III ou ilha	Distrito ou Reg. Aut.				
Abrantes	ABT	Médio Tejo	Santarém		18 450	2 [Expandir]	1916 (14 de junho) ^[1]
Agualva-Cacém	SWT	Grande Lisboa	Lisboa	10,51 km²	81 845	2 [Expandir]	2001 (12 de julho) ^[2]
Águeda	AGD	Baixo Vouga	Aveiro		14 571	1 [Expandir]	1985 (14 de agosto) ^[3]
Albergaria-a-Velha	ALB	Baixo Vouga	Aveiro		8 528	1 [Expandir]	2011 (17 de junho) ^[4]
Albufeira	ABF	Algarve	Faro		19 879	1 [Expandir]	1986 (20 de agosto) ^[5]
Alcácer do Sal	ASL	Alentejo Litoral	Setúbal		6 679	1 [Expandir]	1997 (12 de julho) ^[6]
Alcobaça	ACB	Oeste	Leiria		6 982	3 [Expandir]	1995 (30 de agosto) ^[7]
Alfena	VLG	Grande Porto	Porto	16,7 km²	15 211	1 [Expandir]	2011 (22 de junho) ^[8]
Almada	ALM	Península de Setúbal	Setúbal	13,74 km²	108 615	2 [Expandir]	1973 (14 de junho) ^[9]

Figura 6: As cidades de Portugal

4.2. Fontes consultadas

Lista de vilas em Portugal

Origem: Wikipédia, a enciclopédia livre.

Esta é uma [lista de vilas portuguesas](#). Actualmente (2016), Portugal tem 581 povoações com a categoria de [vila](#):

Brasão	Vila	Concelho	Distrito/ Região Autónoma	NUTS III	Superfície	População	N.º de freguesias	Freguesias que integram a vila	Desde
	A dos Cunhados	Torres Vedras	Lisboa	Oeste	43,98 km²	8459	1	A dos Cunhados	1995 (21 de Junho)
	A dos Francos	Caldas da Rainha	Leiria	Oeste	19,78 km²	1701	1	A dos Francos	2009 (12 de Junho)
	A Ver-o-Mar	Póvoa de Varzim	Porto	Área Metropolitana do Porto	5,21 km²	8675	1	A Ver-o-Mar	2003 (1 de Julho)
	Abragão	Penafiel	Porto	Tâmega	8,87 km²	2341	1	Abragão	2001 (12 de Julho)
	Abraveses	Viseu	Viseu	Dão-Lafões	11,95 km²	8539	1	Abraveses	1997 (24 de Julho)

Figura 7: As vilas de Portugal

A Listagem 4.9 com a função *crawlCidades()* descreve a forma como é recolhida a informação das tabelas das figuras anteriores. Com o auxílio do jsoup é recolhida a página para um documento DOM e selecionada a tabela (linhas 8 e 9). Para cada *Element* são selecionadas as strings que correspondem à cidade e ao distrito (linhas 11 e 12). Em seguida é preenchido um *HashMap* com o Distrito e as suas cidades correspondentes num *ArrayList*. A função *crawlVilas()* (linha 6) funciona de forma similar preenchendo o mesmo *HashMap* com o Distrito e as suas correspondentes Vilas.

```

1 private final String site = "https://pt.wikipedia.org/wiki/
   Lista_de_cidades_em_Portugal";
2 private final String site1 = "https://pt.wikipedia.org/wiki/
   Lista_de_vilas_em_Portugal";
3 private final Map<String, List<String>> distritos = new HashMap<>();
4
5 public Map<String, List<String>> crawlCidades() {
6     crawlVilas();
7     try {
8         Document doc = Jsoup.connect(site).get();
9         Elements links = doc.select("div#mw-content-text > table > tbody >
            tr");
10        for (Element element1 : links) {
11            String cidade = element1.select("td > b > a").text();

```

4.2. Fontes consultadas

```
12     String distrito = element1.select("td > a[title~=Distrito]").
13         text();
14
15     if (!distrito.equals("") && !distrito.equals("Distrito")) {
16         if (distritos.containsKey(distrito)) {
17             distritos.get(distrito).add(cidade);
18         }
19         else{
20             List<String> cidades = new ArrayList<>();
21             cidades.add(cidade);
22             distritos.put(distrito, cidades);
23         }
24     }
25 }
26 ...
27 }
```

Listing 4.9: Extração das cidades de Portugal

Após a recolha de informação proveniente de diversas fontes ser efetuada, é simultaneamente feita a filtragem de informação não relevante para o sistema através das verificações feitas para saber se a localização de um determinado dado recolhido é pertencente a uma localidade em Portugal. O sistema encontra-se na fase da Figura 8, ou seja já se encontra feita a recolha e filtragem da informação.

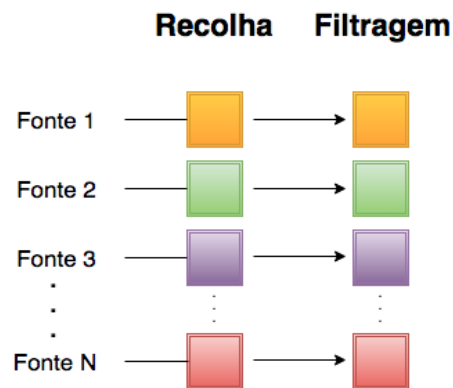


Figura 8: Modelo da recolha e filtragem da informação

4.3. Normalização da informação

4.3 NORMALIZAÇÃO DA INFORMAÇÃO

Devido ao facto de os dados recolhidos serem provenientes de fontes diferentes foi necessário encontrar uma forma de os relacionar. Toda a informação recolhida tem associada uma localização, o que permitiu fazer a interligação dos dados recolhidos.

Numa fase inicial, antes da recolha de todas as cidades de Portugal, os avisos meteorológicos eram relacionados com as ocorrências na estrada através do atributo comum *Distrito* enquanto os registos de atividades sísmicas tinham como referência comum a *Cidade* com os registos de temperaturas. Para melhorar a forma como estes dados se relacionam foi acrescentada informação relativa às cidades e os seus respetivos distritos, uma vez que a informação associada a um distrito também vai afetar todas as suas cidades correspondentes.

Numa fase intermédia quando eram extraídos dados com o atributo *Cidade* eram consultadas as cidades de Portugal previamente recolhidas, caso os campos não coincidissem era descartada a informação extraída. Para evitar a perda de informação foi acrescentada ao serviço SRH a extração das vilas de Portugal. Com o acréscimo das vilas à base de dados a tabela que continha cidades passou a ter vilas também tornando-se assim uma tabela com localidades. Desta forma um Distrito tem associadas a si múltiplas localidades.

4.3.1 Base de dados

Para a melhor compreensão da estrutura da base de dados do tipo MySQL deste serviço é necessária a visualização da sua constituição (Figura 9). Toda a informação que é extraída é guardada nesta base de dados. Apenas as tabelas Cidade e Distrito são preenchidas numa única recolha, todas as outras fazem uma recolha calendarizada.

A normalização da informação recolhida levou às decisões tomadas na construção desta base de dados relacional. A tabela Cidade vai representar as localidades de Portugal, tanto vilas como cidades, sendo que múltiplas localidades têm o mesmo Distrito. As tabelas Sismo e Temperatura têm uma cidade associada, enquanto que as tabelas Aviso e Estrada estão ligadas a Distrito.

A tabela Estrada possui particularidades especiais visto que todos os seus registos vão ser usados como marcadores num mapa. A sua ligação com as outras tabelas não possui grande relevância, sendo o campo distrito escolhido arbitrariamente para relacionar a tabela. Os campos *lat* e *lon* foram acrescentados para representarem a latitude e longitude da ocorrência. O processo de recolha destes valores é feito após a recolha de toda a informação e antes da inserção dos registos na base de dados.

Com o uso do serviço de geocoding via GoogleMaps é possível recolher a latitude e longitude dada uma morada. A junção dos campos inicialmente recolhidos (Estrada, Distrito,

4.3. Normalização da informação

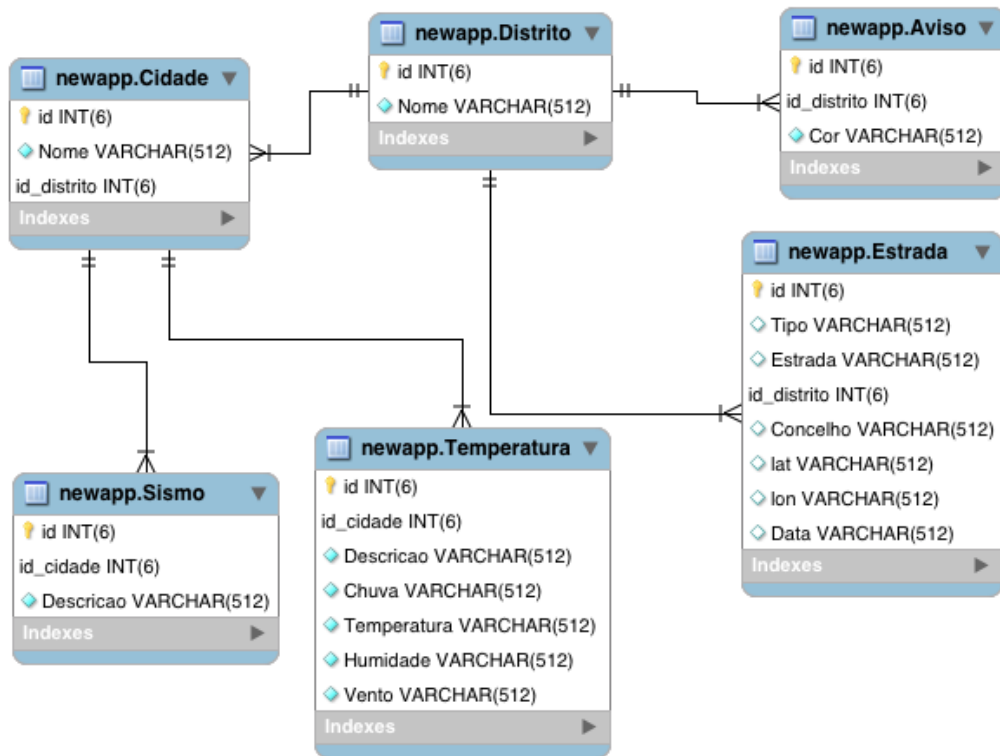


Figura 9: Diagrama da Base de Dados

Concelho) forma uma morada que vai ser enviada conjuntamente com o pedido à API do GoogleMaps (linha 7) e em seguida são tratados os resultados de forma a obter a latitude e longitude de um lugar (linhas 10-18), como é visível na Listagem 4.10.

```
1 public void getGeoPointFromAddress() {
2     String locationAddres = address.replaceAll(" ", "%20");
3     String str = "https://maps.googleapis.com/maps/api/geocode/json?address="
4         + locationAddres + "&sensor=true";
5     JSONObject json, geoMetryObject, locations;
6     try {
7         String ss = Jsoup.connect(str).ignoreContentType(true).execute().
8             body();
9         json = new JSONObject(ss);
10        if (!json.get("status").equals("OVER_QUERY_LIMIT")) {
11            JSONArray jarr = json.getJSONArray("results");
12            int i;
13            for (i = 0; i < jarr.length(); i++) {
14                json = jarr.getJSONObject(i);
15                geoMetryObject = json.getJSONObject("geometry");
```

4.3. Normalização da informação

```
15         locations = geoMetryObject.getJSONObject("location");
16         this.lat = locations.get("lat").toString();
17         this.lon = locations.get("lng").toString();
18     }
19 }
20 }
21 ...
22 }
```

Listing 4.10: Geocoding de uma localização

Para cada extração é gerada uma query SQL para a inserção da informação recolhida na base de dados. Nos casos da recolha de temperaturas e de ocorrências das estradas na classe respetiva existe um método *getQuery()* que tem como resultado a query de inserção (ver Listagem 4.11). Com o auxílio de uma classe que faz a ligação à base de dados (a classe *BD*) é feita uma query do tipo *SELECT* para recolher o *id* da localidade da temperatura em questão (linha 3). Caso seja obtido um resultado positivo vai ser construída a query de inserção com os valores correspondentes (linhas 10 e 11).

```
1  public String getQuery() {
2      BD bd = new BD();
3      int id_cidade = bd.executeQueryR("SELECT id FROM "+ BD.DATABASE+".cidade
4          WHERE nome LIKE _utf8 '%" +this.nome+"%' COLLATE utf8_general_ci");
5      bd.closeConnection();
6      String result;
7      if (id_cidade!=-1) {
8          result = "";
9      }
10     else{
11         result = "INSERT INTO "+ BD.DATABASE+".Temperatura (id_cidade,
12             Descricao, Chuva, Temperatura, Humidade, Vento)"
13         + "VALUES ('" + id_cidade + "','"+ this.descricao + "','"+ this.
14             chuva + "','"+ this.temp + "','"+ this.humidade + "','"+ this
15             .vento + "')";
16     }
17     return result;
18 }
```

Listing 4.11: Geração de uma query de inserção

4.3. Normalização da informação

Após o preenchimento da base de dados uma parte do sistema encontra-se desenvolvida. Na Figura 10 é possível ver qual a parte do sistema desenvolvida, relativamente à arquitetura apresentada na Fig. 1. No próximo capítulo é descrita a tecnologia usada para possibilitar o uso da informação contida na base de dados.

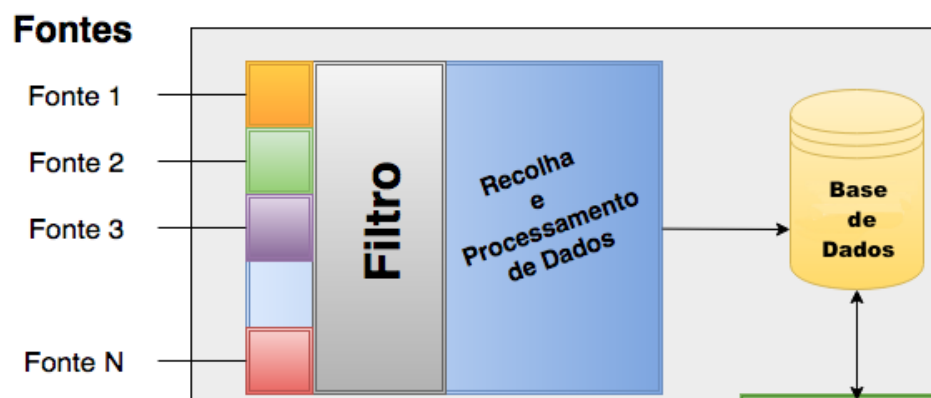


Figura 10: Parte da arquitectura do sistema SRH

SISTEMA WEB

Depois de realizada a extração de informação (ver Cap. 4) foi utilizada uma plataforma que permite o desenvolvimento de um servidor e um cliente que comunicam via API para exposição da informação recolhida. Neste capítulo será apresentada a interface Web criada para que a visualização do material recolhido esteja disponível para qualquer utilizador. No próximo capítulo (ver Cap. 6) é apresentada a aplicação Android que vai permitir acesso à informação recolhida num dispositivo móvel.

Na primeira secção (Sec. 5.1) descreve-se o interpretador de código *Node.js* que usa a linguagem Javascript para o desenvolvimento Web. Na secção seguinte (Sec. 5.2) é introduzida a plataforma escolhida para o servidor e para a interface Web, o StrongLoop LoopBack. Na secção (Sec. 5.3) serão demonstrados os métodos da API do GoogleMaps que foram usados na interface Web. Por fim, na secção 5.4 é demonstrada a interface Web desenvolvida.

5.1 NODE.JS

Dentro do desenvolvimento de *software*, *Node.js* é um interpretador open-source de código Javascript para a implementação do lado do servidor de aplicações Web. O seu objetivo é ajudar programadores na criação de aplicações altamente escaláveis (como um servidor web), com códigos capazes de manipular dezenas de milhares de conexões simultâneas, numa única máquina física. O *Node.js* tem uma arquitetura orientada ao evento com a capacidade de Input/Output assíncrono. Esta escolha de design foi feita com o objetivo de otimizar o *throughput* e escalabilidade em aplicações Web com múltiplas operações de input/output, tal como aplicações Web em tempo real. No entanto *Node.js* não é uma plataforma de Javascript, muitos dos seus módulos básicos são escritos em Javascript, e os programadores podem escrever novos módulos em Javascript. O *Node.js* é baseado no interpretador V8 JavaScript Engine (interpretador de JavaScript open source implementado pelo Google em C++ e utilizado pelo Chrome) (Wikipedia, 2016d).

Atualmente, o *Node.js* está cada vez mais a ser usado por causa da utilização do Javascript como linguagem de programação estar a crescer consideravelmente. Anteriormente o

5.2. StrongLoop LoopBack

Javascript era apenas usado no browser e para programar no lado do servidor teria de ser usada outra linguagem como o PHP.

5.2 STRONGLOOP LOOPBACK

LoopBack é uma plataforma open source completamente extensível, distribuída pela empresa StrongLoop, baseada em *Express*¹ otimizado para a construção de APIs para plataformas móveis, Web e outros dispositivos. Faz a ligação com múltiplas fontes de dados, permite a escrita da lógica de negócio em *Node.js*, integra-se em cima de serviços e dados já existentes (LoopBack, 2016; StrongLoop, 2016).

A plataforma do LoopBack é constituída por um conjunto de módulos *Node.js* que podem ser usados independentemente ou em conjunto para uma construção rápida de aplicações que podem ser expostas a APIs REST. Uma aplicação interage com a fontes de dados através do modelo API do LoopBack, disponível localmente no *Node.js*, remotamente com REST, e via API de clientes nativos para iOS, Android e HTML5. O uso destas APIs permite que as aplicações possam executar queries na base de dados, armazenar dados, fazer upload de ficheiros, enviar emails, criar notificações do tipo *push*, registar utilizadores e várias ações fornecidas pelas fontes de dados e serviços.

Aplicações do tipo Cliente podem aceder à API do LoopBack diretamente com o uso de *Strong Remoting*, uma camada de transporte conectável que permite fornecer APIs de back-end sobre REST, *WebSockets*, entre outros.

Algumas das funcionalidades que esta plataforma permite:

- Criação de APIs REST dinâmicas de ponta a ponta;
- Acesso a fontes de dados da Oracle, MySQL, PostgreSQL, MS SQL Server, MongoDB, SOAP e outras REST APIs;
- Incorporação modelos de dados relacionados;
- Uso de serviços incorporados de push, geolocalização e ficheiros para aplicações móveis;
- Criação de aplicações do lado cliente com o uso de Android, iOS, e JavaScript SDKs.

De seguida apresentamos os conceitos fundamentais do *LoopBack*, nomeadamente:

- Modelos
- Fontes de Dados e conectores
- Componentes do LoopBack

¹ Express é uma plataforma para aplicações Web minimalista e flexível em Node.js, e fornece um conjunto robusto de recursos para Web e aplicações móveis.

5.2. StrongLoop LoopBack

Modelos

Os Modelos são o coração do LoopBack e por isso representam o *back-end* das fontes de dados como bases de dados ou outros serviços de *back-end* (REST, SOAP). Modelos no LoopBack são objetos *Javascript* com *Node* e APIs REST. A particularidade mais importante do LoopBack é que na definição de um modelo, este fica automaticamente associado a uma API REST predefinida com múltiplas operações de criação, leitura, atualização e remoção.

Os modelos têm uma hierarquia de herança visível na Figura 11. O objeto do modelo Básico contém métodos para a adição de *hooks* e validação de dados. Todos os outros objetos de modelo herdam estes métodos. Quando é adicionado um modelo a uma fonte de dados persistente passa a ser um modelo ligado com operações de criação, recolha, atualização e remoção. Os modelos internos (built-in) do LoopBack herdam estas operações (StrongLoop, 2016).

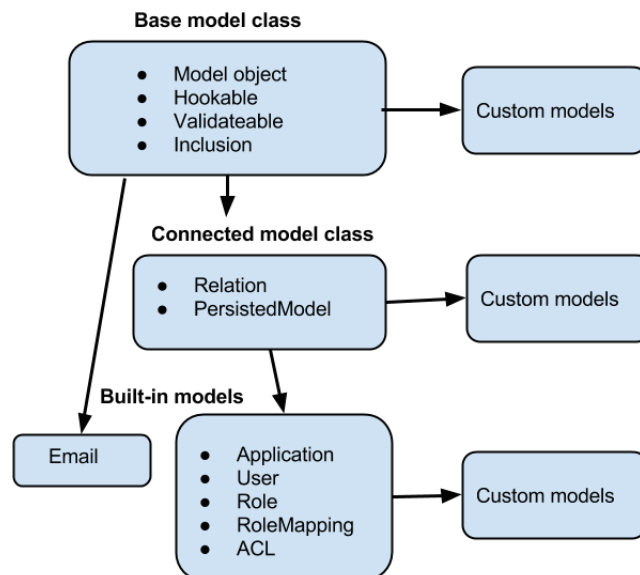


Figura 11: Hierarquia de Herança do Modelo (StrongLoop, 2016)

Os modelos built-in ou internos existem em qualquer aplicação LoopBack como User, Role e Application, para que o programador não necessite criar estes modelos comuns de raiz. Modelos personalizados (Custom models) podem ser criados especificamente para a aplicação ou estendidos dos modelos internos.

5.2. StrongLoop LoopBack

Fontes de Dados e conectores

O LoopBack generaliza serviços de back-end como base de dados, REST APIs, serviços SOAP Web e serviços de armazenamento como fontes de dados. As fontes de dados são suportadas por conectores que comunicam diretamente com a base de dados ou outro serviço de back-end. As aplicações não usam os conectores diretamente, invés vão através das fontes de dados usando as APIs *PersitedModel* e *DataSource* (StrongLoop, 2016).

Componentes do LoopBack

Os componentes do LoopBack fornecem as seguintes funcionalidades *plug-in* adicionais: (StrongLoop, 2016)

- Notificações *Push* - permitem enviar informações adicionais para aplicações móveis para o *display* imediato num *badge*, alerta ou mensagem *pop-up* no dispositivo móvel.
- Serviço de Armazenamento - permite o *upload* e *download* de ficheiros para o armazenamento de provedores de armazenamento *cloud* (Amazon, Rackspace, Openstack, e Azure) tal como o sistema de ficheiros do servidor.
- *Thrid-party login* - integra *Passport* e permite login de um utilizador (ou ligação de uma conta) usando credenciais de terceiros via Facebook, Google, Twitter, Github, ou qualquer sistema que suporte OAuth, OAuth 2, ou OpenID.
- Sincronização - permite que as aplicações móveis operem *offline* e depois sincronizem os dados com a aplicação do servidor quando houver uma recoserão.
- *OAuth 2.0* - permite que as aplicações LoopBack que funcionem como provedores de *OAuth 2.0* para autenticar e autorizar aplicações cliente e utilizadores terem acesso protegido a API endpoints.

5.2.1 *Configurações*

O LoopBack oferece múltiplas funcionalidades de base e são acrescentadas configurações de forma a fazer a integração com a base de dados. Foram definidos os modelos de acordo com as tabelas na base de dados, tal como as suas relações. Estes modelos são comuns para a parte de back-end e front-end. No back-end é definida a localização da base de dados, neste caso é local, para que seja possível consultar toda a informação previamente extraída. É feita a migração da informação relativa aos utilizadores do sistema (Users) para a base de dados local. Esta alteração vai permitir que sejam guardados os utilizadores.

Na Listagem 5.1 é apresentado o modelo, em JSON, para as temperaturas. As propriedades nele definidas correspondem aos campos da tabela Temperatura na base de dados

5.2. StrongLoop LoopBack

(linhas 8-29). É definida uma relação entre o modelo Temperatura e o modelo Cidade (linhas 32-36). Uma temperatura tem a si associada uma chave estrangeira para o *id_cidade* que pertence a uma localidade.

```
1 {
2   "name": "Temperatura",
3   "base": "PersistedModel",
4   "idInjection": true,
5   "options": {
6     "validateUpsert": true
7   },
8   "properties": {
9     "Descricao": {
10      "type": "string",
11      "required": true
12    },
13    "Chuva": {
14      "type": "string",
15      "required": true
16    },
17    "Temperatura": {
18      "type": "string",
19      "required": true
20    },
21    "Humidade": {
22      "type": "string",
23      "required": true
24    },
25    "Vento": {
26      "type": "string",
27      "required": true
28    }
29  },
30  "validations": [],
31  "relations": {
32    "cidade": {
33      "type": "belongsTo",
34      "model": "Cidade",
35      "foreignKey": "id_cidade"
36    }
37  },
38  "acls": [],
39  "methods": {}
40 }
```

Listing 5.1: Modelo da entidade Temperatura

5.3. GoogleMaps

5.3 GOOGLEMAPS

No desenvolvimento do cliente HTML do sistema Web era necessário recorrer ao uso de um mapa que possibilitasse a criação de percursos de viagens para os utilizadores do serviço SRH. Como o GoogleMaps tem disponível uma API pública foi o serviço escolhido para ser integrado no sistema Web.

O GoogleMaps fornece múltiplas funcionalidades, no nosso sistema, foram usados os serviços de Direções e de Place Autocomplete. Para ser possível a definição de um percurso com uma origem e um destino é necessário o uso destes serviços.

Na definição de um percurso seguro seria ideal fazer o desvio de situações perigosas para o utilizador. Com a informação recolhida sobre as ocorrências nas estradas de Portugal seria possível fornecer os pontos por onde o percurso não deveria passar mas não está disponível nenhuma funcionalidade do GoogleMaps que permita a criação de um percurso sem passar por lugares pré definidos. Em alternativa no mapa será feita a colocação de marcadores que representam as ocorrências nas estradas, onde será possível ao utilizador alterar manualmente o seu percurso para se desviar destes marcadores.

Place Autocomplete

O serviço de Place Autocomplete permite que um utilizador escreva o nome de local e seja sugerido o nome completo do local. Esta funcionalidade é usada no sistema para completar automaticamente a origem e o destino de um percurso definido pelo utilizador.



Figura 12: Exemplo do Autocomplete

5.4. Interface Web

Direções

O serviço *Directions* do GoogleMaps vai permitir a criação de percursos dada uma origem, destino e o modo de locomoção. Este parâmetros vão ser definidos pelo utilizador na interface Web, em seguida é feito o pedido de recolha do percurso ao serviço Direções. O resultado será apresentado no mapa juntamente com uma lista de direções do percurso escolhido.

5.4 INTERFACE WEB

Após o desenvolvimento da API REST foi gerado um Cliente em AngularJS que vai consumir a API REST criada. O AngularJS é uma plataforma Javascript open-source, construída com o padrão MVC (model-view-controller) para o desenvolvimento front-end de aplicações. O Loopback AngularJS SDK vai fornecer a capacidade para gerar um serviço AngularJS baseado no modelo Node.js usando as ferramentas provenientes do SDK (Software development kit). O SDK é um conjunto de ferramentas de desenvolvimento de software que permite a criação de aplicações para uma determinada plataforma de desenvolvimento.

5.4.1 *Resultado do Cliente em AngularJS*

As figuras seguintes vão representar o resultado do Cliente em AngularJS, onde os utilizadores autenticados podem visualizar toda a informação que foi recolhida na fase de extração de informação (ver Sec. 4.2).

Uma vez feito o registo e a autenticação do utilizador no sistema, é apresentada a barra de navegação da Figura 13. O email do utilizador que se encontra autenticado e a navegar o sistema é mostrado juntamente com as opções de navegação disponíveis.



Figura 13: Barra de navegação

No separador Ocorrências apresentado na Figura 14 é possível verificar todas as ocorrências nas estradas de Portugal (ver Sec. 4.2.1) em formato de tabela. É fornecida uma barra de procura para facilitar a filtragem da informação.

5.4. Interface Web

SRH

Olá foo@bar.com

Ocorrências Temperaturas Sismos Mapa Sair

Q

Tipo ▾	Estrada	Localidade	Distrito	Data
Acidente	A43 ao km 1	Porto	Porto	2016-10-12, 16:41 a 2016-10-12, 16:41
Acidente	N125 ao km 36	Portimão	Faro	2016-10-12, 15:30 a 2016-10-12, 15:30
Condicionamento	IP2 ao km 34	Bragança	Bragança	2016-09-14, 12:57 a 2016-09-14, 12:57
Condicionamento	IC5 ao km 154	Vila Flor	Bragança	2016-09-01, 14:25 a 2016-09-01, 14:25
Condicionamento	IP2 ao km 114	Trancoso	Guarda	2016-08-29, 14:42 a 2016-08-29, 14:42
Condicionamento	IP2 ao km 90	Mêda	Guarda	2016-08-05, 13:31 a 2016-08-05, 13:31
Outros	N239 ao km 45	Idanha-A-Nova	Castelo Branco	2016-11-06, 09:30 a 2016-11-06, 13:00
Trabalhos	N395 ao km 53	Albufeira	Faro	2016-10-12, 10:00 a 2016-10-14, 19:00
Trabalhos	M18 ao km 46	Covilhã	Castelo Branco	2016-10-10, 11:39 a 2016-10-21, 20:00
Trabalhos	N9 ao km 57	Torres Vedras	Lisboa	2016-10-10, 11:00 a 2016-10-21, 18:00
Trabalhos	N2 ao km 692	Loulé	Faro	2016-10-10, 10:22 a 2016-10-12, 19:00

Figura 14: Separador com a tabela de ocorrências

No separador Temperaturas apresentado na Figura 15 é possível verificar todas as temperaturas de Portugal recolhidas via OpenWeatherMap (ver Sec. 4.2.3) juntamente com a informação de atividades sísmicas e alertas meteorológicos (ver Sec. 4.2.2) em formato de tabela. É fornecida uma barra de procura para facilitar a filtragem da informação.

SRH

Olá foo@bar.com

Ocorrências **Temperaturas** Sismos Mapa Sair

Q

Localidade	Descrição	Volume de Chuva (3h)	Graus	Humidade (%)	Vento (m/s)	Alerta	Sismo ▾
Faro	Algumas nuvens		18.98	91	5.47	Amarelo	O Instituto Português do Mar e da Atmosfera informa que no dia 01-08-2016 pelas 04:48 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 4.0 (Richter) e cujo epicentro se localizou a cerca de 110 km a Sul de Faro
Sesimbra	Algumas nuvens		18.28	96	6.83	Amarelo	O Instituto Português do Mar e da Atmosfera informa que no dia 14-05-2016 pelas 07:58 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 3.5 (Richter) e cujo epicentro se localizou a cerca de 55 km a Sudoeste de Sesimbra
Peniche	Chuva leve	0.195	16.63	100	4.37	Amarelo	O Instituto Português do Mar e da Atmosfera informa que no dia 19-08-2016 pelas 12:44 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 4.1 (Richter) e cujo epicentro se localizou a cerca de 80 km a Oeste de Peniche
Évora	Chuva leve	0.875	17.48	92	7.17	Amarelo	
Corroios	Chuva intensa	12.3	15.43	100	4.92	Amarelo	

Figura 15: Separador com a tabela de temperaturas

5.4. Interface Web

No separador Sismos apresentado na Figura 16 são apresentados os dados relativos às atividades sísmicas em Portugal Continental. É também fornecida uma barra de procura para facilitar a filtragem da informação.

SRH

Olá foo@bar.com

Ocorrências Temperaturas **Sismos** Mapa Sair

Q Search

Localidade	Sismo ▾
Faro	O Instituto Português do Mar e da Atmosfera informa que no dia 01-08-2016 pelas 04:48 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 4.0 (Richter) e cujo epicentro se localizou a cerca de 110 km a Sul de Faro
Viana do Alentejo	O Instituto Português do Mar e da Atmosfera informa que no dia 03-11-2015 pelas 02:43 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 2.8 (Richter) e cujo epicentro se localizou a cerca de 6 km a Oeste de Viana do Alentejo
Ponte de Lima	O Instituto Português do Mar e da Atmosfera informa que no dia 04-08-2016 pelas 18:44 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 2.3 (Richter) e cujo epicentro se localizou a cerca de 8 km a Sudoeste de Ponte de Lima
Monchique	O Instituto Português do Mar e da Atmosfera informa que no dia 10-10-2016 pelas 16:50 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 2.6 (Richter) e cujo epicentro se localizou a cerca de 8 km a Sul de Monchique
Sesimbra	O Instituto Português do Mar e da Atmosfera informa que no dia 14-05-2016 pelas 07:58 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 3.5 (Richter) e cujo epicentro se localizou a cerca de 55 km a Sudoeste de Sesimbra
Peniche	O Instituto Português do Mar e da Atmosfera informa que no dia 19-08-2016 pelas 12:44 (hora local) foi registado nas estações da Rede Sísmica do Continente, um sismo de magnitude 4.1 (Richter) e cujo epicentro se localizou a cerca de 80 km a Oeste de Peniche

Figura 16: Separador com a tabela de sismos

No separador Mapa apresentado na Figura 17, existem várias funcionalidades e informações. É disponibilizado um mapa com opções para mudar o tipo de terreno. São colocados marcadores no mapa que representam as ocorrências nas estradas de Portugal. É fornecida a opção de mostrar no mapa a localização atual do utilizador para este poder saber quais as ocorrências que o rodeiam.

Na Figura 18 é possível ver, na continuação da página da Figura 17, as opções de inserção da origem, do destino e do método de viagem para o qual se pretende obter direções. Quando forem introduzidos os lugares de origem e destino, vai ser sugerido através do uso do Autocomplete (Sec. 5.3), o resto da localização.

A título de exemplo, foi escolhido um percurso com origem em Braga e Barcelos como destino sendo a viagem feita de carro. Na Figura 19 é demonstrado o resultado do percurso pedido. No mapa é possível ver que existem algumas ocorrências no percurso escolhido, o utilizador pode escolher uma das alternativas ou arrastar a linha do percurso de forma a evitar estes condicionamentos na estrada. São apresentadas também as temperaturas nos locais de origem e destino em forma de tabela. Estas informações vão permitir o planeamento de uma viagem em segurança.

5.4. Interface Web

SRH

Olá foo@bar.com

Ocorrências Temperaturas Sismos **Mapa** Sair

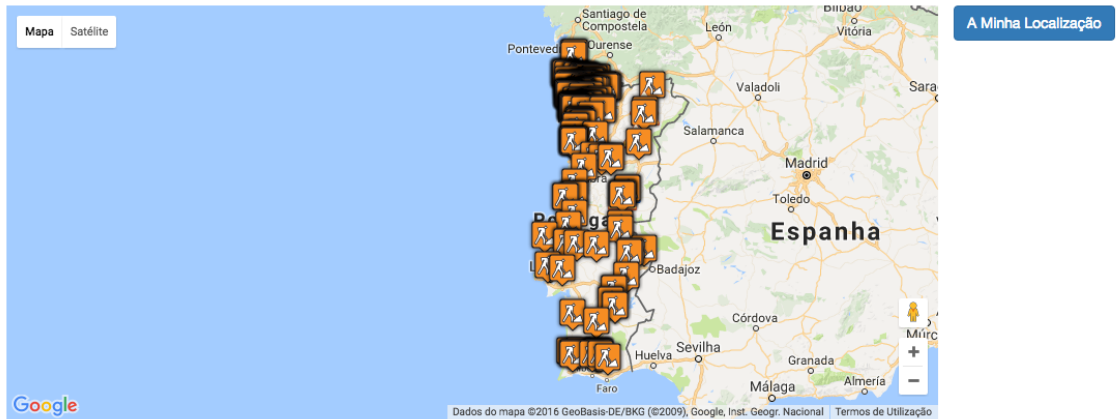


Figura 17: Separador com o mapa

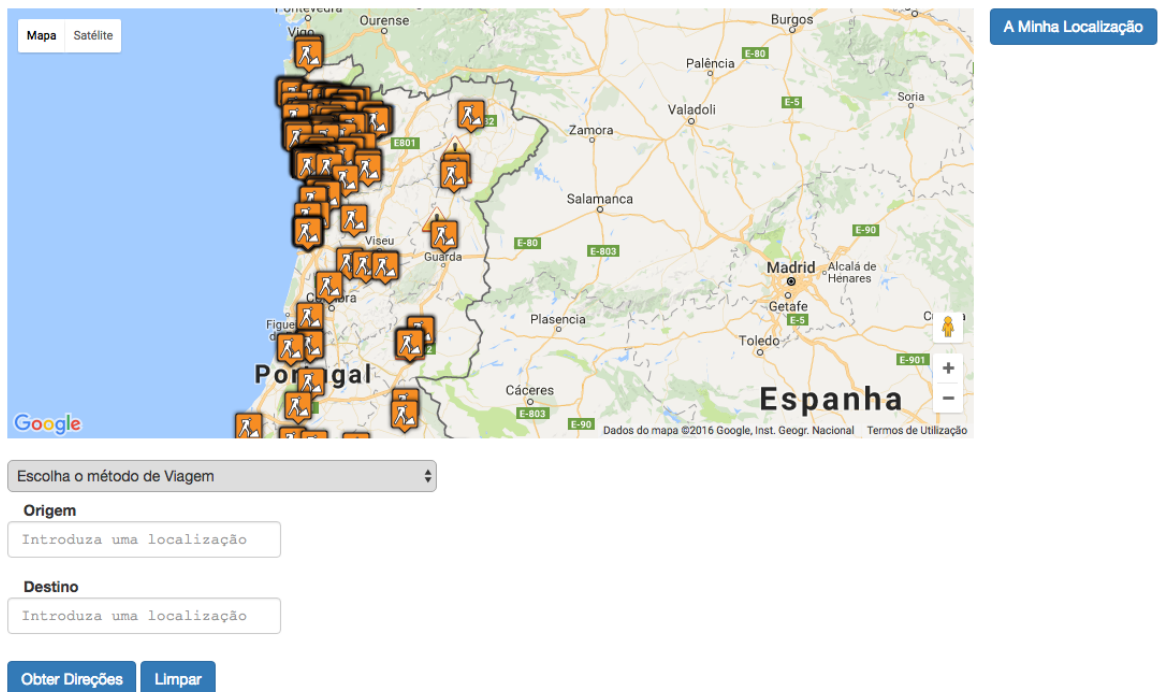
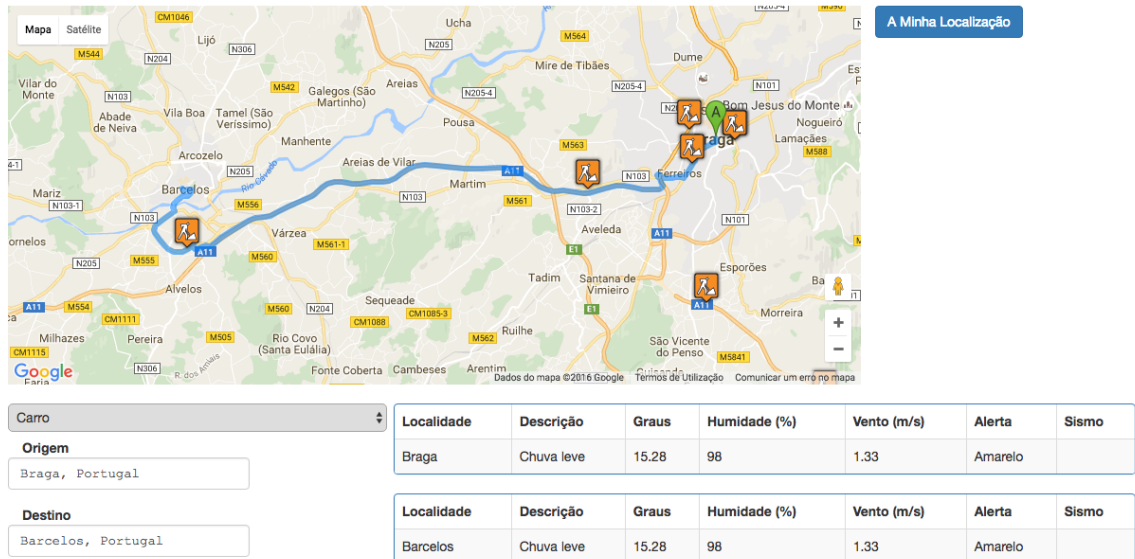


Figura 18: Continuação do separador com o mapa

A Figura 20 é a continuação da figura anterior. Aqui são mostradas as direções do percurso escolhido, tal como as alternativas possíveis com o tempo de viagem correspondente.

5.4. Interface Web

Ocorrências Temperaturas Sismos **Mapa** Sair



A Minha Localização

Carro

Origem
Braga, Portugal

Destino
Barcelos, Portugal

Localidade	Descrição	Graus	Humidade (%)	Vento (m/s)	Alerta	Sismo
Braga	Chuva leve	15.28	98	1.33	Amarelo	

Localidade	Descrição	Graus	Humidade (%)	Vento (m/s)	Alerta	Sismo
Barcelos	Chuva leve	15.28	98	1.33	Amarelo	

Figura 19: Exemplo de um percurso escolhido

Carro

Origem
Braga, Portugal

Destino
Barcelos, Portugal

Localidade	Descrição	Graus	Humidade (%)	Vento (m/s)	Alerta	Sismo
Braga	Chuva leve	15.28	98	1.33	Amarelo	

Localidade	Descrição	Graus	Humidade (%)	Vento (m/s)	Alerta	Sismo
Barcelos	Chuva leve	15.28	98	1.33	Amarelo	

Obter Direções Limpas

Trajetos sugeridos
 A11 22,2 km. Cerca de 23 min.
 N103 18,9 km. Cerca de 28 min.
 N205 28,6 km. Cerca de 35 min.

R. de São Geraldo 70-96, 4700 Braga, Portugal

22,2 km. Cerca de 23 min.

1. Siga para sul na R. de São Geraldo em direção a Av. da Imac. Conceição/N103 0,1 km
2. Vire à direita em direção a Av. da Imac. Conceição/N103 0,4 km
3. Mantenha-se à esquerda para continuar na Av. da Imac. Conceição/N103 0,5 km
Continue a seguir N103
4. Na rotunda, siga pela 2.ª saída para Av. Cidade do Porto/N14 0,6 km
5. Na rotunda, siga pela 1.ª saída para R. Frei José Vilaça 0,2 km
6. Na rotunda, siga pela 2.ª saída e continue na R. Frei José Vilaça 0,5 km
7. Na rotunda, siga pela 2.ª saída para N103 0,1 km
8. Siga pela rampa para A3/A11/N14/N101/Porto/Valença/Guimarães/Barcelos 0,3 km
9. Convergir com Barcelos/Porto/A11/A3 0,2 km

Figura 20: Direções do percurso escolhido

APLICAÇÃO ANDROID

Com o servidor e aplicação Web já apresentados no Capítulo 5, vai ser demonstrada a aplicação Android neste capítulo.

Na primeira secção (Sec. 6.1) é descrito o ambiente de desenvolvimento usado para a criação da aplicação Android. Na secção (Sec. 6.2) seguinte vão ser explicitados os serviços da GoogleMaps que foram usados. Na secção (Sec. 6.3) é apresentada a forma de como foi integrada a informação do servidor na aplicação móvel. Por fim na secção (Sec. 6.4) é demonstrada a aplicação desenvolvida.

6.1 ANDROID STUDIO

O Android Studio foi o Integrated Development Environment (IDE) escolhido como plataforma de desenvolvimento da aplicação Android. Múltiplos recursos são disponibilizados para a criação de aplicações Android, como:

- Um sistema de compilação flexível baseado no Gradle;
- Um emulador rápido com múltiplos recursos;
- Instant Run para enviar alterações a aplicações em execução sem compilar um novo Android Package (APK);
- Ferramentas e estruturas de teste abrangentes.

O Android Studio usa o Gradle como base do sistema de compilação. Este sistema é executado como uma ferramenta integrada no menu do Android Studio e de forma independente na linha de comandos. É possível usar os recursos do Gradle para:

- Personalizar, configurar e ampliar o processo de compilação;
- Criar diversos APKs para a aplicação, com diferentes recursos, usando o mesmo projeto e os mesmos módulos;
- Reutilizar código e recursos em diversos conjuntos de origem.

6.2. GoogleMaps

O nome do ficheiro de compilação do Android Studio é o *build.gradle*. É usada a sintaxe do Groovy para configurar a compilação com os elementos fornecidos pelo Android Plugin para Gradle (Studio, 2016).

6.2 GOOGLEMAPS

Como a aplicação requer o uso de um mapa é usada a API do GoogleMaps, sendo que aqui é direcionada para o desenvolvimento em ambiente Android. Vai ser usado o serviço das Direções que calcula um percurso entre duas localizações. Uma vez feito um pedido à API para obtenção de um percurso, a consequente resposta será tratada de forma a representar o percurso no mapa entre uma origem e um destino. O mapa também é usado para a colocação de marcadores com as ocorrências nas estradas de Portugal. Assim um utilizador da aplicação fica informado das situações que podem afetar o seu percurso. Sendo a mesmo serviço que na interface Web (ver Sec. 5.3) também não é possível fazer um percurso que não passe por localizações dadas, ou seja um percurso que consiga evitar as ocorrências na estrada.

6.3 INTEGRAÇÃO DA INFORMAÇÃO EXTRAÍDA

A aplicação Android vai consumir a API REST desenvolvida no capítulo anterior (ver Sec. 5.2), para serem feitos os pedidos ao servidor é o usado o cliente REST Retrofit.

O Retrofit é um cliente REST para Android desenvolvido pela empresa Square. Esta biblioteca fornece uma plataforma para autenticação e interação com APIs e para o envio de pedidos com *OkHttp*. O *OkHttp* é uma biblioteca desenvolvida pela Square para enviar e receber pedidos HTTP.

O Retrofit faz o download dos dados em JSON ou XML de uma API Web. Em seguida é feito o parsing para um objeto do tipo POJO (Plain Old Java Object) que é definido para cada recurso na resposta.

Nas próximas listagens é demonstrado o uso do Retrofit para a recolha das ocorrências nas estradas de Portugal através de um pedido via API ao servidor. No Retrofit, um endpoint é definido dentro de uma interface que usa anotações específicas do Retrofit para codificar detalhes sobre os parâmetros e métodos dos pedidos.

Considerando que o construtor da classe *Estrada* é o seguinte *Estrada (String tipo, String estrada, String concelho, String lat, String lon, String data, Integer id, Integer idDistrito)*, esta classe vai ser o recurso obtido na resposta ao pedido enviado. Como exemplo de um endpoint é usada a interface *EstradasService* na Listagem 6.1 para ser feita a recolha de todas as ocorrências nas estradas do servidor. Será feito um pedido à API e será enviada uma

6.3. Integração da informação extraída

resposta do tipo `List<Estrada>`, ou seja uma lista de ocorrências.

```
1 public interface EstradasService {
2     @GET("Estradas")
3     Call<List<Estrada>> listEstradas();
4 }
```

Listing 6.1: Interface EstradasService

Na Listagem 6.2 é descrita a geração do serviço de recolha de ocorrências. A *App* é a classe principal da aplicação (linhas 19-26) onde vão ser gerados os serviços usando o método `createService()` da classe *ServiceGenerator* (linhas 2-16). Para enviar pedidos à API é necessário criar uma instância da classe *Retrofit builder* (linha 5-10). O *BASEURL* (linha 2) vai identificar a localização da API que vai responder ao pedidos feitos com o Retrofit. Com o início da aplicação são criados os serviços (linhas 20-25), sendo então criado um cliente Retrofit para cada um desses serviços.

```
1 //Na classe ServiceGenerator
2 private static final String BASEURL = "http://192.168.1.90:3000/api/";
3 private static OkHttpClient httpClient = new OkHttpClient();
4
5 private static Retrofit.Builder builder =
6     new Retrofit.Builder()
7         .baseUrl(BASEURL)
8         .addConverterFactory(GsonConverterFactory.create(new
9             GsonBuilder()
10                .setDateFormat("yyyy-MM-dd'T'HH:mm:ss")
11                .create())));
12 public static <S> S createService(Class<S> serviceClass, ...) {
13     ...
14     Retrofit retrofit = builder.client(httpClient).build();
15     return retrofit.create(serviceClass);
16 }
17
18 //Na classe App
19 private static EstradasService estradasService;
20 public void onCreate() {
21     super.onCreate();
22     loginService = ServiceGenerator.createService(LoginService.class);
23     estradasService = ServiceGenerator.createService(EstradasService.class);
24     temperaturasService = ServiceGenerator.createService(TemperaturasService.
25         class);
26 }
```

Listing 6.2: Criação do serviço EstradasService

6.4. Aplicação

Na Listagem 6.3, a função *estradasProcessWithRetrofit()* vai proceder ao pedido de recolha das ocorrências via API ao servidor. Na linha 2 é recolhido o serviço *EstradasService* que já se encontra criado na classe *App*. O Retrofit vai fazer o download e o parsing da informação proveniente via API numa thread em background, e em seguida devolver os resultados nos métodos *onResponse* (linha 5-16) ou *onFailure* (linhas 18-20). É executado o pedido à API e no caso de existir uma resposta é recolhida a *List<Estrada>* (linha 6). Para cada ocorrência será feito um marcador que é adicionado ao mapa (linhas 10-14).

```
1 public void estradasProcessWithRetrofit(){
2     final Call<List<Estrada>> mService = App.getEstradasService().
3         listEstradas();
4     mService.enqueue(new Callback<List<Estrada>>() {
5         @Override
6         public void onResponse(Response<List<Estrada>> response, Retrofit
7             retrofit) {
8             List<Estrada> estradas = response.body();
9             for (Estrada res : estradas) {
10                 LatLng lnt = new LatLng(Double.parseDouble(res.getLat()),
11                     Double.parseDouble(res.getLon()));
12                 ...
13                 ocorrenciasMarkers.add(mMap.addMarker(new MarkerOptions()
14                     .title(res.getTipo() + " - " + res.getConcelho())
15                     .position(lnt)
16                     .icon(icon)
17                 ));
18             }
19         }
20         @Override
21         public void onFailure(Throwable t) {
22             Toast.makeText(MapsActivity.this, t.getMessage(), Toast.
23                 LENGTH_LONG).show();
24         }
25     });
26 }
```

Listing 6.3: Exemplo do uso de Retrofit na recolha de ocorrências

6.4 APLICAÇÃO

A aplicação Android desenvolvida permite aos utilizadores autenticados definir um percurso de viagem e serem informados de situações de ocorrências nas estradas, das situações meteorológicas dos locais de origem e de destino. Uma vez definido um percurso, o utiliza-

6.4. Aplicação

dor começa um temporizador de acordo com o tempo esperado da viagem. Este temporizador deve ser desligado pelo utilizador, caso contrário é considerada uma possível situação de emergência.

Na Figura 21 é apresentado o ecrã inicial da aplicação. O utilizador deve introduzir as suas credenciais (email, password) para entrar no sistema.

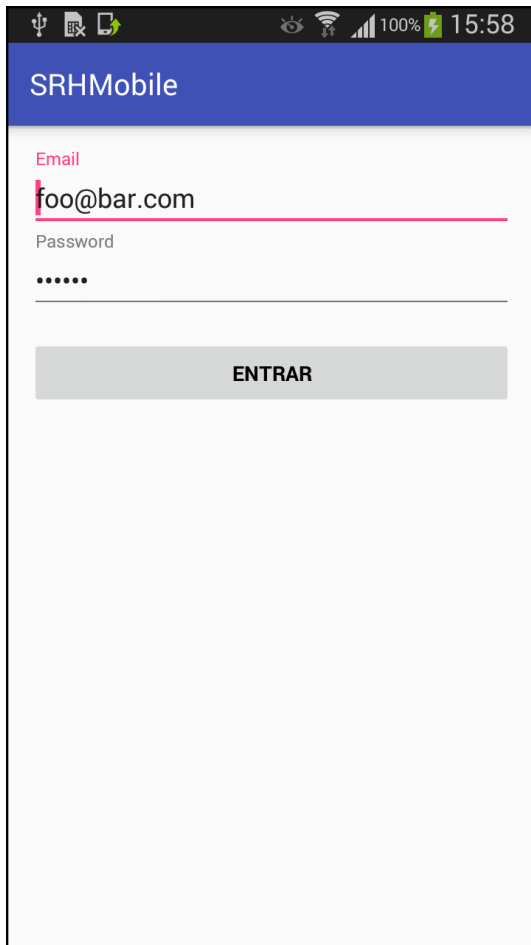


Figura 21: Login do utilizador



Figura 22: Ecrã de boas vindas

Com o utilizador autenticado a aplicação vai avançar para o ecrã da Figura 22, onde é feita referência ao email do utilizador que acabou de entrar na aplicação, neste caso *foo@bar.com*. Neste ecrã tem um botão que vai redirecionar o ecrã para o ecrã do mapa.

6.4. Aplicação

Na Figura 23 é apresentado o ecrã com o mapa. É possível verificar as ocorrências nas estradas de Portugal Continental através dos marcadores no mapa. Na parte superior do ecrã encontram-se os campos para o utilizador preencher com a origem e destino do percurso, o botão de procura para iniciar a procura do percurso.

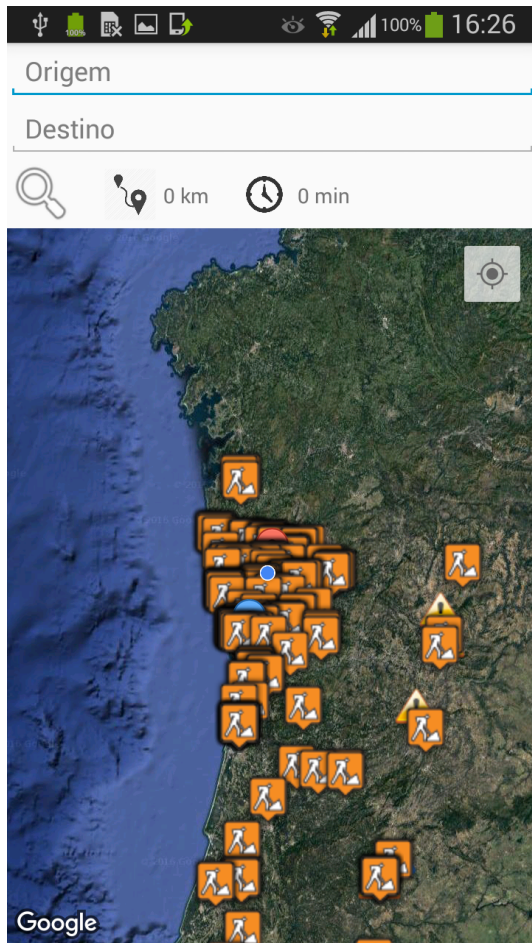


Figura 23: Ecrã com o Mapa

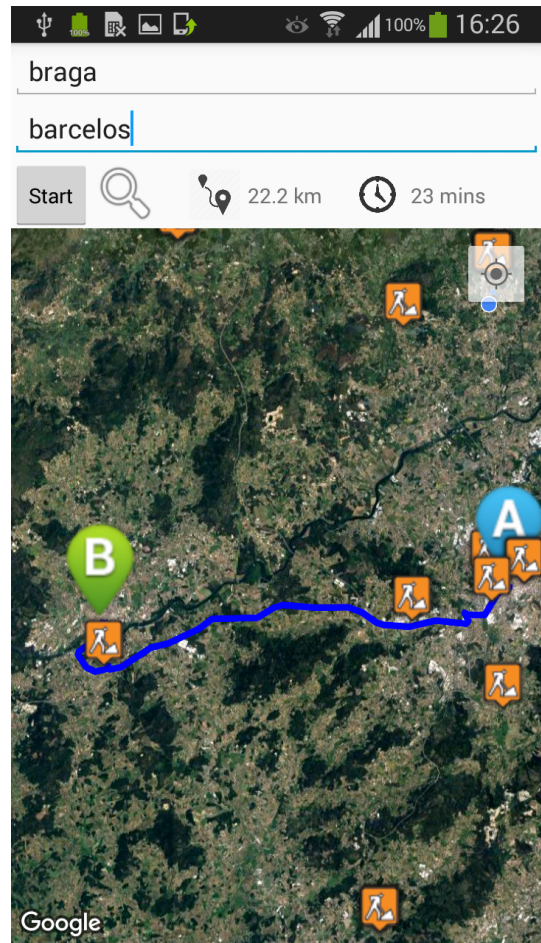


Figura 24: Percurso Braga-Barcelos

A Figura 24 apresenta um exemplo de percurso, neste caso entre Braga (origem A) e Barcelos (destino B). É também apresentada informação relativa à distância e à duração da viagem.

6.4. Aplicação

Estando um percurso definido é possível obter informações relativamente à temperatura da origem e do destino com um clique em cima do marcador A ou B (Figura 25). Com um toque no marcador de Braga (A) é possível ver as suas condições climáticas (uma breve descrição, temperatura e aviso meteorológico).

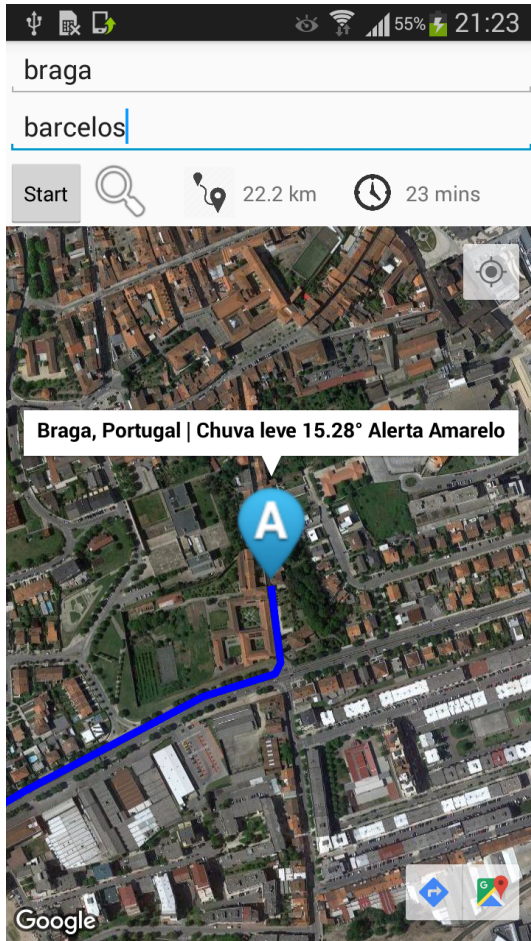


Figura 25: Temperatura de Braga

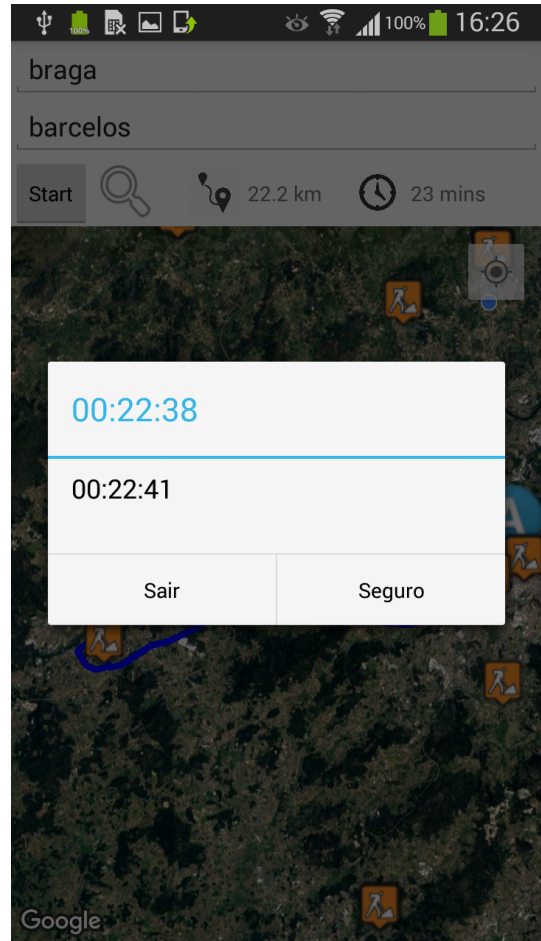


Figura 26: Diálogo com o contador

Para iniciar uma viagem em segurança o utilizador carrega no botão *Start* e é iniciado um dialogo com um contador decrescente com o tempo estimado de viagem (Figura 26). Quando este contador acabar é feita uma espera de 20 segundos para que o utilizador clique no botão *Seguro* para cancelar o contador, caso o utilizador não responda seria guardada a última localização do smartphone e contactado um contacto de emergência (funcionalidade a desenvolver), também é iniciado um novo contador com 5 minutos. Este novo contador terá o mesmo propósito que o anterior, caso o utilizador não responda seria contactada a polícia (funcionalidade a desenvolver).

CONCLUSÃO E TRABALHO FUTURO

No âmbito do desenvolvimento de sistemas de segurança para o dia-a-dia com o objetivo de auxiliar qualquer pessoa a chegar a casa em segurança surgem os serviços de safe-return-home. Com a finalidade de aglomerar informação proveniente de múltiplas fontes diversas na Web para fundamentar a base de um serviço de safe-return-home. Este serviço na forma de aplicação Web e Android vai permitir aumentar a segurança de qualquer pessoa.

Os principais objetivos foram atingidos. Foi construído um serviço de safe-return-home baseado na recolha e análise de informação disponível na Web. A primeira fase do serviço foi o desenvolvimento de um processo de recolha e tratamento da informação recolhida online. Com os dados tratados e compostos numa base de dados foi criado um servidor capaz de responder a pedidos de múltiplas plataformas. No final foram desenvolvidas as aplicações para permitir a utilizadores comuns acesso ao serviço SRH.

Como a recolha de informação é feita em fontes da Web é necessária a procura frequente de novas fontes bem como se torna importante a constante verificação de atualizações. A informação pode mudar de um dia para outro ou deixar de ser atualizada, tendo em conta estes fatores para o serviço ser em tempo real são necessárias fontes de informação que forneçam dados atualizados. Por diversas razões (ver Sec. 4.2) as fontes inicialmente escolhidas para este serviço foram trocadas pelas que servem atualmente o serviço SRH desenvolvido.

Existem ainda alguns possíveis melhoramentos que podem ser explorados e que vão ser descritos em seguida.

7.1 TRABALHO FUTURO

Algumas funcionalidades e melhoramentos que podem ser realizados.

Aumentar o número de fontes O sistema de recolha de informação atual permite o acréscimo de informação proveniente de novas fontes. Com um maior número de fontes seria possível ter mais informação e portanto fornecer melhores condições de segurança aos

7.1. Trabalho Futuro

utilizadores da aplicação.

Melhoramentos na aplicação Android A aplicação Android atual permite demonstrar as funcionalidades propostas para o desenvolvimento desta tese. Melhoramentos no design do layout da aplicação seriam essenciais para que fosse facilitada a utilização da mesma para um público mais idoso ou mais jovem. O desenvolvimento das funcionalidades de contactar contactos de emergência e da polícia.

BIBLIOGRAFIA

- AndroidCentral. Androidcentral. <http://goo.gl/hmdW40>, 2016. Accessed 2016-02-13.
- bSafe. Bsafe. <http://goo.gl/RRBdV1>, 2016. Accessed: 2016-01-13.
- Bugle. Bugle. <http://goo.gl/USc8Q1>, 2016. Accessed: 2016-01-13.
- Circleof6. Circle of 6. <http://goo.gl/UVDMU3>, 2016. Accessed 2016-02-13.
- Estradas.pt. Estradas.pt. <http://www.estradas.pt>, 2016. Accessed August 25, 2016.
- HtmlUnit. Htmlunit. <http://htmlunit.sourceforge.net/>, 2016. Accessed August 25, 2016.
- Jsoup. Jsoup. <http://jsoup.org/>, 2016. Accessed August 25, 2016.
- Kitestring. Kitestring. <http://kitestring.io>, 2016. Accessed January 14, 2016.
- Jae-Won Lee, Ji-Seong Jeong, Mihye Kim, and Kwan-Hee Yoo. Real time safe-return-home service based on big data analytics. 2014.
- Project Lifesaver. Project lifesaver. <http://www.projectlifesaver.org>, 2016. Accessed January 13, 2016.
- LoopBack. Loopback. <http://loopback.io/>, 2016. Accessed August 25, 2016.
- MedicAlert+. Medicalert+. <http://www.medicalert.org>, 2016. Accessed January 13, 2016.
- Gina Petonito, Glenn W. Muschert, Dawn C. Carr, Jennifer M. Kinney, Emily J. Robbins, and J. Scott Brown. Programs to locate missing and critically wandering elders: A critical review and a call for multiphasic evaluation. *The Gerontologist*, 53(1):17–25, feb 2013. ISSN 1758-5341. doi: 10.1093/geront/gns060. URL <http://dx.doi.org/10.1093/geront/gns060>.
- SafeTrek. Safetrek. <http://goo.gl/j9mZJw>, 2016. Accessed: 2016-01-13.
- Selenium. Selenium. <http://www.seleniumhq.org>, 2016. Accessed August 25, 2016.
- StrongLoop. Strongloop. <http://docs.strongloop.com>, 2016. Accessed August 25, 2016.
- Android Studio. Android studio. <http://developer.android.com/studio/intro/index.html>, 2016. Accessed October 12, 2016.

Bibliografia

- Wikipedia. Big data. http://pt.wikipedia.org/wiki/Big_data, 2016a. Accessed January 13, 2016.
- Wikipedia. Web crawler. http://en.wikipedia.org/wiki/Web_crawler, 2016b. Accessed August 25, 2016.
- Wikipedia. Jsoup. <http://en.wikipedia.org/wiki/Jsoup>, 2016c. Accessed August 25, 2016.
- Wikipedia. Node.js. <http://en.wikipedia.org/wiki/Node.js>, 2016d. Accessed August 25, 2016.
- Wikipedia. Web scraping. http://en.wikipedia.org/wiki/Web_scraping, 2016e. Accessed August 25, 2016.
- Wikipedia. Selenium. [http://en.wikipedia.org/wiki/Selenium_\(software\)](http://en.wikipedia.org/wiki/Selenium_(software)), 2016f. Accessed August 25, 2016.
- Wikipedia. Htmlunit. <http://en.wikipedia.org/wiki/HtmlUnit>, 2016g. Accessed August 25, 2016.
- Zoemob. Zoemob. <http://www.zoemob.com>, 2016. Accessed January 13, 2016.