

Universidade do Minho

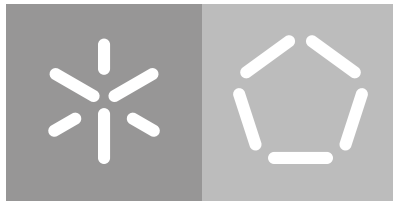
Escola de Engenharia

Departamento de Informática

Pedro Daniel Gomes Fonseca

Smart City Geofences for Vulnerable Road Users

October 2019



Universidade do Minho

Escola de Engenharia

Departamento de Informática

Pedro Daniel Gomes Fonseca

Smart City Geofences for Vulnerable Road Users

Master dissertation

Master Degree in Computer Science

Dissertation supervised by

Cesar Analide

Bruno Fernandes

October 2019

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

Este é um trabalho académico que pode ser utilizado por terceiros desde que respeitadas as regras e boas práticas internacionalmente aceites, no que concerne aos direitos de autor e direitos conexos.

Assim, o presente trabalho pode ser utilizado nos termos previstos na licença abaixo indicada.

Caso o utilizador necessite de permissão para poder fazer um uso do trabalho em condições não previstas no licenciamento indicado, deverá contactar o autor, através do RepositóriUM da Universidade do Minho.



Atribuição-NãoComercial-CompartilhaIgual
CC BY-NC-SA

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

ACKNOWLEDGEMENTS

Em primeiro lugar queria agradecer ao meu orientador Cesar Analide e ao meu coorientador Bruno Fernandes pela disponibilidade demonstrada para me guiar e ajudar durante toda a dissertação, com um agradecimento especial ao Bruno por todo o empenho demonstrado e disponibilidade para me ajudar em todos os passos da construção desta dissertação.

Queria agradecer também aos meus pais, que tudo fizeram para me dar as melhores condições possíveis para a conclusão deste capítulo da minha vida.

Um agradecimento especial também aos meus amigos que sempre estiveram presentes ao longo deste percurso, todas as vivências e todas as memórias criadas e também a todos os meus colegas ao longo dos anos na Associação Académica da Universidade do Minho por todo o conhecimento partilhado, as experiências que partilhamos nas várias atividades realizadas.

Por fim, mas não menos importante, um grande agradecimento à minha namorada que foi um porto seguro para mim, que nunca me deixou perder o foco e que sempre me apoiou em todos os momentos e foi elemento essencial para que conseguisse entregar esta dissertação.

ABSTRACT

Some road users have been targeted in a way to try and create systems that help increase their safety. We are referring to, in this instance, pedestrians, cyclists and even motorcyclists whom we call *Vulnerable Road Users (VRU)* due to the fragility they present when compared to other users that circulate on the same roads, in particular cars. These systems fall within the ambit of the Smart Cities that are increasingly growing all over the world.

The objective of this dissertation is to take advantage of geofencing technology, i.e., to create a virtual perimeter for a real geographical location in which it is desired to detect at any moment the entrance or exit of an user. To achieve such goals a mobile app was developed with the purpose of notifying the user as soon as he enters or exits such geofences. A web-based platform was also conceived and designed for city hall administrators where geofences may be created, updated, deleted and classified.

The main focus went towards the creation of geofences in specific points of interest for *VRUs*, i.e., areas that may be considered dangerous for pedestrians when crossing the road, due to bad visibility for instance, signaling road blocks or construction areas that may cause disruption in traffic or even a road accident.

Keywords: Geofences, Road Safety, Smart Alerts and Notifications, Smart Cities, Vulnerable Road Users.

RESUMO

Certos utilizadores dos meios rodoviários têm sido alvo de estudo de forma a tentar criar sistemas que ajudem a aumentar a segurança dos mesmos. Estamos a falar em concreto de peões, ciclistas e até mesmo motociclistas e estes denominam-se *VRU* sendo que o seu nome advém da fragilidade que apresentam perante os demais utilizadores que circulam nas mesmas estradas, em particular os carros. Estes sistemas inserem-se no âmbito das *Smart Cities*, cidades tecnológicas que cada vez mais estão em crescendo pelo mundo inteiro.

O objetivo desta dissertação é tirar partido das vantagens da tecnologia das *geofences*, isto é, criar um perímetro virtual para uma localização geográfica real na qual se pretende detetar a todo o momento a entrada ou saída de um utilizador. Para atingir estes objetivos desenvolveu-se uma *mobile app* com o propósito de notificar um utilizador, seja este um pedestre, um ciclista ou um condutor, sempre que este entre ou saia de uma dessas *geofences*. Foi também desenvolvida uma plataforma *web* para um administrador por exemplo um gestor de uma Câmara Municipal, onde pode criar e remover *geofences*.

O foco principal desta dissertação estará direcionado para a criação de *geofences* em pontos de interesse específicos para os *VRU*, isto é, áreas que podem ser consideradas perigosas para travessias de peões, devido a má visibilidade, de forma a sinalizar acidentes rodoviários ou ainda obras na via pública que possam causar disrupções no tráfego.

Palavras-Chave: Avisos Inteligentes, *Geofences*, Segurança Rodoviária, *Smart Cities*, *Vulnerable Road Users*.

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

CONTEÚDO

1	INTRODUÇÃO	1
1.1	Motivação	2
1.2	Objetivos	2
1.3	Estrutura do documento	3
2	REVISÃO DO ESTADO DA ARTE	4
2.1	Conceitos	4
2.1.1	<i>Vulnerable Road Users</i>	4
2.1.2	<i>Smart Cities</i>	5
2.1.3	<i>Geofences</i>	6
2.2	Trabalho Relacionado	8
2.2.1	Spotique Service	8
2.2.2	Vehicle-to-Pedestrian Communication Modeling and Collision Avoiding Method in Connected Vehicle Environment	9
2.2.3	<i>MotoWarn</i>	11
2.2.4	Safety Enhancement Service for Vulnerable Users using P2V	12
3	ANÁLISE DE FERRAMENTAS E TECNOLOGIAS	14
3.1	Beacons	14
3.2	TomTom Geofencing API	15
3.3	Google Geofencing API	17
4	DESIGN E IMPLEMENTAÇÃO	19
4.1	O Problema, Os Seus Desafios e a Solução Proposta	19
4.2	Arquitetura do sistema	21
4.2.1	Plataforma Web	22
4.2.2	Mobile Application	23
4.3	Design e Modulação	24
4.3.1	Diagramas de Sequência	24
4.3.2	Mock ups	27
4.4	Implementação	30
4.4.1	Plataforma Web	30
4.4.2	Mobile Application	33
4.5	Interface	36
4.5.1	Plataforma Web	36
4.5.2	Mobile Application	38

5	CASOS DE ESTUDO E RESULTADOS	41
5.1	Caso de Estudo	41
5.2	Resultados	48
6	CONCLUSÕES E TRABALHO FUTURO	51
6.1	Síntese do Trabalho Realizado	51
6.2	Trabalho Futuro	52

LISTA DE FIGURAS

Figura 1	Exemplo da estrutura uma <i>Smart City</i> (Artusse (n.d.))	5
Figura 2	Exemplo de uso de uma <i>geofence</i> Bonack (n.d.)	7
Figura 3	Deteção de <i>Wi-Fi beacons</i> (Spotique)	9
Figura 4	Apresentação dos vários cenários: (a) Peão atravessa a estrada em local sem linha de visão, (b) curva à direita numa interseção, (c) curva à esquerda numa interseção, (d) descrição abstrata e generalizada de um sistema de movimento relativo	10
Figura 5	Stack do modelo de comunicação	11
Figura 6	Gimbal 10 Series Beacons	14
Figura 7	Exemplo da implementação da <i>Application Programming Interface</i> (API) da TomTom (n.d.) numa <i>mobile app</i>	16
Figura 8	Exemplo da implementação da API da Google (n.d.b)	18
Figura 9	Possível local de teste	21
Figura 10	Arquitetura do sistema	22
Figura 11	Diagrama de sequência de adicionar <i>geofences</i>	25
Figura 12	Diagrama de sequência de remover <i>geofences</i>	25
Figura 13	Diagrama de sequência de adicionar sugestões	26
Figura 14	Diagrama de sequência de remover sugestões	27
Figura 15	Esboço do login na plataforma <i>web</i>	27
Figura 16	Esboço da página principal da plataforma <i>web</i>	28
Figura 17	Esboço da <i>mobile app</i>	29
Figura 18	Exemplo da estruturação de uma <i>geofence</i>	30
Figura 19	Exemplo da estruturação da coleção de <i>geofences</i>	31
Figura 20	Exemplo da estruturação da lista de sugestões	33
Figura 21	Método para adicionar <i>geofences</i> à lista que é posteriormente monitorizada	34
Figura 22	Método que determina o <i>Intent</i> de cada <i>geofence</i>	35
Figura 23	Método responsável por determinar o tipo da <i>transição</i> e a <i>geofence</i> transitada	35
Figura 24	Método responsável por construir e retornar o <i>Geofencing Request</i> e definição do que será monitorizado	36

Figura 25	Ecrã de login da plataforma <i>web</i>	37
Figura 26	Ecrã completo da plataforma <i>web</i>	37
Figura 27	Aviso de erro no preenchimento do formulário por falta de certos campos	38
Figura 28	Ecrã principal da <i>mobile app</i>	39
Figura 29	Ecrã principal da <i>mobile app</i> centrado na <i>geofence</i> mais próxima	39
Figura 30	Plataforma <i>Web</i> depois de adicionar a <i>geofence</i>	42
Figura 31	Notificação de entrada recebida pelo <i>VRU</i>	43
Figura 32	Notificação de saída recebida pelo <i>VRU</i>	44
Figura 33	Plataforma <i>Web</i> depois de adicionar as <i>geofences</i>	45
Figura 34	Notificação de entrada recebida pelo <i>VRU</i>	45
Figura 35	Notificação de saída recebida pelo <i>VRU</i>	46
Figura 36	Notificação recebida pelo condutor	47
Figura 37	Notificação recebida pelo condutor	48
Figura 38	Gráfico com os tempos de reposta das notificações recebidas	49

LISTA DE TABELAS

Tabela 1	Comparação entre as várias <i>API</i>	20
----------	---------------------------------------	----

INTRODUÇÃO

O crescimento do número de automóveis em circulação no mundo inteiro em parte devido ao aumento populacional que se tem verificado a nível mundial, é um fator crucial no que diz respeito à segurança rodoviária. Este fator, conjugado com muitos outros como más condições das estradas, fraca visibilidade em lugares de alto congestionamento ou tráfego pedestre demasiado denso, tem levado a que, cada vez mais, existam acidentes rodoviários que por vezes sendo graves, resultam na morte das pessoas envolvidas (Organization (2015)). Daí que o avanço tecnológico que se tem verificado nos últimos anos na área da deteção de objetos e/ou pessoas em tempo real, tem proporcionado a criação de sistemas para tentar diminuir os números da sinistralidade rodoviária (Ota et al. (2017)).

Fazendo uma análise aos números divulgados pela Autoridade Nacional de Segurança Rodoviária, constatamos que no ano de 2018 77% dos acidentes ocorreram em localidades (ASNR (2018a)). Muitos destes devem-se a fatores referidos anteriormente como fraca visibilidade, entre outros, sendo que 50% das vítimas mortais encontram-se dentro das localidades e um número muito significativo corresponde a atropelamentos (ASNR (2018b)). Estes atropelamentos tanto dizem respeito a peões que circulam nas estradas, como a ciclistas, que também se incluem no grupo dos *VRUs*. Muitos dos acidentes registados ao longo do ano que envolvem *VRUs*, devem-se por exemplo ao facto de circularem paralelamente aos restantes veículos, ou ainda porque num certo momento estavam num ponto morto da visão do condutor ou simplesmente por pura distração.

Em suma, fazendo uma análise a estes dados verificamos que existe uma necessidade cada vez maior de criação de sistemas automatizados inseridos no contexto das *Smart Cities*, cidades estas que utilizam sistemas tecnológicos para melhorar a infraestrutura urbana e melhorar a qualidade de vida, de forma a diminuir os números da sinistralidade rodoviária através do uso, por exemplo, de sistemas de avisos inteligentes como as *geofences*.

1.1 MOTIVAÇÃO

Cada vez mais caminhamos para um mundo onde *Smart Cities* serão uma realidade, possuindo sistemas robustos para suprimir certas dificuldades que por vezes ultrapassam o âmbito do ser humano. No caso mais concreto da segurança rodoviária com maior foco para os chamados *VRU*, dado que a sinistralidade rodoviária continua a ter números preocupantes, existe uma necessidade de se criarem sistemas inteligentes que através da análise de padrões de condução, do envio de avisos inteligentes sobre peões ou veículos nas proximidades, ajudem a diminuir estes números. Considerando estes fatores assim como a possibilidade de contribuir para as cidades do amanhã temos patente as maiores motivações para o tema desta dissertação, o desenvolvimento de um sistema que detete a presença de peões e veículos em certos locais considerados de maior risco para os *VRU*, de forma a prevenir cenários como atropelamentos que são um grave problema no mundo inteiro e a contribuição que é feita para a sociedade com sistemas deste tipo que se provam cada vez mais essenciais para melhorar a qualidade de vida no nosso planeta.

1.2 OBJETIVOS

O objetivo principal desta dissertação é o de desenvolver um sistema que permita criar e remover *geofences*, nas quais os utilizadores numa *mobile app* conseguem em tempo real receber notificações da sua entrada e saída das *geofences* assim como informações sobre a categorização das mesmas. Para tal é necessário a criação de uma plataforma *web* na qual se faz toda a gestão da criação, remoção e categorização das *geofences* e uma aplicação para o utilizador no qual este é notificado das suas entradas e saídas das *geofences* ativas mas também onde possa visualizar as mesmas e onde possa ele próprio fazer sugestões para o administrador, que será responsável pela verificação dos dados recebidos. Esta dissertação providencia então as seguintes contribuições:

- Conceber e desenvolver uma arquitetura modular e extensível para a base de um sistema para detetar a presença de *VRU* na área das *geofences*;
- Identificação de locais adequados para a implementação de uma *geofence*;
- Identificação e caracterização de entradas, saídas e presenças de *VRU*;
- Criação de uma plataforma *web* para gestão de *geofences*;
- Criação de uma *mobile app* que faz uso de um sistema de avisos inteligentes para notificar *VRUs* da presença, entrada e saída de *geofences*;
- Criação de avisos inteligentes para serem fornecidos aos *VRU*.

1.3 ESTRUTURA DO DOCUMENTO

Este documento está organizado da seguinte maneira: No Capítulo 2 é feita uma revisão do estado da arte começando por olhar para conceitos e tecnologias que servem de base ao problema em si. Serão apresentados outros sistemas e soluções anteriormente feitas para esta problemática, dando um foco maior para os que já utilizem uma tecnologia semelhante à que será usada para apresentar a solução do tema desta dissertação.

No Capítulo 3 serão apresentadas e analisadas várias tecnologias que poderiam ser utilizadas para esta dissertação como os *Gimbal 10 Series Beacons*, *Google Geofencing API* e *TomTom Geofencing API*, determinando os seus pontos fortes e pontos fracos, fazendo sempre comparações entre elas de forma a ajudar na determinação da mais adequada para esta dissertação.

No Capítulo 4 é feita uma abordagem ao problema em mãos, descrevendo todos os fatores importantes, os principais desafios a ultrapassar, a comparação final entre as várias soluções referidas no capítulo anterior e por fim, a solução proposta. É apresentada a solução implementada e o seu design. Refere-se de forma geral a arquitetura de todo sistema, constroem-se diagramas de sequência para as principais funcionalidades, apresentam-se *mock ups* para ambas as componentes do sistema, descreve-se a metodologia utilizada para a implementação do sistema e por fim apresenta-se a interface final de tanto a plataforma *web* como da *mobile app*.

No Capítulo 5 apresenta-se um caso de estudo que serve de base para a obtenção de resultados que posteriormente serão analisados de forma a concluir a utilidade do sistema.

Por fim, no Capítulo 6 faz-se uma conclusão sobre o trabalho desenvolvido assim como uma introspeção sobre o que foi realizado e os resultados obtidos, fazendo por fim uma ilação do que poderia ser melhorado como trabalho futuro.

REVISÃO DO ESTADO DA ARTE

A revisão do estado da arte é feita de forma a que seja possível entender todo o trabalho já realizado sobre esta problemática por outros investigadores, como abordaram o problema, que ferramentas utilizaram, a que conclusões chegaram e qual o trabalho futuro que pode ser desenvolvido para melhorar ou aperfeiçoar a sua abordagem. Esta revisão deve ser feita tanto para artigos em que tenha sido utilizada a tecnologia das *geofences*, assim como outros métodos que tenham como objetivo aumentar a segurança dos chamados *VRU*, num contexto de *Smart City*. Assim no capítulo 2.1.1 é apresentado o conceito de *VRU*, no capítulo 2.1.2 é definido o conceito de *Smart City* e por fim no capítulo 2.1.3 é explicado o que são *geofences*.

2.1 CONCEITOS

2.1.1 *Vulnerable Road Users*

Como foi referido anteriormente, o foco principal desta dissertação são os *VRU* e como aumentar a sua segurança. Primeiramente é importante perceber quem são estes *VRU*. São considerados como *VRU*, quaisquer utilizadores de rodovias cujo veículo utilizado seja considerado como frágil, em caso de acidente. Mais frágil ainda será aquele que nem veículo utiliza, que neste caso serão os peões. Ora, quando falamos de veículos frágeis estamos-nos a referir a bicicletas e motociclos (Constant and Lagarde (2010)). Em caso de acidente, este tipo de utilizadores são os mais frágeis e mais sujeitos a consequências graves, devido à falta de proteção que apresentam perante os demais utilizadores como carros, autocarros, camiões etc.

Como tal, têm sido feitos estudos e propostas para a criação de sistemas que tentam aumentar o nível de segurança dos *VRU* quando circulam nas estradas. A grande parte deste tipo de sistemas está, neste momento, mais vocacionado para ser utilizado nos carros pois é o veículo onde será mais fácil introduzir tecnologia que muitas vezes ocupa algum espaço ou necessita de certas características que motociclos e bicicletas não possuem. Falamos de sistemas de prevenção de colisão, por exemplo, que até englobam para além dos *VRU*, ou-

tros veículos. Esta tecnologia está a ter grande avanço no mercado automóvel sendo que muitos carros hoje em dia já possuem sistemas que, abaixo de uma certa velocidade fazem o carro travar automaticamente caso este detete um objeto no seu caminho. Existem muitas marcas que já têm esta tecnologia implementada nos seus carros como por exemplo, a *Toyota*, a *Tesla* e muitos outros (Wong (n.d.), Tesla (2018)).

2.1.2 *Smart Cities*

Smart Cities é um conceito em crescendo de importância na sociedade e que tem como objetivo melhorar a qualidade de vida dos seus habitantes. Isto é alcançado através da recolha de dados utilizando por exemplo sensores, o que obriga a um posterior tratamento destes mesmos dados para que se possa fazer a correta gestão dos dispositivos em uso e feitos ajustes para melhorar a eficiência dos sistemas implementados para tirar o maior proveito do conceito de *Smart Cities* (Figura 1).



Figura 1: Exemplo da estrutura uma *Smart City* (Artusse (n.d.))

As *Smart Cities* definem-se pela sua infra-estrutura de comunicação, pelos seus espaços digitais assim como pelas ferramentas públicas que disponibiliza aos seus habitantes. Ou seja, uma série de sistemas que contribuam para a melhoria da qualidade de vida de todos os seus habitantes. Vamos então desta forma apresentar definições do que é uma *Smart City* sabendo sempre que a sua definição e conceptualização ainda está em progresso: (Boulton and Brunn (2011), Hollands (2008))

- “The use of Smart Computing technologies to make the critical infrastructure components and services of a city—which include city administration, education, healthcare, public safety, real estate, transportation, and utilities—more intelligent, interconnected, and efficient.” (Washburn (2010)).

- "A city that monitors and integrates conditions of all of its critical infrastructures, including roads, bridges, tunnels, rails, subways, airports, seaports, communications, water, power, even major buildings, can better optimize its resources, plan its preventive maintenance activities, and monitor security aspects while maximizing services to its citizens."(Hall (2000)).
- "A city well performing in a forward-looking way in economy, people, governance, mobility, environment, and living, built on the smart combination of endowments and activities of self-decisive, independent and aware citizens."(Giffinger (2007)).

Da leitura destas definições e aglomerando numa só definição, concluímos que o propósito de uma *Smart City* será incluir uma variedade de sistemas e tecnologias que monitorizem fatores e índices presentes numa cidade e que comuniquem entre si para otimizar os vários recursos e sectores de uma cidade, de forma a aumentar a qualidade de vida de todos os seus habitantes.

Existe uma grande variedade de abordagens existentes para resolver certos problemas no que diz respeito, por exemplo, à segurança de peões no meio rodoviário. Neste caso em concreto, o tema desta dissertação foca-se em aplicar o conceito de *geofences* de forma a tentar providenciar maior segurança aos chamados *VRU*. Daí que o tema desta dissertação seja *Smart City Geofences for Vulnerable Road Users*.

2.1.3 Geofences

Um conceito muito importante para esta dissertação são as *geofences*. As *geofences* são perímetros virtuais para um local geográfico real, no qual é controlada a entrada, saída e presença de aparelhos, pessoas e outros (Figura 2).

Uma aplicação desta tecnologia pode ser por exemplo numa empresa de venda de produtos, esta pode aumentar as suas vendas se, por exemplo, através do uso *geofences* nas imediações dos seus estabelecimentos, envie uma mensagem com um cupão para os utilizadores usarem. Um caso muito conhecido de utilização de *geofences* ocorreu recentemente e envolveu o *Burger King* e o *McDonald's*. O *Burger King* utilizou a tecnologia das *geofences* integrada na sua nova aplicação, na qual sempre que um utilizador estivesse num raio de sensivelmente 200 metros de um restaurante *McDonald's*, a aplicação enviava uma notificação na qual oferecia uma promoção num certo *hamburger* ficando este pelo preço de um *penny* (Garcia (2018)).

Também no mercado do imobiliário, em concreto nos Estados Unidos da América, está a ser estudada a possibilidade de utilizar *geofences* para o chamado *Open House Marketing*, no qual é enviada uma mensagem a potenciais compradores sempre que estes estejam nas imediações de uma destas casas (Suart (n.d.)).



Figura 2: Exemplo de uso de uma *geofence* Bonack (n.d.)

Outra aplicação pode ser para controlo parental, determina-se um perímetro do qual uma criança não pode sair e caso o faça os pais recebem uma notificação no seu *smartphone* (De Lara (2008)).

Outro contexto de aplicação desta tecnologia passa por utilizar *geofences* como método para a ativação ou desativação de armas de fogo, isto é, um utilizador que possua uma arma em casa para auto-defesa, esta apenas funcionará no perímetro da sua residência sendo desativada sempre que esteja fora do mesmo, impossibilitando-a de ser usado noutra local para outro propósito.

Existem várias formas de implementar uma *geofence*, a Google por exemplo já dá acesso a uma *API* que permite a criação de *geofences* (Google (n.d.a)). Outro método passa pela utilização de vários sensores para criar um perímetro virtual numa localização geográfica real. Através do uso de sensores pode-se controlar o número de entradas, saídas e presenças de pessoas (Fernandes et al. (2018)). Este modo de implementação pode ter muitos usos, especialmente no foro comercial mas também no foro da segurança. Por exemplo, para uma superfície comercial, saber quantas pessoas entraram em determinadas lojas ou visitaram certas partes de um centro comercial pode ser informação muito valiosa para perceber se, por exemplo, uma certa loja tem pouco negócio porque poucas pessoas, em média passam por aquela zona do centro comercial.

2.2 TRABALHO RELACIONADO

Neste capítulo será feita uma exposição de todo o trabalho já realizado sobre esta temática da segurança dos *VRU* e como garantir um aumento da sua segurança quando circulam em rodovias. No capítulo 2.2.2 é introduzido um exemplo da arquitetura de um sistema que usa uma rede *Vehicle-to-Pedestrian (V2P)* com comunicações em tempo real de forma a prevenir acidentes. No capítulo 2.2.3 apresentamos uma solução de uma aplicação que utiliza comunicação unilateral entre o veículo e os demais. No capítulo 2.2.4 é apresentado um sistema de comunicação *Pedestrian-to-Vehicular (P2V)* para a segurança de *VRU*.

2.2.1 *Spotique Service*

Este projeto é feito no contexto de *Geofence and Network Proximity* onde é feito um estudo e posteriormente criação de um sistema que tem como objetivo a substituição da chamada *geo data* por *network proximity*. Esta necessidade advém em parte, de um grande problema que os chamados *Location Based Systems (LBS)* atualmente têm, que é um grande consumo energético. Quando nos referimos a este consumo estamos claro a falar no lado do cliente pois este valor só é elevado devido à reduzida capacidade das baterias no qual este sistemas correm, que neste caso em concreto serão os *smartphones* (Namiot and Snep-Snepe (2013)).

Este serviço tira partido de sinais de *Wi-Fi* que estejam nas imediações de um *Access Point*, por exemplo, e graças aos *probes* que por norma são feitos entre os aparelhos, é capaz de retirar informações como a força do sinal e o endereço *MAC* (Namiot and Snep-Snepe (2013)). Através do uso de serviços como o *Google Cloud Messaging for Android* e do *Apple Push Notification* é possível enviar para todos os utilizadores que se encontrem dentro do perímetro da *geofence* uma mensagem sem que em momento algum seja necessário explicitamente localizar o cliente, permitindo assim manter um certo nível de privacidade. Este serviço pelas suas características permite também que se reduza muito o consumo energético nos aparelhos móveis comparativamente aos métodos tradicionais. Esta forma de implementar *geofences* demonstra uma boa aplicação para um sistema no interior de um edifício ou algo semelhante, não sendo o âmbito pretendido para esta dissertação mas que serve de exemplo para demonstrar outras possíveis aplicações da tecnologia.

Apresentamos por fim a figura 3 retirada de Namiot and Snep-Snepe (2013) que representa o caso de estudo para a aplicar esta tecnologia.



Figura 3: Detecção de Wi-Fi beacons (Spotique)

2.2.2 *Vehicle-to-Pedestrian Communication Modeling and Collision Avoiding Method in Connected Vehicle Environment*

Como referenciado em He et al. (2008) uma possível abordagem a esta problemática passa por desenvolver uma rede *V2P* com comunicações em tempo real. O sistema implementado funciona à base de comunicações indiretas a uma distância curta entre os veículos e os peões, através do uso de um equipamento idealmente colocado à beira da estrada que funciona como nodo central entre as comunicações e os intervenientes. Esta comunicação *V2P* é modelada através da criação de um modelo geométrico estocástico combinado com uma função que efetua estimativas de risco. Este modelo, formula os variados efeitos tendo em consideração as variadas incertezas ou dificuldades que este sistema *V2P* pode encontrar (He et al. (2008)).

Os autores identificaram neste artigo certas dificuldades que maioritariamente são consideradas como globais à temática aqui estudada:

- Latência entre comunicações feitas entre o veículo e o peão;
- Incerteza da precisão dos sistemas de localização para a determinação da posição do veículo e do peão;
- Incerteza no comportamento do peão.

A mais relevante destas incertezas é sem dúvida a latência entre as comunicações feitas entre o veículo e o peão. Dado que situações como a referida anteriormente são *time sensitive*, haver um atraso considerável entre as comunicações pode significar que o sistema não responde a tempo e portanto não é possível evitar-se um acidente podendo resultar na perda de uma ou mais vidas, no caso mais drástico.

A incerteza da precisão dos sistemas utilizados para determinar a posição dos intervenientes é também uma dificuldade que põe em causa o correto funcionamento do sistema. A maior parte dos sistemas que utilizam *Global Positioning System (GPS)* não são capazes de determinar uma localização precisa pois por norma, o *GPS* tem uma imprecisão de até 10 metros, sendo que esta imprecisão pode ainda piorar quando conjugada com outros fatores existentes que podem levar a que a precisão do *GPS* diminua como, condições climáticas ou obstrução do sinal por parte de infraestruturas como, túneis, edifícios, árvores, entre outros (He et al. (2008)).

Por último, a incerteza no comportamento do peão é um fator que não pode ser desconsiderado de maneira alguma. Por vezes é imprevisível que um peão atravesse a estrada sem antes verificar se o pode fazer em segurança e este tipo de comportamentos assim como muitos outros podem afetar os resultados obtidos nos testes efetuados para determinar a viabilidade do sistema.

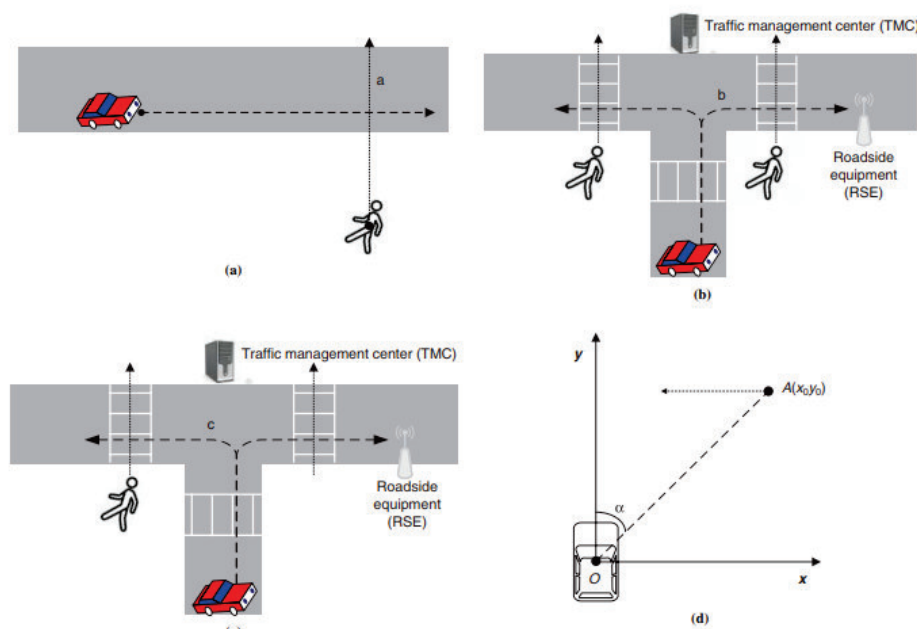


Figura 4: Apresentação dos vários cenários: (a) Peão atravessa a estrada em local sem linha de visão, (b) curva à direita numa interseção, (c) curva à esquerda numa interseção, (d) descrição abstrata e generalizada de um sistema de movimento relativo

Neste caso em concreto optaram por testar a tecnologia num cenário em que não existe campo de visão entre o condutor e o peão. Para melhor compreensão apresenta-se na figura 4, uma figura retirada de He et al. (2008) onde se exemplifica os casos de teste estudados para a realização do artigo. Após a realização de vários testes no *campus* da Universidade de Alberta os autores concluíram que se o veículo se deslocar a velocidades mais elevadas é necessário que a precisão do *GPS* seja maior, assim como uma menor

latência na comunicação *V2P*. Consideraram também que para velocidades superiores a 90 km/h adoptar uma tecnologia como o *Bluetooth* ou o *Wi-Fi* como método responsivo e de controlo não será suficiente, ao invés, o uso de uma *Dedicated Short-Range Communication (DSRC)* com a sua baixa latência e combinado com a tecnologia do *Bluetooth*, seria capaz de cumprir os requisitos para que de facto fosse capaz de providenciar proteção ativa a peões que circulem nas rodovias.

2.2.3 MotoWarn

Anaya et al. (2015) no seu artigo sobre a deteção de *VRU*, abordam a problemática da segurança dos mesmos propondo um modelo em que o centro da informação estará do lado do veículo e a comunicação será unilateral. Neste exemplo em concreto não são incluídos peões mas sim bicicletas e motociclos. Para o efeito, apresentam uma *Vehicular Ad-Hoc Network* onde os nodos serão veículos ou *access points* colocados junto às estradas, onde não existe uma infraestrutura pré-definida e a rede será descentralizada. As comunicações serão feitas através de uma *VANET* onde o veículo ligeiro será o principal emissor e recetor de informação. Este tipo de redes são conhecidas como *Vehicle-to-X (V2X)*.

De forma a manter uma boa qualidade nas comunicações e para que sejam o mais completas possíveis, cada nodo da *VANET* terá de conter uma série de protocolos organizado como demonstra na figura 5, a qual foi retirada de Anaya et al. (2015).

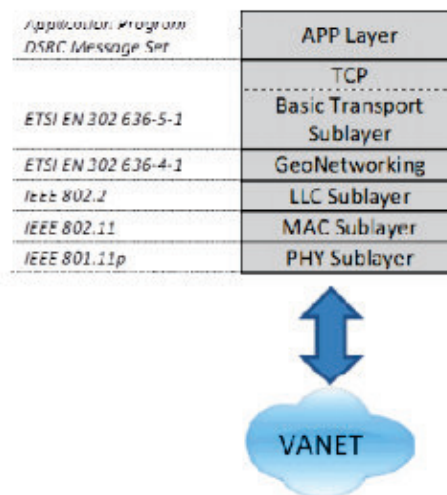


Figura 5: Stack do modelo de comunicação

Como referido anteriormente, motociclos e bicicletas são dois dos grupos pertencentes aos *VRU* devido às limitações que os mesmos possuem, por falta de espaço e pouco peso, para a inclusão de tecnologia com o intuito de aumentar a sua segurança. Neste sistema o

foco principal está no veículo principal que terá instalada a aplicação e que terá em si a responsabilidade de, mediante os dados recebidos ajustar a sua condução para indiretamente aumentar a segurança na circulação dos *VRU*.

Esta aplicação deve ser capaz de detetar *VRUs* que se encontrem nas imediações mas também de retirar outros tipos de informações como a distância a que se encontram por exemplo. O *iBeacon* referido na imagem será um transmissor que através do envio de sinais de *Bluetooth* avisa os aparelhos *V2X* até 50 metros de distância, que estarão equipados tanto nos motociclos como nos veículos, da sua presença.

A grande vantagem associada a esta aplicação está na precisão que tem na determinação do posicionamento das unidades detetadas no perímetro dos aparelhos *V2X*. Como tal é capaz de em tempo real, mostrar a posição relativa de outros veículos assim como a distância a que se encontram, emitindo avisos periódicos ao condutor do veículo ligeiro quanto a alterações da posição dos que o rodeiam apresentando sempre as distâncias entre os mesmos.

A aplicação desenvolvida por Anaya et al. (2015) teve bons resultados nos testes efetuados e provou ser uma boa solução para a diminuição do número de acidentes rodoviários tendo em conta que tira partido de tecnologias que atualmente são bastante acessíveis tanto monetariamente como logisticamente no que diz respeito à inclusão dos mesmos nos futuros modelos tanto bicicletas, motociclos como de carros.

2.2.4 *Safety Enhancement Service for Vulnerable Users using P2V*

Modelos que utilizem comunicações *P2V* têm na sua maioria grande sucesso quando implementados em sistemas cujo intuito seja aumentar a segurança de *VRU*. Cho (2014) desenhou uma arquitetura de um sistema a implementar em locais previamente definidos, sinalizados como de alto congestionamento para tentar prevenir atropelamentos a *VRU*, com especial atenção a crianças e pessoas idosas. Para tal, definiu-se que o sistema estaria dividido em três partes:

- *On-board unit (OBU)*
- *Road-side unit (RSU)*
- *Portable unit (PU)*

Assim, a *OBU* seria instalada nos veículos, a *RSU* estaria localizada junto à estrada nesses lugares identificados como muito congestionados e o *PU* seria utilizado pelos peões. Neste caso o *RSU* irá funcionar como nodo central, isto é, o *OBU* apenas comunica com o *RSU*, assim como o *PU* apenas comunica com o *RSU*. Desta forma sempre que um veículo entrar no perímetro de comunicação de uma *RSU*, esta irá enviar uma mensagem de aviso para

todas as *PU* presentes nas imediações e o inverso também acontecerá, sempre que um *PU* entrar no perímetro de comunicação de uma *RSU*, esta irá enviar uma mensagem de aviso para todas as *OBU*.

Duenkler (2013) também implementou um tecnologia semelhante. Trata-se de uma espécie de sensor (*KO-TAG*) que consegue detetar, identificar, localizar e seguir todos os passos de um peão. Este sensor pode ser embutido em roupas, malas ou mesmo *smartphones*. É idealizado com maior propósito para cenários em que não existe linha de visão entre os *VRU* e os veículos que circulam na sua direção. Para surtir efeitos é necessário que seja implementado um sistema que seja capaz de detetar os chamados *KO-TAG* nas suas imediações e caso identifique uma situação onde possa haver risco de colisão, por exemplo uma distância muito curta entre o veículo e o sensor, então o sistema irá emitir um aviso ao condutor para que este possa tomar providências de forma a evitar um acidente ou no mínimo para que não seja tão grave.

Se a utilização do *GPS* ou de alguma das suas variantes é suficiente para medir a localização dos *OBU* e *PU* ou outro tipo de tecnologia de comunicação para o mesmo efeito. Deve-se ainda considerar sobre a utilização do *Wi-Fi* ou *IEEE 802.11p* para fazer a comunicação entre *OBU-RSU* e *PU-RSU*. Este sistema prova-se então uma viável solução para se implementar em certos locais de forma a aumentar a segurança dos *VRU*.

ANÁLISE DE FERRAMENTAS E TECNOLOGIAS

3.1 BEACONS

A primeira ferramenta estudada foram os *Gimbal 10 Series Beacons*, representados na figura 6, pois estes possuem características importantes. Cada um destes *beacons* emite um sinal *Bluetooth Smart/LE* que facilmente é detetado por outros dispositivos, como *smartphones*, entre outros, sendo necessário que estes dispositivos possuam uma aplicação capaz de detetar os sinais emitidos por estes *beacons*.

De forma a se poder construir uma *geofence*, estes teriam de ser colocados fisicamente no local geográfico desejado, no qual posteriormente seríamos capazes de controlar a presença, entrada ou saída de *VRUs*, automóveis, entre outros. Cada *beacon* é então detetado por qualquer dispositivo capaz de receber e tratar sinais *Bluetooth Smart/LE*, e que possua uma aplicação desenvolvida para o efeito, instalada.



Figura 6: Gimbal 10 Series Beacons

O modo de funcionamento destes *beacons* caracteriza-se por dois momentos distintos, num primeiro ele encontra-se num estado de adormecimento. No segundo momento ele acorda deste adormecimento, emitindo um sinal para as suas redondezas, voltando de

seguida ao estado inicial. Estes beacons caracterizam-se ainda, pelo seu alcance de cerca de 50 metros.

Como visto no capítulo da revisão do estado da arte, um dos maiores problemas que aflige este tipo de tecnologia é a latência, mais ainda, quando o alcance dos *beacons* é reduzido, o que obriga a que a precisão dos mesmos seja muito elevada. Exemplificando, um automóvel em andamento acima de uma certa velocidade, dificilmente detetaria os *beacons* em tempo útil, de forma a poder evitar uma possível colisão ou atropelamento. Não obstante, em teoria, seriam raras as vezes em que estes seriam colocados num local onde os automóveis circulassem a tão alta velocidade.

Estes *beacons* apresentam ainda certas limitações que podem ser um constrangimento para a sua implementação. Primeiramente, requerem que cada um deles seja ativado e configurado na página *Gimbal Manager*, o que em projetos de larga escala não se prova eficiente. Necessitam também que, para que possam ser utilizados, utilizemos a *Gimbal's SDK* para a deteção dos *beacons*, sendo que este *SDK* não é *open source*, o que dificulta ainda mais o processo de implementação e teste de qualquer solução, requerendo ainda que se tenha acesso à internet de forma a validar o id de cada um deles. A conjugação destes e outros fatores não contribuem de forma positiva para que seja implementada uma solução utilizando esta tecnologia.

Por fim, fatores que também pesam na escolha desta *API*, são o custo unitário destes *beacons*, 5 dólares por cada unidade e mais ainda, a necessidade de serem colocados fisicamente nos locais onde se desejasse criar *geofences*, o que comparativamente a outras soluções estudadas é um constrangimento relevante, pois obriga a toda uma logística de fazer a seleção dos locais onde se deseja fazer este tipo de controlo, necessitando que algum recurso humano se tenha de deslocar a estes locais para proceder à colocação de cada um dos *beacons* necessários para criar o perímetro da *geofence*, o que obviamente acresce aos custos referidos anteriormente.

3.2 TOMTOM GEOFENCING API

Numa fase embrionária do projeto estudou-se a possibilidade de se implementar a *API* da *TomTom* para criar, remover e controlar as entradas e saídas das *geofences* selecionadas, de forma a poder criar avisos inteligentes com o intuito de informar os utilizadores sobre potenciais perigos na estrada com maior foco na segurança dos *VRU*.

Esta *API* permite determinar se um certo elemento, neste caso um utilizador, uma pessoa, um ciclista ou mesmo um carro, se encontra dentro, fora, ou nas proximidades destas localizações geográficas.

Face ao intuito inicial desta tese, a *API* não permite, por exemplo, controlar mais do que um objeto num servidor central, ou numa plataforma web, de forma a poder fazer

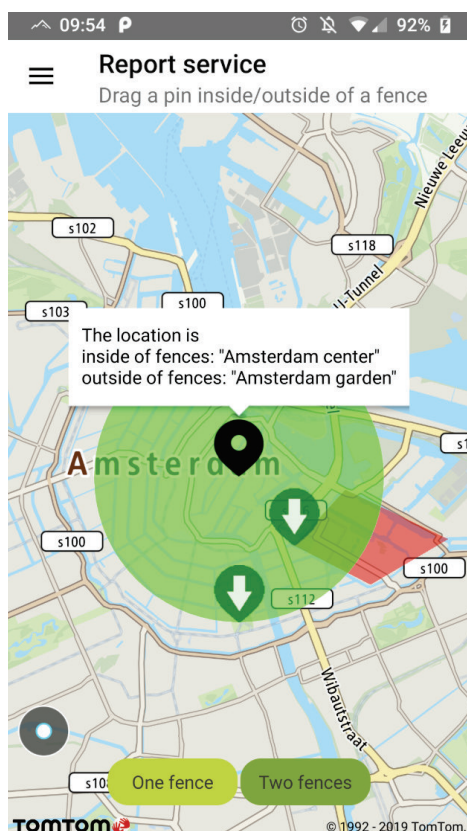


Figura 7: Exemplo da implementação da *API* da TomTom (n.d.) numa *mobile app*

aviso inteligente para aumentar a segurança dos *VRU*. Permite no entanto, definir até mil *geofences* em simultâneo, assim como fazer o controlo de até mil objetos dentro do mesmo projeto.

Perante estes dados, especialmente no que diz respeito ao número de objetos que é possível controlar, num caso real, este número seria bastante limitador, pois seria expectável que em vários momentos de vida do sistema, este número fosse superior a mil e como tal, a aplicação não teria capacidade suficiente para comportar o número de utilizadores em qualquer dado momento o que causaria falhas no serviço.

Neste caso em concreto, seria necessário combinar tanto uma plataforma web para gestão das *geofences*, mas também uma *mobile app* como exemplificado na figura 7, para o utilizador poder visualizar as mesmas e se possível ser notificado das suas transições (entradas e saídas).

A plataforma web poderia ser implementada com a *API* disponibilizada pela TomTom, dando uso a comandos *curl* e ficheiros *JSON* para desenhar as *geofences* e as suas propriedades. Esta *API* é bastante intuitiva e muito customizável, especialmente no aspeto mais importante, o desenho das *geofences*.

A *API* para Android pode ser implementada em *Java* ou *Kotlin* sobre o *Maps SDK for Android* e tendo como ponto fulcral o denominado *Report Service*, responsável por fazer a deteção das entradas e saídas dos utilizadores. Para tal, é necessário criar uma *Query* e de seguida registar um *Result Listener* de forma a receber uma *callback* com o *Report Service Response*. Como passo final, é necessário fazer o *Report Service Request* para receber a lista de *geofences* em que o objeto se encontra dentro de mas também nas que ele se encontra fora de.

Esta forma de implementação apresenta certas limitações, pois necessita que o utilizador esteja dentro da aplicação para determinar se se encontra dentro de alguma *geofence* o que se provaria muito pouco prático e não seria ideal para aumentar a segurança rodoviária dos *VRU* pois este teriam de estar constantemente a verificar se de facto tinham entrado ou saído das *geofences* ativas. Mais ainda, dado que esta tecnologia ainda é bastante recente, tem pouca documentação e algumas limitações comparativamente a outras existentes.

3.3 GOOGLE GEOFENCING API

A última opção estudada foi a *Google Geofencing API*, que à semelhança da *API* da *TomTom* permite a criação e remoção das *geofences*, mas contrariamente à última, esta permite criar avisos mediante a entrada e saída dos utilizadores das *geofences* ativas, sendo isto um ponto fulcral.

Comparativamente à opção dos *beacons*, tanto a *API* da Google como a da *TomTom* estão em desvantagem no que diz respeito à capacidade de criar avisos inteligentes, pois estas não permitem que um utilizador local receba informações em tempo real da localização de outros utilizadores presentes na mesma *geofence*, sendo que nem seriam precisas o suficientes para o fazer. No entanto, esta difere também noutro fator importante, pois permite ainda determinar a transição denominada de *Dwell* que ao invés de emitir avisos aquando da entrada ou saída de um utilizador de uma *geofence* ativa, emite um aviso caso ele permaneça mais do que um certo período de tempo.

Outro fator importante a considerar é que, neste caso em concreto, também seria necessário combinar tanto uma plataforma *web* para toda a gestão das *geofences*, adicionar e remover, como uma *mobile app* para o utilizador. No entanto, ao invés da *TomTom*, a *API* da Google não tem um *SDK* para a criação e armazenamento direto das *geofences*, e como tal é necessário recorrer a ferramentas como o *Google Firebase* para guardar os dados das *geofences* para, posteriormente, serem lidas pela *mobile app* de forma a que sejam então também criadas localmente.

O processo de criação das *geofences* nesta *API* também difere da sua contra-parte, necessitando de informações como:

- Latitude

- Longitude
- Raio

A latitude e longitude definem ponto central da geofence que combinado com o raio define o perímetro da *geofence*. (Figura 8)



Figura 8: Exemplo da implementação da API da Google (n.d.b)

Após a criação das *geofences*, é necessário ficar à escuta por transições de entrada e saída das mesmas, para tal utiliza-se o *Broadcast Receiver* que é acionado quando o *Location Service* envia um *Intent*. Apresenta ainda uma limitação, no que diz respeito ao número de *geofences* ativas num dado momento, pois este nunca poderá ultrapassar os cem, sendo que poderá na verdade ser suficiente mediante o tamanho da zona que se pretende fazer este controlo e gestão.

DESIGN E IMPLEMENTAÇÃO

Neste capítulo irá ser feita uma abordagem ao problema, os seus principais desafios e a solução proposta. É ainda apresentada a arquitetura do sistema e a sua ilustração. Recorrendo à linguagem *Unified Modeling Language (UML)* serão criados diagramas de sequência para as várias atividades existentes, como adicionar e remover *geofences* e adicionar e remover sugestões. Irá ser feita também uma descrição de todo o processo de implementação da plataforma *web* assim como da *mobile app*, serão apresentadas as *mock ups* desenhadas para ambos na ferramenta *MarvelApp* e por fim, será feita uma apresentação da interface final destas duas componentes.

4.1 O PROBLEMA, OS SEUS DESAFIOS E A SOLUÇÃO PROPOSTA

Como já foi indicado anteriormente, para a implementação do sistema será necessário o desenvolvimento de uma plataforma web de forma a criar as *geofences* nos locais identificados como de relevância pelos mais variados motivos. É necessário também, criar uma aplicação funcional que faça os avisos inteligentes mediante a entrada e saída das *geofences* ativas.

As soluções existentes para a problemática da segurança de *VRUs* aquando da sua circulação nas estradas acabam, em grande parte, por ter bons resultados mas são ainda, de certa forma, limitadas e por vezes não extensíveis. Mais ainda, algumas destas soluções necessitam que sejam criados aparelhos a serem integrados em veículos para que se possa fazer o controlo da sua localização de forma correta, não sendo isto prático.

A solução proposta aqui, passa por identificar locais de alto congestionamento ou de má visibilidade, entre outros, propensos a que existam acidentes envolvendo *VRU*, criando uma *geofence* no perímetro desse local. Passa ainda, por permitir sinalizar acidentes na rodovia graças ao serviço de *geofencing* ou ainda avisar da existência de zonas que se encontram em obras para o que o utilizador possa adequar o seu trajeto mediante essa informação.

A tecnologia escolhida para este fim foi a da *Google*. Olhando para a tabela 1, observamos que existem muitas similaridades entre a *API* da *Google* e da *TomTom*, no entanto a facilidade de criação de avisos inteligentes para os utilizadores pesou na escolha, mais ainda com

	Gimbal	TomTom	Google
Criar geofences	✓	✓	✓
Remover geofences	✓	✓	✓
Detetar transições	✓	✓	✓
Detetar outros utilizadores dentro da geofence	✓		
Emitir avisos inteligentes	✓		✓
Sem custo de implementação		✓	✓
Sem colocação física das geofences		✓	✓

Tabela 1: Comparação entre as várias *API*

fatores extra como o facto de esta ter bastante mais documentação de suporte. Comparando com os *beacons* da Gimbal, denotamos que esta última permite a deteção em qualquer momento de utilizadores dentro das *geofences* o que permitiria à *mobile app* fazer avisos mais completos de forma a aumentar a segurança dos *VRU*. No entanto, utilizando estes *beacons* para a criação das *geofences*, é necessário a colocação física dos beacons nos vários locais a controlar o que não se prova muito prático acarretando ainda um custo por cada *beacon* utilizado pois é necessário fazer a sua compra. Mais ainda, numa fase inicial do trabalho de investigação, experimentou-se implementar um simples caso de estudo recorrendo a *beacons*. Contudo, após vários problemas e falta de suporte decidiu-se contactar a empresa responsável pelos mesmos, tendo sido verificado que, na União Europeia, os *beacons* tinham sido bloqueados devido ao Regulamento Geral de Proteção de Dados (RGPD), pelo que a opção dos *beacons* se tornou inviável.

A identificação dos locais a controlar será um passo de grande importância após a conclusão do sistema. Um dos desafios será fazer a triagem dos locais que possam satisfazer os critérios desejados, triagem esta que pode ou não ser demorada mediante a área escolhida para testar o sistema. O ideal será encontrar locais de muito congestionamento tanto de carros como de *VRU*. Outros locais podem também cumprir os requisitos caso sejam considerados perigosos, e possivelmente serem selecionados para local de teste. Dado que se pretende que o administrador seja, por exemplo, uma entidade como uma Câmara Municipal, criar uma *geofence* do tipo obras será perfeitamente determinável de antemão dado que esta tem de ser informada deste tipo de acontecimentos. Quanto à determinação da existência de um acidente na via pública, isto pode-se processar de duas formas, a primeira será pelos canais tradicionais com que este tipo de entidades são informadas, a segunda será através de *feedback* dos utilizadores, que será proporcionado pela *mobile app*. Um local que numa fase inicial pode servir de local de teste do tipo Zona Perigosa, situa-se junto à cantina da Universidade do Minho, como mostra na figura 9. É um local onde todos os dias passam centenas, se não milhares de pessoas por dia que, devido aos carros estacionados,

tem pouca visibilidade e portanto, como caso inicial de teste adequa-se para este tipo de *geofence*.



Figura 9: Possível local de teste

De forma a implementar este sistema, será necessário criar duas aplicações. Uma primeira, plataforma *web*, na qual um administrador tem de ser capaz de adicionar uma nova *geofence*, tem de ser capaz de remover *geofences* ativas e ainda poder-las categorizar e visualizar num mapa. O segundo passo será criar uma aplicação destinada ao *end user* na qual este deverá ser capaz de visualizar a sua posição, assim como as *geofences* ativas. Deve ainda, utilizando os serviços de localização, corretamente identificar a entrada, saída e presença dos utilizadores, com maior ênfase para os *VRU*. Com esta informação, a aplicação deverá emitir avisos em resposta a estes eventos identificando a *geofence* na qual houve uma transição assim como o tipo da mesma, de forma a que o utilizador possa antecipar e ajustar o seu comportamento. Os avisos devem ser o mais completos e corretos possíveis para que a eficácia do sistema tenha impacto direto na segurança de quem circula na estrada.

4.2 ARQUITETURA DO SISTEMA

Como referido anteriormente, a implementação deste sistema dividiu-se em duas partes. Primeiramente, o desenvolvimento da plataforma *web* para a gestão por parte de um administrador de todo o sistema, na qual, este será capaz de adicionar novas *geofences*, remover as existentes, assim como visualizá-las num mapa recorrendo à tecnologia *Google Maps*. Terá ainda adicionalmente, acesso a uma lista de sugestões com coordenadas geográficas enviadas pelos utilizadores da *mobile app* de forma a poder ajudar e complementar todo o processo de identificação de potenciais locais que devem ser monitorizados pelo sistema.

A segunda parte, passa pelo desenvolvimento da referida aplicação para os utilizadores, na qual estes serão capazes de visualizar num mapa, criado novamente recorrendo à *API Google Maps*, centrado na sua posição, todas as *geofences* ativas naquele instante, recebendo notificações de entrada e saída das mesmas de forma a que possam adequar o seu com-

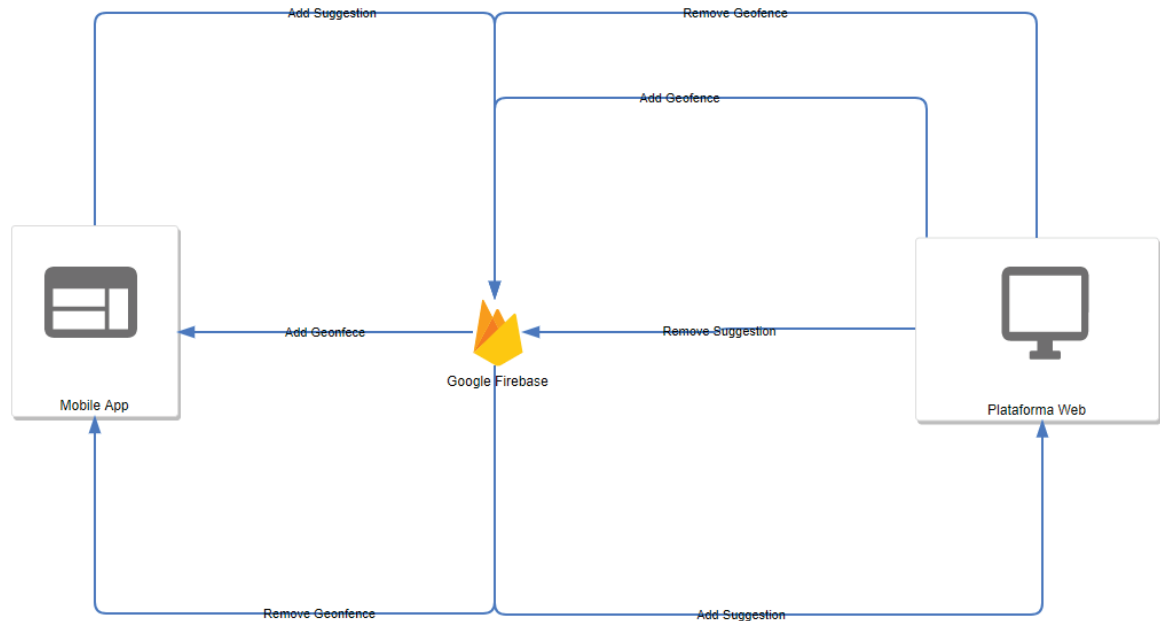


Figura 10: Arquitetura do sistema

portamento enquanto lá permanecerem. Terá ainda a possibilidade de, carregando num botão, visualizar no mapa a localização da *geofence* mais próxima em relação à sua posição atual, assim como sugerir ao administrador do sistema, novas *geofences* a adicionar, caso considere pertinente.

Apresenta-se na figura 10 a arquitetura do sistema.

4.2.1 Plataforma Web

Para o desenvolvimento da plataforma *web* utilizou-se tecnologias como o *JavaScript*, *HTML*, *CSS* e recorreu-se ainda ao *Google Firebase* para fazer armazenamento persistente dos dados necessários para criar as *geofences*.

Definiu-se então que esta plataforma deveria ter as seguintes componentes:

- Formulário no qual o utilizador pode preencher os dados para a criação das *geofences*:
 - Nome;
 - Latitude;
 - Longitude;
 - Raio;
 - Tipo da *geofence*.

- Tabela com a lista das *geofences* ativas:
 - Nome que lhes foi atribuído;
 - Tipo da *geofence*;
 - Botão para as remover.
- Tabela com a lista de sugestões enviadas pelos utilizadores:
 - Coordenadas geográficas enviadas pelo utilizador;
 - Botão para as remover.
- Mapa onde possa visualizar as *geofences* ativas:
 - Botão para entrar em modo *Street View* para melhor visualizar certos locais;
 - Permitir com um clique no mapa determinar as coordenadas geográficas daquele ponto;
 - Permitir com um clique nas *geofences* ativas determinar a qual se refere, detalhando os seus parâmetros que a constituem;
 - Permitir ainda visualizar o estado do trânsito, sendo este um bom indicador para determinar possíveis locais de acidentes.

Para que todo este sistema funcione e seja persistente é necessário guardar os dados submetidos de cada *geofence* para que elas sejam visíveis para todos os utilizadores, a qualquer momento.

4.2.2 Mobile Application

Esta aplicação será disponível para todos os potenciais utilizadores, com foco especial nos *VRU* pois são os mais impactados de forma positiva pelo sistema delineado que tem como objetivo aumentar a segurança rodoviária.

Para todo o desenvolvimento da aplicação escolheu-se utilizar a linguagem *Java*, utilizando o *Android Studio* como ambiente de desenvolvimento.

Em termos de interface esta aplicação será bastante simples, tendo apenas um ecrã no qual o utilizador consegue visualizar num mapa todas as *geofences* ativas, dispondo ainda de um conjunto de funcionalidades como:

- Centrar o mapa na sua posição.
- Pesquisar no mapa pela *geofence* mais próxima em relação à sua posição atual.
- Sugerir a adição de uma nova *geofence* ao sistema, enviando as suas coordenadas geográficas.

- Notificações de entrada e saída das *geofences* ativas.

Para a implementação da aplicação utilizou-se, como referido anteriormente, a *API* da *Google*.

4.3 DESIGN E MODULAÇÃO

De forma a melhorar a compreensão de um qualquer sistema é costume utilizar-se linguagens de modulação como o *UML* para criar modelos e ou diagramas que, de forma visual, descrevem o funcionamento de todo o sistema assim como das várias componentes que o integram. Como tal, nesta secção iremos recorrer a diagramas de sequência para ilustrar todo o processo das funcionalidades mais importantes do sistema. No que diz respeito à parte do design de todo o sistema, nesta secção serão ainda apresentadas as *mock ups* desenhadas para representarem a interface do sistema para que sirvam de base para a criação da interface final funcional tanto da plataforma *web* como da *mobile app*.

4.3.1 Diagramas de Sequência

Para ilustrar de uma forma visual toda a sequência de eventos que compõem o processo de execução das funcionalidades mais importantes do sistema, recorreu-se à linguagem *UML* para criar diagramas de sequência.

Adicionar Geofences

O administrador de sistema após preencher o formulário de adição de novas *geofences* ao sistema, dá início ao processo ilustrado na figura 11. Este formulário terá de ser validado pela plataforma *web*, pelo que se cria então uma alternativa, representada pelo *alt*. Caso o formulário seja válido, a plataforma *web* envia então os dados recebidos para a base de dados presente no *Google Firebase*. Caso não seja, o sistema emite então um aviso de erro para que o administrador possa tentar novamente. Na sequência do formulário ser válido e os dados serem enviados para a base de dados, aquando da inserção é enviado um sinal do tipo *child added* tanto para a plataforma *web* como para a *mobile app*, contendo a informação que foi adicionada que neste caso será uma *geofence*. Similarmente o processo iniciado em ambos será então de adicionar a *geofence* recebida, dando-se então por terminada toda esta sequência.

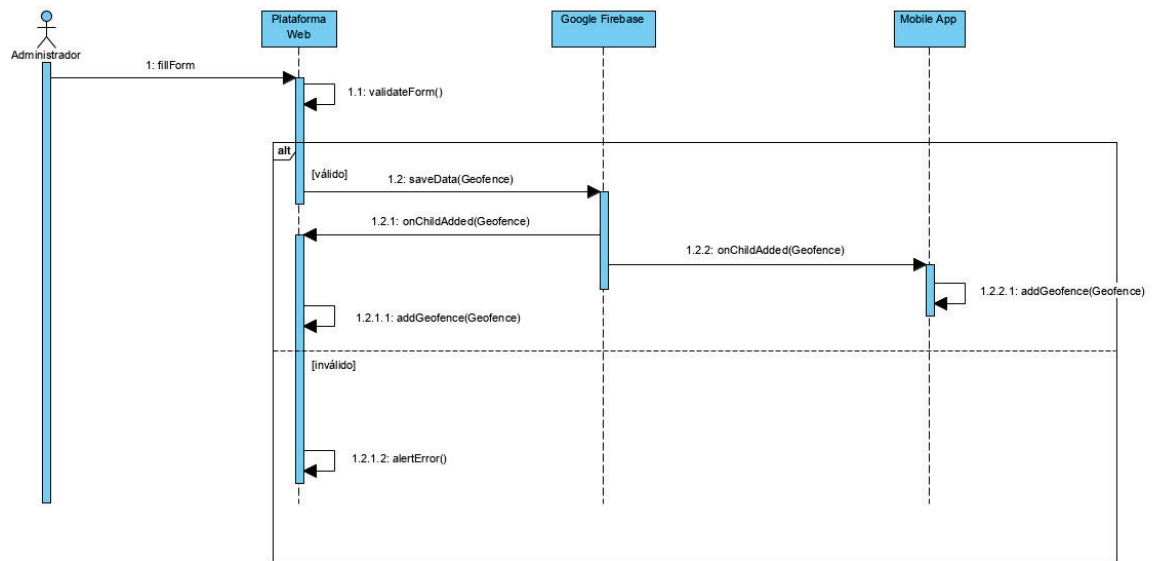


Figura 11: Diagrama de sequência de adicionar *geofences*

Remover Geofences

O ator será novamente o administrador de sistema que, após selecionar a linha da tabela a que corresponde a *geofence* a remover e submeter o formulário, inicia o processo ilustrado na figura 12. É então chamado o método responsável pela remoção da *geofence* da base de dados, sendo também emitido um sinal *child removed* tanto para a plataforma *web* como para a *mobile app* para que possam remover localmente do sistema esta *geofence*.



Figura 12: Diagrama de sequência de remover *geofences*

Adicionar Sugestão

Neste diagrama o nosso ator será o Utilizador, pois este é o único com a possibilidade de fazer uma sugestão, figura (13). O processo inicia-se com o utilizador a submeter o envio da sua sugestão, sendo que esta é então enviada para a base de dados das sugestões presente no *Google Firebase*. À semelhança dos outros casos o *Firebase* envia um sinal de *child added* para a plataforma *web* de forma a que esta a adicione então à tabela de sugestões.



Figura 13: Diagrama de sequência de adicionar sugestões

Remover Sugestão

Por fim a última funcionalidade descrita é a de remover sugestões da tabela presente na plataforma *web*, ilustrada na figura 14. Para este cenário, o ator volta a ser o Administrador, pois apenas ele pode fazer a remoção de sugestões desta tabela. O processo inicia-se quando este seleciona a linha da tabela correspondente à sugestão que ele deseja remover da tabela e envia o formulário. É então de seguida chamado o método responsável por fazer a remoção da sugestão selecionada da base de dados no *Google Firebase*. Neste ponto é enviado um sinal de *child removed* à plataforma *web* para que esta posteriormente possa então removê-la da tabela de sugestões.

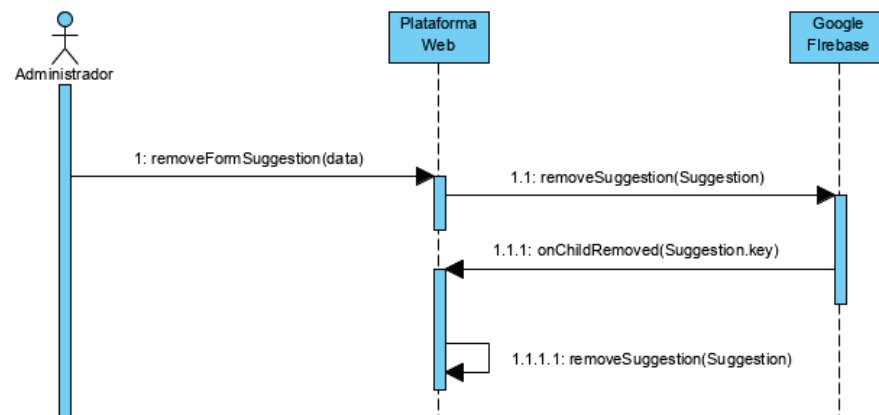


Figura 14: Diagrama de sequência de remover sugestões

4.3.2 Mock ups

Recorrendo à ferramenta *Marvel App*, uma ferramenta especialmente criada para o design de *interfaces*, fez-se esboços para os três ecrãs. O primeiro ecrã, representado pela figura 15, será o ecrã no qual o administrador irá efetuar o *login* com as suas credenciais na plataforma *web*.

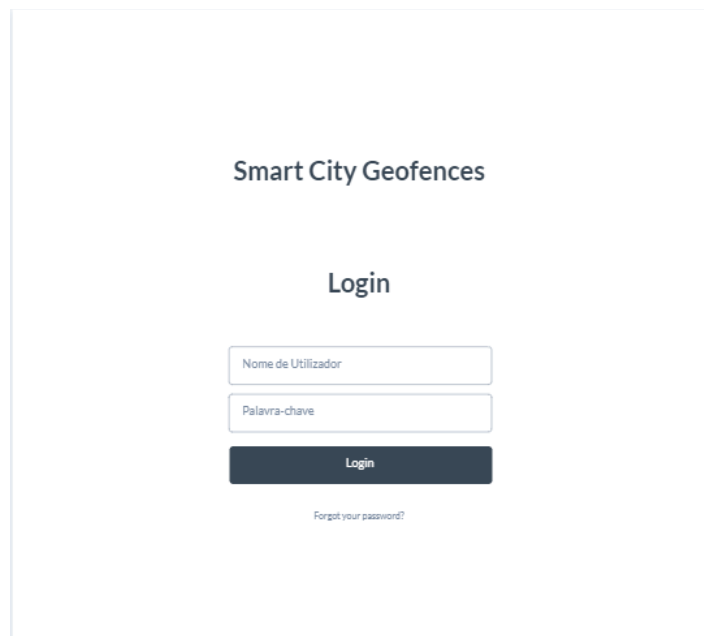


Figura 15: Esboço do login na plataforma *web*

O segundo ecrã diz respeito à página principal da plataforma *web* na qual o administrador fará toda a gestão do sistema. Como podemos observar na figura 16, do lado esquerdo, localiza-se o formulário para adicionar uma nova *geofence*. Depois de adicionada, esta irá ser representada no mapa presente em baixo, por um círculo com a cor correspondente ao seu tipo, sendo ainda adicionada também à lista de *geofences*, representada por uma tabela “Lista de *geofences*” como visto na *mock up*. Temos ainda do lado direito a última componente da plataforma que é a lista de sugestões enviadas pelos utilizadores, representada por uma tabela como se observa na figura.

Adicionar Geovedação

☐ Zona Perigosa
 ☐ Obras
 ☐ Acidente

Lista de Geovedações

Geovedação	Tipo	Remover
Universidade	Perigosa	☐
Variante	Obras	☐

Lista de Sugestões

Latitude	Longitude	Remover
38.7279679	-9.1593448	☐
38.7173315	-9.1598078	☐

Figura 16: Esboço da página principal da plataforma *web*

Por fim, o último ecrã diz respeito à *mobile app*. Nesta componente do sistema apenas iremos ter um único ecrã, apresentado na figura 17, onde como podemos observar apenas se encontra um mapa e um conjunto de botões que representam as funcionalidades presentes na aplicação. As *geofences* ativas irão estar desenhadas no mapa como ilustrado na figura, tendo também três botões, que representam funcionalidades, onde o utilizador pode centrar o mapa na sua posição, pesquisar pela *geofence* mais próxima ou ainda enviar uma sugestão para o administrador do sistema.

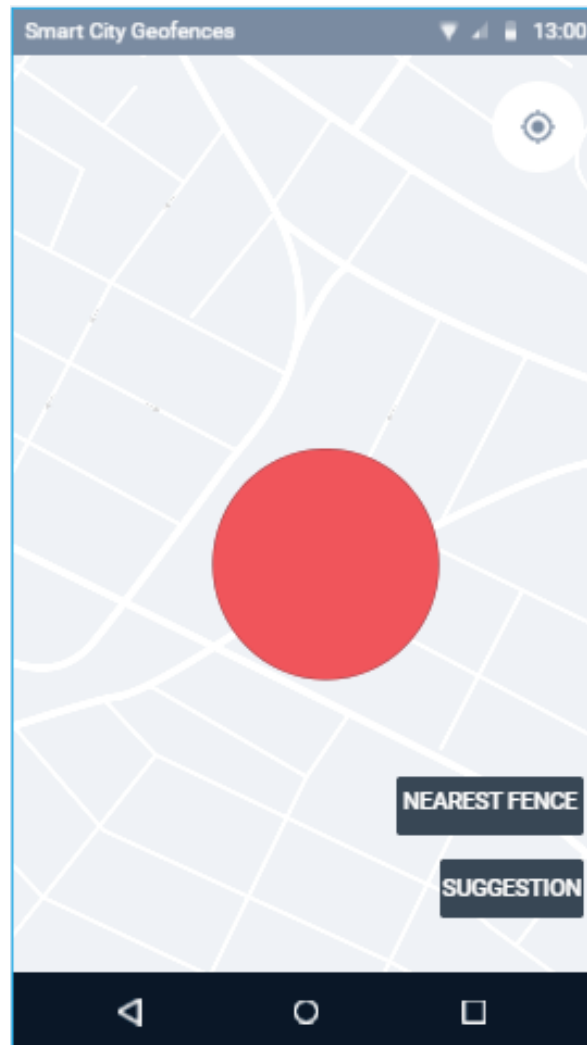


Figura 17: Esboço da *mobile app*

4.4 IMPLEMENTAÇÃO

Vamos agora, nesta secção, fazer uma descrição, de uma forma geral, de todo o processo de implementação das duas componentes que constituem o sistema, das definições e detalhes mais importantes e ainda a explicação de alguns algoritmos essenciais à lógica da aplicação.

4.4.1 Plataforma Web

No que diz respeito à plataforma *web* a sua implementação dividiu-se numa série de objetivos a cumprir para que as três principais funcionalidades fossem executadas:

1. Estabelecer a ligação à base de dados da *Google Firebase*.
2. Incorporar a *API* da *Google Maps* para o desenho das *geofences*.
3. Criar um formulário lógico e funcional que envie os dados necessários para armazenar e posteriormente criar e desenhar as novas *geofences*.
4. Criar as tabelas com a lista de *geofences* e de sugestões, que se mantenham sempre atualizadas.

Os dois primeiros passos são relativamente triviais pois dependem apenas da adaptação de linhas de código já existentes. Para o terceiro ponto, após o administrador submeter o formulário para a criação da *geofence* especificada, este é enviado e guardado na base de dados no *Firebase* numa coleção denominada de *geofences* onde cada uma delas terá o formato da figura 18.

```
Lisboa{  
  lat : 38.7517004203319  
  lng : -9.17090404889086  
  rad : 150  
  type : obras  
}
```

Figura 18: Exemplo da estruturação de uma *geofence*

Esta coleção terá nela contida todas as *geofences* ativas, tendo o formato especificado da figura 19.

Depois de adicionadas à base de dados o passo seguinte é fazer o seu desenho no mapa e adicioná-las à tabela de *geofences* ativas presente na plataforma *web*.

```

geofences{
  Universidade{
    lat : 41.5615903758030
    lng : -8.3974170684814
    rad : 250
    type : perigosa
  }
  Lisboa{
    lat : 38.7517004203319
    lng : -9.17090404889086
    rad : 150
    type : obras
  }
}

```

Figura 19: Exemplo da estruturação da coleção de *geofences*

Para esse efeito, é colocado um *Listener* à escuta de um novo *child added* para que quando uma nova *geofence* seja adicionada, este faça uma chamada do método responsável por fazer o desenho da mesma no mapa, assim como a sua inserção na tabela local das *geofences* que se encontram ativas naquele momento, para que o administrador possa fazer a gestão das mesmas e caso assim o entenda, a remoção de *geofences* que já não façam sentido.

Adicionar geofences

O processo de adicionar novas *geofences* inicia-se com o preenchimento de um formulário. Para que o formulário seja submetido, é necessário que todos os dados requeridos sejam corretamente preenchidos. De forma a facilitar o processo de recolha dos dados por parte do administrador, no mapa presente na página *web* é possível com um clique determinar as coordenadas geográficas de qualquer local, pois o sistema emite essa informação. O administrador dispõe ainda da possibilidade de entrar em modo *Street View* para ter confirmação ocular da localização onde pretende de facto adicionar a *geofence*.

Para garantir que todo este processo funciona de forma correta foram colocados certos controlos aquando do preenchimento do formulário para a criação das *geofences*, como por exemplo:

- O formulário não pode ser enviado com campos em falta;
- O formulário não pode ser enviado caso o nome da *geofence* já exista na lista de *geofences* ativas, pois este serve de chave para a inserção na base de dados.

Caso algum destes casos surja, o sistema emite um aviso de erro auto-explicativo ao administrador, para que possa corrigir e tentar novamente (Figura 27).

Tendo o processo de submissão do formulário para a criação das *geofences* sido concluído, é então feita uma chamada ao método de desenho das mesmas no mapa, para que o administrador tenha uma visualização gráfica de todo o sistema. O método chamado vai precisar de receber um ponto central do tipo *LatLng* que é um par composto pela latitude e longitude desse ponto, precisa ainda de receber o raio do círculo a desenhar assim como a sua cor de preenchimento e da linha que compõem o perímetro, de forma a que seja criado um objeto do tipo *Circle* para que possa então ser desenhado no mapa. Para melhor interpretação dos utilizadores do sistema foi definido que para cada tipo de *geofence* haveria uma cor correspondente, sendo neste caso escolhidas as seguintes:

- Zona Perigosa - Vermelho
- Zona de Acidente - Verde
- Zona de Obras - Azul

Desta forma o utilizador consegue saber de antemão o tipo da *geofence* que está a observar. Esta distinção prova-se ainda mais útil para o utilizador da *mobile app* pois não estará tão familiarizado com este fator como o administrador da plataforma.

Remover geofences

O administrador tem também, como referido anteriormente, acesso à tabela composta pela lista de *geofences* ativas para que melhor possa gerir as que já foram adicionadas e ainda, caso assim o determine, remover *geofences* que já não façam sentido pois, por exemplo o problema já foi resolvido, no caso de serem ou um acidente ou um local de obras.

Para tal, basta seguir esta sequência de passos:

1. Fazer *check* na *checkbox* correspondente à *geofence* a remover;
2. Fazer submissão do formulário para remover a *geofence* seleccionada.

Após a submissão deste formulário é chamado então o método que faz a remoção da *geofence* da lista que se efetua fazendo *match* da *key* que identifica a mesma, correspondendo neste caso em concreto ao nome da própria *geofence*. Por fim, fica a faltar fazer a remoção da mesma do mapa que se efetua dando uso a outro *Listener* que neste caso estará à escuta de um *child removed* e que aquando da deteção desse evento, é responsável por apagar da tabela a linha correspondente à *geofence* removida, assim como pela chamada do método que apaga a *geofence* do mapa presente na plataforma *web*, necessitando para tal da *key* que a identifica para que possa percorrer a lista de *geofences* criadas localmente composta pelos objetos do tipo *Circle*, para que seja então apagada do mapa.

Sugestões

O administrador dispõe ainda de outra ferramenta, a lista de sugestões enviadas pelos utilizadores da *mobile app*, que o pode ajudar na identificação de novas *geofences* a adicionar ao sistema. Esta lista é apenas e somente alimentada pelos utilizadores da *app* e como tal requer da parte do administrador uma capacidade de filtragem dos dados que poderão ser reais e verdadeiros daqueles que podem ser incorretos ou falsos. Esta lista é também, à semelhança da lista de *geofences*, guardada numa coleção no *Google Firebase*, sendo que para que esta seja então enviada para a plataforma do administrador, recorre-se novamente a um *Listener* que fica à escuta de eventos do tipo *child added* para que possa de seguida adicionar a sugestão à tabela que compõem a lista de sugestões, que terá o formato da figura 20.

```
Suggestions{
  -KxCf1XabxF6vJgh4U{
    lat : 38.1232112332112
    lng : -9.4561236543212
  }
  KxCf3XdbxF6vEgh4Ue{
    lat : 43.1236112532112
    lng : -8.4561237543212
  }
}
```

Figura 20: Exemplo da estruturação da lista de sugestões

O administrador quando assim o determinar, pode também fazer a remoção de sugestões da sua tabela utilizando o mesmo procedimento da remoção das *geofences*, para que possa melhor controlar a quantidade de informação que está presente na tabela e se é pertinente ou não para a sua função.

4.4.2 Mobile Application

Para a implementação da *mobile app* dividiu-se nas seguintes etapas os objetivos pretendidos para alcançar o nível de funcionamento desejado:

1. Estabelecer a ligação à base de dados da *Google Firebase*;
2. Incorporar a *API* da *Google Maps* para o desenho das *geofences*;

3. Adicionar as *geofences* presentes na base de dados à lista que será monitorizada pela *app* e fazer o seu desenho no mapa;
4. Emitir avisos de transições de entrada e saída destas *geofences*.

Para aceder à base de dados presente no *Google Firebase*, ao iniciar a aplicação criámos uma série de *Listeners* para detetar eventos de adição e remoção de *geofences*. As *geofences* ativas são então tratadas pela classe *Geofence*, que definimos para encapsular os dados recebidos dado que a própria classe que a *API* dispõe não é informativa o suficiente para os requisitos definidos.

```
private void populateGeofenceList() {
    for (Map.Entry<String, com.google.android.gms.location.sample.geofencing.Geofence> entry : Constants.FLAGGED_LOCATIONS.entrySet()) {
        double lat = parseDouble(entry.getValue().getLat());
        double lng = parseDouble(entry.getValue().getLng());
        int rad = parseInt(entry.getValue().getRad());

        mGeofenceList.add(new Geofence.Builder()
            // Set the request ID of the geofence. This is a string to identify this
            // geofence.
            .setRequestId(entry.getKey() + " - Zona " + entry.getValue().getType())

            // Set the circular region of this geofence.
            .setCircularRegion(
                lat,
                lng,
                rad
            )

            // Set the expiration duration of the geofence. This geofence gets automatically
            // removed after this period of time.
            .setExpirationDuration(Constants.GEOFENCE_EXPIRATION_IN_MILLISECONDS)

            // Set the transition types of interest. Alerts are only generated for these
            // transition. We track entry and exit transitions in this sample.
            .setTransitionTypes(Geofence.GEOFENCE_TRANSITION_ENTER |
                Geofence.GEOFENCE_TRANSITION_EXIT)

            // Create the geofence.
            .build());
    }
}
```

Figura 21: Método para adicionar *geofences* à lista que é posteriormente monitorizada

Como podemos ver na figura 21 é necessário fazer a conversão dos dados recebidos da base de dados para os tipos corretos de forma a que possamos então construir a *Geofence* que iremos adicionar à lista. Depois disto define-se *RequestId* que será a chave que identifica a *geofence*. De seguida, necessitamos de definir a zona circular que neste caso necessita do ponto central da *geofence* assim como do seu raio. O parâmetro talvez de maior importância será a definição dos tipos de transições a controlar, que para este caso em concreto, serão a entrada e saída das *geofences*, *GEOFENCE TRANSITION ENTER* e *GEOFENCE TRANSITION EXIT* respetivamente.

Precisamos também de definir os *Intents* para que o *Location Services* quando detete que um utilizador entrou ou saiu de uma *geofence* envie este sinal. Para detetar este sinal definimos um *Geofence Broadcast Receiver* que a partir da deteção deste *Intent* consegue obter o evento de *geofencing*, determinando o tipo da transição assim como a *geofence* que

foi transitada. É ainda responsável por emitir a notificação ao utilizador detalhando qual a *geofence* transitada, o seu tipo e também o tipo de transição que ocorreu, fazendo ainda *log* dos detalhes desta transição.

```
private PendingIntent getGeofencePendingIntent() {
    // Reuse the PendingIntent if we already have it.
    if (mGeofencePendingIntent != null) {
        return mGeofencePendingIntent;
    }
    Intent intent = new Intent( packageContext: this, GeofenceBroadcastReceiver.class);
    // We use FLAG_UPDATE_CURRENT so that we get the same pending intent back when calling
    // addGeofences() and removeGeofences().
    mGeofencePendingIntent = PendingIntent.getBroadcast( context: this, requestCode: 0, intent, PendingIntent.FLAG_UPDATE_CURRENT);
    return mGeofencePendingIntent;
}
```

Figura 22: Método que determina o *Intent* de cada *geofence*

No método da figura 22 determinamos o *Pending Request* para ser enviado com o pedido para adicionar ou remover *geofences*. Desta forma, sempre que o *Location Services* detetar uma transição de alguma das *geofences* presentes na lista, este irá emitir o *Intent* aqui definido.

O método da figura 23 é responsável por determinar os detalhes da transição de cada *geofence*, isto é, determina o tipo da transição e o *id* da *geofence* transitada e devolve uma *string* com estes dados para ser utilizada no envio da notificação ao utilizador.

```
private String getGeofenceTransitionDetails(
    int geofenceTransition,
    List<Geofence> triggeringGeofences) {

    String geofenceTransitionString = getTransitionString(geofenceTransition);

    // Get the Ids of each geofence that was triggered.
    ArrayList<String> triggeringGeofencesIdsList = new ArrayList<>();
    for (Geofence geofence : triggeringGeofences) {
        triggeringGeofencesIdsList.add(geofence.getRequestId());
    }
    String triggeringGeofencesIdsString = TextUtils.join( delimiter: " ", triggeringGeofencesIdsList);

    return geofenceTransitionString + ": " + triggeringGeofencesIdsString;
}
```

Figura 23: Método responsável por determinar o tipo da *transição* e a *geofence* transitada

No método da figura 24 construímos e retornamos um *Geofencing Request*. Definimos também que caso o dispositivo já se encontre dentro de uma *geofence* no momento da sua criação, deve ser enviada uma notificação na mesma (*INITIAL TRIGGER ENTER*).

É aqui também que definimos a lista de *geofences* que irá ser monitorizada para a deteção de transições. O utilizador pode ainda, caso navegue no mapa, recentrar o mesmo na sua posição atual carregando no botão da Minha Localização. Pode ainda pesquisar pela

```

private GeofencingRequest getGeofencingRequest() {
    GeofencingRequest.Builder builder = new GeofencingRequest.Builder();

    // The INITIAL_TRIGGER_ENTER flag indicates that geofencing service should trigger a
    // GEOFENCE_TRANSITION_ENTER notification when the geofence is added and if the device
    // is already inside that geofence.
    builder.setInitialTrigger(GeofencingRequest.INITIAL_TRIGGER_ENTER);

    // Add the geofences to be monitored by geofencing service.
    builder.addGeofences(mGeofenceList);

    // Return a GeofencingRequest.
    return builder.build();
}

```

Figura 24: Método responsável por construir e retornar o *Geofencing Request* e definição do que será monitorizado

geofence mais próxima, que será calculada com base na posição atual do utilizador e o ponto central de cada *geofence* ativa.

Dispõe ainda da possibilidade de dar a sua contribuição para acrescentar valor ao sistema, podendo enviar sugestões de locais que este identifique que necessitem de ser monitorizados pela criação de uma nova *geofence*, sendo que esta informação enviada será apenas a sua posição atual.

4.5 INTERFACE

4.5.1 Plataforma Web

Com base nos *mock ups* apresentados anteriormente, procedeu-se então ao desenho final da *interface* onde seriam incluídos todos os componentes e funcionalidades especificados anteriormente nos demais capítulos desta dissertação.

O primeiro ecrã apresentado na figura 25 diz respeito ao ecrã inicial onde o administrador irá efetuar o *login* com as suas credenciais.



Figura 25: Ecrã de login da plataforma *web*

Após efetuar o *login* ele será redirecionado para a página principal onde poderá efetuar todas as operações de gestão ao sistema em si (Figura 26).



Figura 26: Ecrã completo da plataforma *web*

Primeiramente temos o formulário para adicionar novas *geofences* ao sistema. Como requerido pelo sistema, o administrador tem de preencher os campos todos obrigatoriamente. Caso não o faça o sistema irá emitir um aviso como ilustrado pela figura 27. Não pode também utilizar um nome que já exista na lista de *geofences*, pois o sistema irá emitir um aviso de erro.



Figura 27: Aviso de erro no preenchimento do formulário por falta de certos campos

Caso o preenchimento esteja de acordo com os parâmetros requeridos então, após a submissão do formulário, a *geofence* será adicionada à tabela e desenhada no mapa. Tem ainda a possibilidade de, posteriormente, selecionando a *checkbox* de uma *geofence*, a remover do sistema para deixar de ser monitorizada. Por fim, temos ainda a tabela de sugestões enviadas pelos utilizadores da *mobile app*, na qual o administrador pode recolher uma grande variedade de dados importantes que podem de certa forma ajudar a todo o trabalho de seleção dos locais apropriados a serem controlados pela aplicação. À medida que for validando as sugestões recebidas, o administrador pode remover essas sugestões selecionando a *checkbox* correspondente à sugestão que pretende remover e de seguida carregando no botão de Remover Sugestão. Esta tabela é muito útil pois permite uma interação entre os utilizadores da aplicação e o administrador de sistema, que pode em caso de ser informação válida, facilitar muito o trabalho do administrador.

4.5.2 Mobile Application

Uma das maiores preocupações com o planeamento da *mobile app* foi garantir que a *interface* seria intuitiva e simples de utilizar. Como tal, ficou delineado que a aplicação só teria um ecrã no qual o utilizador conseguiria ver imediatamente o mapa e as *geofences* ativas.

Com base na *mock up* criada para servir de base para o desenho da *interface* final, criou-se então o ecrã como representado na figura 28. Como requerido anteriormente, temos três botões, no canto superior direito o botão para centrar o mapa na localização atual do utilizador, um botão para pesquisar e centrar na *geofence* mais próxima e por fim um botão para poder sugerir novas *geofences*.



Figura 28: Ecrã principal da *mobile app*

Neste caso o utilizador ao carregar no botão do *Nearest Fence* será transportado para o local que representa o centro da *geofence* mais próxima. (Figura 29)

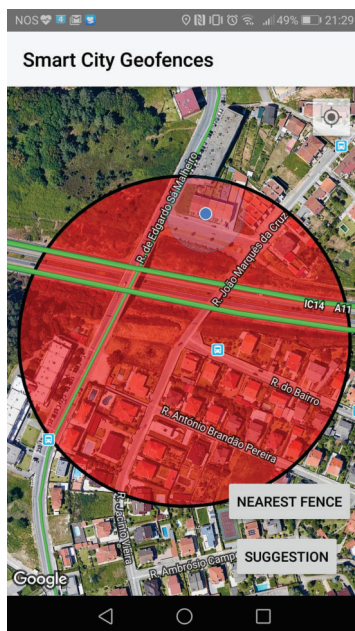


Figura 29: Ecrã principal da *mobile app* centrado na *geofence* mais próxima

Ao clicar no botão de *Suggestion* será enviada a sua localização geográfica para a lista de sugestões presente na plataforma *web*.

Adicionalmente acrescentou-se como se pode ver na figura, a *Traffic API* da *Google* para poder assitir os utilizadores a calcular melhor as suas rotas, no caso de estarem dentro de um veículo, por exemplo.

CASOS DE ESTUDO E RESULTADOS

Nesta secção irá ser feito e descrito um caso de estudo ilustrativo que testa a capacidade de resposta do sistema aos problemas a que se propõem, sendo feita uma análise aos resultados obtidos.

Com isto em mente considerou-se necessário ter três tipos de atores neste caso de estudo. Primeiramente um administrador, responsável por fazer a gestão da adição, remoção e categorização das *geofences*, um *VRU* que irá navegar pela cidade passando nas *geofences* previamente adicionadas e fazendo sugestões de novas *geofences* a ser adicionadas. Consideramos ainda importante ter também um outro ator, que neste caso será um condutor, que fará uma travessia pela cidade acompanhado de um passageiro responsável por lhe transmitir a informação recebida na aplicação.

5.1 CASO DE ESTUDO

O caso de estudo iniciou-se com uma análise e selecção de locais considerados ideais a serem controlados por serem zonas perigosas para os *VRUs*. Tendo completado este passo, o administrador tratou de fazer a adição de *geofences* a esses pontos de interesse. O primeiro local identificado foi em Ferreiros, num cruzamento com pouca visibilidade onde ocorrem muitas vezes acidentes e onde não existem passadeiras para os *VRUs* atravessarem a estrada em segurança. Como tal, o administrador de sistema adicionou nesse ponto de interesse uma *geofence* (Figura 30).

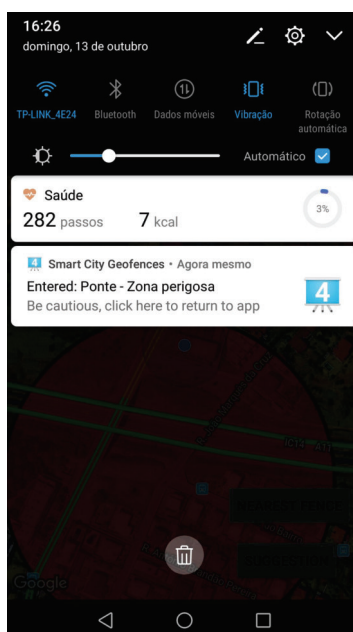


Figura 31: Notificação de entrada recebida pelo *VRU*

Prosseguindo a sua caminhada em direção ao local de saída da *geofence*, recebeu nova notificação, neste caso da sua saída como se pode ver na figura 32. Novamente, denotou-se a existência de um atraso de cerca de três a quatro segundos no envio da notificação. Dado que o *VRU* circula a pé este atraso é aceitável. Na eventualidade do local ser propício à passagem de veículos a circular a alta velocidade, o raio da *geofence* terá de ser ajustado para que tenha uma margem de segurança de forma a garantir que a notificação é enviada a tempo.

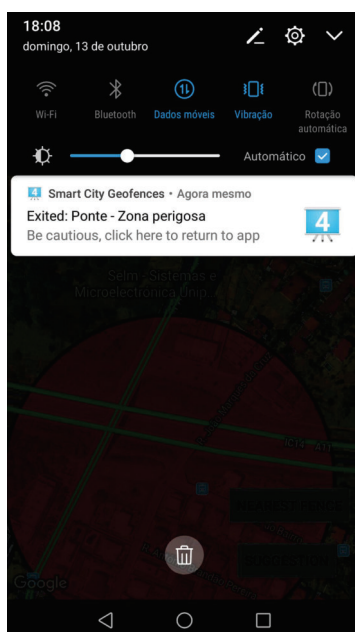


Figura 32: Notificação de saída recebida pelo *VRU*

Outro local assinalado, encontra-se junto à Universidade, onde se indentificou um grande tráfego tanto de veículos como de *VRUs*. Como tal considerou-se este local propício à ocorrência de acidentes. Adicionou-se então uma *geofence* nesse ponto de interesse, representada na figura 33 pelo círculo do meio a vermelho, com um raio de cerca de duzentos e cinquenta metros para dar margem ao sistema para o envio atempadamente das notificações. Para testar a responsividade do sistema enviou-se novamente um *VRU* para o local fazendo uma travessia de entrada e saída desta *geofence*.

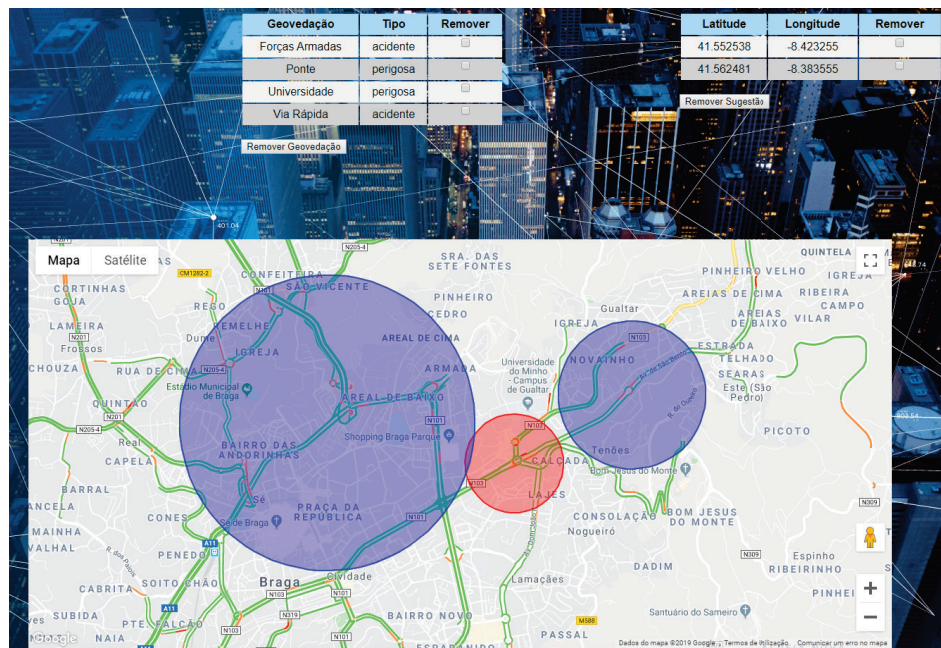


Figura 33: Plataforma Web depois de adicionar as geofences

Neste caso, verificou-se que a notificação de entrada na *geofence* demorou entre quatro e cinco segundos a ser enviada. Novamente, a responsividade do sistema foi dentro dos parâmetros esperados pelo que os resultados continuam a ser positivos.

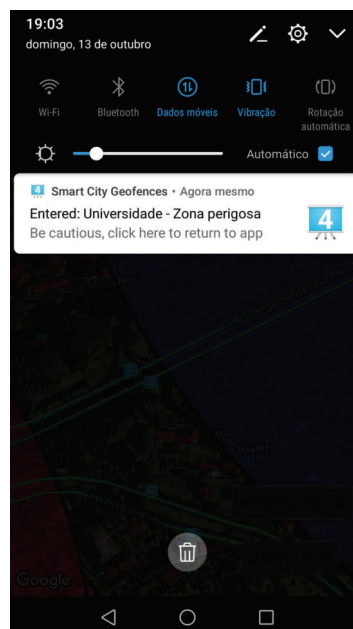


Figura 34: Notificação de entrada recebida pelo VRU

Iniciando o caso da transição de saída, o *VRU* continuou a sua caminhada até ao fim da *geofence*. A notificação de saída demorou neste caso entre três e quatro segundos a ser enviada como se pode ver na figura 35. Temos novamente à semelhança dos exemplos anteriores, o tipo da transição, o nome e tipo da *geofence*.

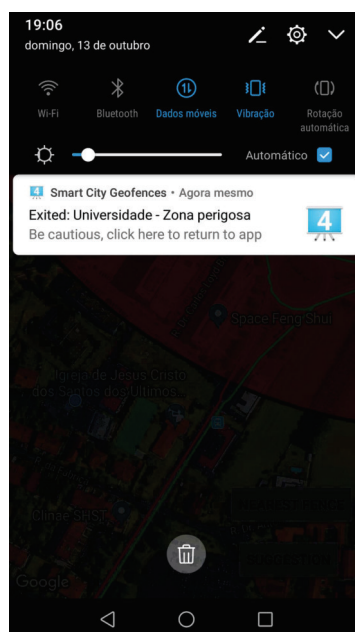


Figura 35: Notificação de saída recebida pelo *VRU*

Tratando-se de um dia chuvoso enquanto o condutor fazia uma viagem pela cidade, deparou-se com um acidente junto ao Nó das Enfiás que provocou muita fila e congestionamento no tráfego rodoviário. Como tal, o nosso condutor enviou na *mobile app* uma sugestão para o administrador. O administrador recebendo a sugestão e olhando para o estado do trânsito segundo a *API* da *Google* tratou de adicionar uma *geofence* no local com um raio de mil e quinhentos metros de forma a avisar com antecedência condutores que se dirigissem para esse local. Dado que a *geofence* é do tipo Acidente, o círculo fica então desenhado a azul. Após a adição da *geofence*, representada na figura 33 pelo círculo da esquerda a azul, o condutor recebeu no seu telemóvel a notificação de que tinha entrado numa *geofence* como mostrado na figura 36. Como neste caso em concreto o veículo já se encontrava dentro da *geofence* e circulava a baixa velocidade, pois se encontrava numa fila, a notificação recebida foi quase instantânea pois foi acionado o *INITIAL TRIGGER*. Como tal, comparativamente às situações anteriormente expostas não se notou uma diferença de tempos.

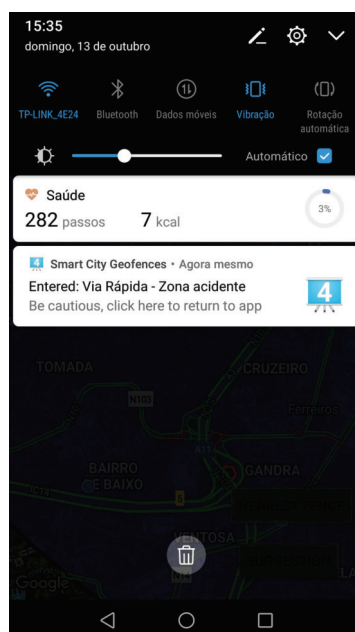


Figura 36: Notificação recebida pelo condutor

À saída desta *geofence* verificou-se que o envio da notificação também não teve grande atraso, mantendo-se na ordem dos três, quatro segundos, pois as condições de viagem mantiveram-se semelhantes às que ocorreram no período aquando da sua entrada na *geofence*.

Continuando o seu percurso, o condutor voltou-se a deparar com um novo acidente agora na estrada que vai para a Póvoa de Lanhoso, na rotunda das Forças Armadas. Enviou então nova sugestão para o administrador. O administrador adicionou a nova *geofence* no local enviado para a lista de sugestões, como representado na figura 33 pelo círculo da direita a azul. Para esta *geofence* após uma análise ao local, determinou-se ser necessário um raio de setecentos e cinquenta metros que cobria as saídas mais próximas onde um condutor poderia ajustar a sua rota para não passar pela zona do acidente. Depois desta ter sido adicionada, o condutor recebeu novamente uma notificação como se segue na figura 37. No entanto, para este caso a notificação de entrada atrasou-se bastante, mais demorando quase um minuto a ser enviada. Isto pode acontecer devido a fatores como por exemplo, fraca cobertura de rede que causa imprecisão no sinal de *GPS*.

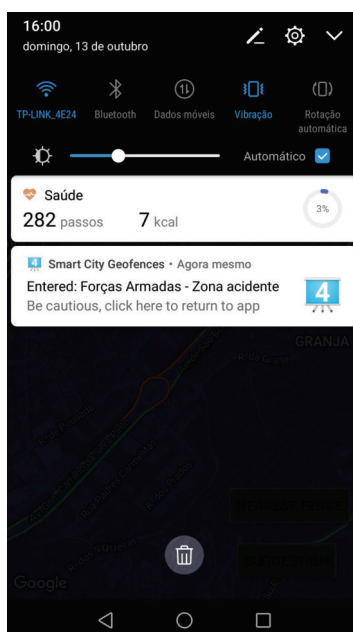


Figura 37: Notificação recebida pelo condutor

5.2 RESULTADOS

O caso de estudo foi realizado num telemóvel *Huawei P8 Lite 2017* para o condutor e num *Xiaomi 5X* para o *VRU*.

Como podemos ver na figura 38, verificou-se que no geral, para um *VRU*, o tempo de envio das notificações rondava os três, quatro segundos. Dado que a sua circulação se dá a baixa velocidade, pois estes deslocam-se a pé ou eventualmente de bicicleta, este tempo de resposta é aceitável e praticável. Provou-se que enviando este tipo de avisos, se consegue atempadamente garantir que um *VRU* ajuste o seu comportamento na via pública mediante as informações recebidas. No que diz respeito ao caso de estudo na vertente de um condutor, já encontramos resultados mais díspares. No primeiro caso, que foi feita uma travessia da *geofence* a baixa velocidade, os tempos de resposta mantiveram-se semelhantes aos registados pelo *VRU*, recebendo notificações com cerca de quatro segundos de atraso. No entanto circulando a maior velocidade isto já não se verificou, pois chegou-se a registar um atraso de quase um minuto. Como tal, é necessário que o sinal de *GPS* seja muito preciso. Neste caso em concreto, o dispositivo utilizado para o caso de estudo possui um fraco sinal de *GPS* pelo que acima de uma certa velocidade torna-se bastante impreciso. Desta forma é possível que tenha influenciado negativamente os tempos de resposta no caso estudado em que circulava a maior velocidade. Outros fatores a considerar são o

facto de o caso de estudo se ter realizado num dia chuvoso. Isto influencia por exemplo a cobertura de rede do dispositivo, que por sua vez afeta a precisão do sinal de *GPS*.

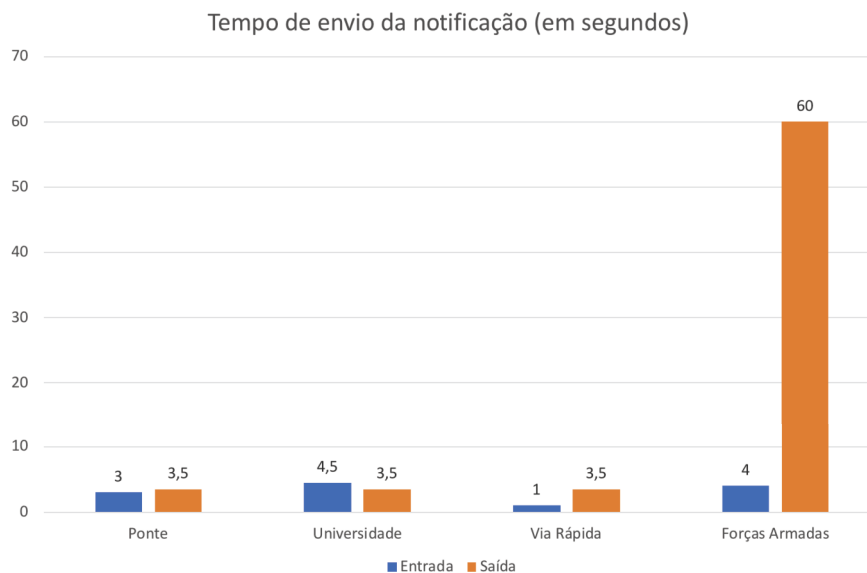


Figura 38: Gráfico com os tempos de reposta das notificações recebidas

Olhando para o funcionamento da plataforma *web* não há grandes considerações a serem feitas. A adição e desenho das *geofences* é quase instantânea. A remoção tanto das *geofences* como das sugestões tem apenas o detalhe de que não é possível remover mais do que uma em cada *submit* ao *form*. Isto constitui uma pequena limitação mas não muito relevante ao funcionamento geral do sistema.

Para a *mobile app*, num caso de estudo à parte chegou-se a verificar que houve uma falha no envio da notificação. A situação na qual isso se deu foi num local onde existiam duas *geofences* consecutivamente. Não tendo reproduzido mais nenhum caso semelhante, não é possível concluir se foi mesmo devido a essa questão ou não.

Outra consideração importante diz respeito ao funcionamento em *background*. A aplicação funciona em *background* e envia notificações caso o telemóvel não esteja bloqueado. Na eventualidade de o telemóvel estar bloqueado, a aplicação só envia as notificações quando ele é desbloqueado. Com desbloqueado não queremos dizer necessariamente que caso tenha ecrã de bloqueio que seja necessário entrar no ecrã principal, basta carregar no botão de desbloqueio que a notificação é recebida. A limitação só ocorre se o utilizador quiser mesmo que o seu telemóvel toque quando entra numa *geofence* e não apenas quando desbloqueia o seu dispositivo, no entanto isto deve-se às atualizações feitas ao sistema operativo *Android*.

Outro fator denotado no caso de estudo diz respeito às diferenças entre *geofences* grandes e pequenas. Em *geofences* pequenas, se o utilizador se estiver a mover muito depressa é

possível que este entre e saia delas sem que o *Location Services* faça uma *query* pela sua posição no intervalo de tempo em que esteve dentro das mesmas. Como tal, pode-se dar uma falha na emissão dos avisos de entrada e ou saída de uma *geofence* que por norma nunca irá acontecer caso a *geofence* tenha um raio grande.

A funcionalidade de pesquisar pela *geofence* mais próxima necessitou que se fizesse uma *query* para receber a posição atual do utilizador, pois caso não seja feita, a *LastKnownLocation* será a localização onde o utilizador iniciou a aplicação. Tendo sido feita esta alteração, o botão passa a fazer uma transição no mapa primeiro para a localização do utilizador e de seguida para a *geofence* mais próxima da sua posição.

A funcionalidade do envio de sugestão é bastante trivial. Não foi detetada qualquer falha, sendo o seu envio para a plataforma *web* quase instantâneo.

Com o que foi referenciado anteriormente, conclui-se que o sistema tem potencial pois é bastante escalável, está no entanto dependente de atualizações à sua *API* para poder tirar partido de mais funcionalidades que acrescentam valor. Conclui-se também que é ainda muito dependente da qualidade dos dispositivos onde é utilizado, pois tendo estes fraca precisão no sinal *GPS* o sistema não executa como esperado.

Com as condições corretas para o funcionamento do sistema, verificamos que apresenta resultados de forma célere, com informação detalhada nos avisos emitidos, com uma plataforma de gestão e aplicação bastante intuitivas e simples, muito escaláveis e fidedignas, capazes de responder às necessidades dos seus utilizadores.

CONCLUSÕES E TRABALHO FUTURO

6.1 SÍNTESE DO TRABALHO REALIZADO

No início desta dissertação definiu-se a criação de um sistema com o intuito de aumentar a segurança dos *VRUs* recorrendo ao conceito de *geofences*. *Geofencing* consiste em criar um perímetro virtual num local real onde é feita a monitorização da presença, entrada e saída de um ou mais utilizadores. Isto prova-se bastante útil no contexto desta dissertação, pois conseguimos mediante a presença de um utilizador nestes locais, enviar avisos inteligentes para ajudar na prevenção de acidentes rodoviários.

Para a criação do sistema, definiu-se que teria de existir uma *mobile application* onde um utilizador pudesse ser notificado da presença, entrada ou saída de uma *geofence* para que pudesse ajustar o seu comportamento, diminuindo o risco de sofrer qualquer tipo de incidente. Definiu-se também que teria de haver alguma entidade responsável por toda a gestão do sistema. Como tal, passou a ser necessário também criar uma plataforma *web* onde este administrador fosse capaz de adicionar, remover e categorizar *geofences*. De forma a estabelecer a ligação entre estas duas componentes era necessário criar uma ponte, que neste caso seria a base de dados onde seriam armazenadas as informações das *geofences* adicionadas. Desta forma, recorreu-se ao *Google Firebase*, uma tecnologia que permite a criação de uma base de dados persistente, de fácil utilização, que comunica em tempo real com tanto a plataforma *web* como a *mobile app* para que mantenham os seus dados atualizados.

Paralelamente idealizou-se que, de forma a complementar o sistema, os utilizadores da *mobile app* deveriam ser capazes de enviar sugestões para o administrador de sistema, de potenciais pontos de interesse que deveriam passar a ser monitorizados. Obviamente que esta funcionalidade acarreta um certo risco, pois requer ao administrador de sistema uma capacidade de triagem dos dados recebidos muito alta, para que consiga filtrar de facto as sugestões realmente importantes e relevantes, daquelas que possam ser falaciosas.

Com todos estes requisitos em mente, construiu-se a plataforma *web*, assim como a *mobile app*, compondo desta forma um sistema robusto e muito escalável para ser integrado

na rede de tecnologias e dispositivos que compõem o mundo das *Smart Cities*. Face aos resultados analisados, verificamos que existe uma contribuição positiva para aumentar a segurança dos *VRUs* pois conseguimos emitir avisos em tempo real para os utilizadores da *mobile app* que circulem dentro dos pontos de interesse onde foram colocadas *geofences*, contribuindo também por consequência, para a diminuição do risco dos demais utilizadores da via pública.

Outra característica importante de referir diz respeito ao facto deste sistema permitir ser aplicado a outros domínios não relacionados com a segurança dos *VRUs*. Sendo escalável podemos definir qualquer tipo de intuito às notificações enviadas nas transições de entrada e saída das *geofences*, pelo que dar informações relativas a locais em concreto, como monumentos, edifícios históricos, entre outros é passível de ser realizado neste sistema. Pode-se ainda pensar também numa vertente comercial, onde os avisos enviados podem estar relacionados com o marketing de uma certa empresa, ou como referido no capítulo da revisão do estado da arte, oferta de promoções para lojas ou estabelecimentos nas redondezas.

Em suma, o sistema desenvolvido para esta dissertação para além do foco na segurança dos *VRUs*, caracteriza-se ainda pela sua escalabilidade, sendo possível aplicá-lo a uma grande variedade de contextos que não o âmbito inicial desta dissertação.

6.2 TRABALHO FUTURO

Existem ainda otimizações possíveis e necessárias para que o sistema se torne mais completo e robusto. Primeiramente, começar por criar uma aplicação disponível para o *Android Auto* de forma a que condutores que circulem na rodovia sejam mais facilmente notificados das suas entradas, saídas e presenças nas *geofences* ativas para que a responsabilidade de adequar o seu comportamento passe também para o seu lado e não somente dos *VRUs*, tendo obviamente ciente que na eventualidade de ter um passageiro com a aplicação ligada já se consegue suprir esta lacuna.

Um aspeto que pode ser escalado diz respeito aos tipos das *geofences*. Neste momento definiu-se que seriam apenas três, mas mediante as necessidades do sistema e toda a sua evolução, futuramente fará sentido adicionar mais alguns para acrescentar ainda mais valor ao sistema.

Outro aspeto, para comodidade e realidade do sistema, diz respeito à forma da figura da *geofence*. De momento todas as *geofences* adicionadas são círculos, pois é assim que a *API* está definida, fazendo todo o sentido que seja possível definir outras geometrias para controlar *geofences*, pois torna a localização das mesmas mais realista.

Outra funcionalidade que poderia ser melhorada diz respeito às sugestões dos utilizadores. Neste momento, o utilizador envia meramente as suas coordenadas geográficas como informação para o administrador o que obriga, como referido anteriormente nesta

dissertação, a que exista uma filtragem e verificação destes dados. Como trabalho futuro as sugestões dos utilizadores deveriam passar a incluir o possível tipo da *geofence* assim como uma pequena descrição que pudesse explicar o porquê de achar que se deveria ter naquele local uma *geofence*, assim como outras informações que o utilizador considerasse relevantes. Assim este canal de *feedback* passaria a ser muito mais completo e traria grandes vantagens a todo o sistema.

BIBLIOGRAFIA

- José J. Anaya, Edgar Talavera, David Giménez, Nuria Gómez, Felipe Jiménez, and José E. Naranjo. *Vulnerable Road Users Detection using V2X Communications*. 2015 IEE 18th International Conference on Intelligent Transportation Systems, 2015.
- François Artusse. *The Importance of Smart Cities*. Available from <https://medium.com/zify/the-importance-of-smart-cities-2a4f7f89a6cd>, n.d.
- ASNR. *Relatorios De Sinistralidade*. Available from <http://www.ansr.pt/Estatisticas/RelatoriosDeSinistralidade/Pages/default.aspx>, 2018a.
- ASNR. *Relatorios De Sinistralidade*. Available from <http://www.ansr.pt/Estatisticas/RelatoriosDeSinistralidade/Documents/2018/RELAT2018b>.
- Bryan Bonack. *Geofences: What They Are, What They Aren't, and Why They're Effective*. Available from <https://www.geospatialworld.net/blogs/geofences-what-they-are-what-they-arent-and-why-theyre-effective/>, n.d.
- A. Boulton and L. Brunn, S.D. and Devriendt. *Cyberinfrastructures and "smart" world cities: Physical, human, and soft infrastructures*. In Taylor, P., Derudder, B., Hoyler, M., Witlox, F. (Eds.), *International Handbook of Globalization and World Cities*. Cheltenham, UK: Edward Elgar, 2011.
- Woong Cho. *Safety enhancement service for vulnerable users using P2V communications*. International Conference on Connected Vehicles and Expo (ICCVE), 2014.
- Aymery Constant and Emmanuel Lagarde. Protecting vulnerable road users from injury. *PLOS Medicine*, 7(3):1-4, 03 2010. doi: 10.1371/journal.pmed.1000228. URL <https://doi.org/10.1371/journal.pmed.1000228>.
- Eyal; Anthony LaMarca; Mahadev Satyanarayanan De Lara. *Location Systems: An Introduction to the Technology Behind Location Awareness*. Morgan Claypool Publishers. p. 88. ISBN 978-1-59829-581-8., 2008.
- R. Duenkler. *Security for vulnerable road users through Ko-TAG*. Available from: <https://www.iis.fraunhofer.de/en/ff/lv/lok/proj/kotag.html>., 2013.
- Bruno Fernandes, Fabio Silva, Cesar Analide, and Jose Maia Neves. *Crowd Sensing for Urban Security in Smart Cities*. *Journal of Universal Computer Science* - June 2018, 2018. POCI-01-0145-FEDER-007043.

- Tonya Garcia. Available from <https://www.marketwatch.com/story/burger-king-trolls-mcdonalds-with-penny-whoppers-to-promote-new-app-2018-12-04>, 2018.
- Fertner C. Kramar H. Kalasek R. Pichler-Milanović N. Meijers E. Giffinger, R. *Smart Cities: Ranking of European Medium-Sized Cities*. Available from http://www.smartcities.eu/download/smart_cities_final_report.pdf, 2007.
- Google. Available from <https://developer.android.com/training/location/geofencing>, n.d.a.
- Google. Available from <https://developers.google.com/location-context/geofencing/?hl=zh-TW>, n.d.b.
- R. E. Hall. *The vision of a smart city*. In *Proceedings of the 2nd International Life Extension Technology Workshop*. Available from <http://www.osti.gov/bridge/servlets/purl/773961-oyxp82/webviewable/773961.pdf>, 2000.
- Shuxian He, Jiangchen Li, and Tony Z. Qiu. *Vehicle-to-Pedestrian Communication Modeling Collision Avoiding Method in Connected Vehicle Environment*. Journal of the Transportation Research Board, 2008.
- R.G. Hollands. *Will the real smart city please stand up?* *City*, 12(3), 303-320. 2008.
- Dmitry Namiot and Manfred Sneps-Sneppé. *Geofence and Network Proximity*. Part of the Lecture Notes in Computer Science book series (LNCS, volume 8121), 2013.
- World Health Organization. *Global Status Report on Road Safety 2015*. Available from <https://books.google.pt/books?hl=pt-PT&lr=id=wV4oDgAAQBAJoi=fndpg=PP1dq=road+safetyots=DJXwwXdYwgsig=HncrieHGuQXUgC160yv=onepageq=road2015>.
- K. Ota, T. Kumrai, M. Dong, J. Kishigami, and M. Guo. Smart infrastructure design for smart cities. *IT Professional*, pages 1–1, 2017. doi: 10.1109/MITP.2017.265110715.
- Kim Suart. Available from <https://mobilewalletmarketer.com/using-ibeacons-geofencing-real-estate-open-house-marketing/>, n.d.
- Tesla. Available from https://www.tesla.com/pt_PT/blog/q3-2018-vehicle-safety-report?redirect=no, 2018.
- TomTom. Available from <https://developer.tomtom.com/maps-sdk-android/geofencing-examples>, n.d.
- Sindhu U. Balaouras S. Dines R. A.-Hayes N. M. Nelson L. E. Washburn, D. *Helping CIOs Understand "Smart City" Initiatives: Defining the Smart City, Its Drivers, and the Role of the CIO*. Cambridge, MA: Forrester Research, Inc., 2010.

SY Wong. *Toyota Develops Automatic Brake System Assisted by GPS Technology for Safety Driving*. Available from: <https://www.mydigitallife.net/toyota-develops-automatic-brake-system-assisted-by-gps-technology-for-safety-driving/>, n.d.