Universidade do Minho
Escola de Engenharia

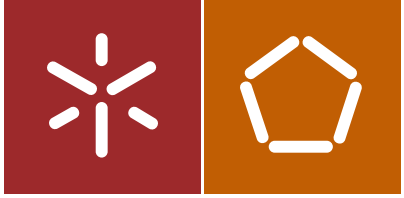
Vera Patrícia Barbosa Barroso

Posicionamento colaborativo em redes Wi-Fi - Where@UM2

Vera Patrícia Barbosa Barroso Posicionamento colaborativo em redes Wi-Fi - Where@UM2

UMinho | 2015

outubro de 2015



Universidade do Minho
Escola de Engenharia

Vera Patrícia Barbosa Barroso

Posicionamento colaborativo em redes Wi-Fi - Where@UM2

Dissertação de Mestrado
Mestrado em Sistemas de Informação

Trabalho efectuado sob a orientação do
Professor Doutor Filipe Meneses
Professor Doutor Adriano Moreira

outubro de 2015

DECLARAÇÃO

Nome: Vera Patrícia Barbosa Barroso

Número do Bilhete de Identidade: 13944938

Endereço de correio eletrónico: pg25201@alunos.uminho.pt

Telefone: 961114454

Título da dissertação: POSICIONAMENTO COLABORATIVO EM REDES WI-FI - WHERE@UM2

Orientadores:
Prof. Doutor Filipe Meneses
Prof. Doutor Adriano Moreira

Ano de conclusão: 2015

Designação do Mestrado: Mestrado em Sistemas de Informação

Universidade do Minho, ___/___/_____

Assinatura: _____

DEDICATÓRIA

Aos meus pais e irmão

AGRADECIMENTOS

Neste momento que completo mais uma etapa importante da minha vida gostaria de expressar o meu agradecimento a todos aqueles que contribuíram com a sua ajuda e apoio.

Primeiramente começo por agradecer aos orientadores, Professor Doutor Filipe Meneses e Professor Doutor Adriano Moreira, pelo facto de me terem aceitado neste projeto, pela ajuda e disponibilidade prestada. Foi de facto um grande privilégio realizar esta dissertação sob a vossa orientação.

Aos meus pais Maria Emília e Manuel Barroso e ao meu irmão Miguel Barroso por todo o apoio e incentivo que deram não só durante a dissertação, mas durante todo o meu percurso académico. Agradeço ainda mais os sacrifícios suportados para concluir mais esta etapa da minha vida. Agradeço também aos meus avôs Amélia e Porfírio Barbosa que sempre me incentivaram e apoiaram.

Quero agradecer aos meus amigos que foram muito importantes nesta fase da minha vida, mesmo alguns estando distantes, nomeadamente ao Ilídio Sousa pelo apoio e incentivo, ao Ricardo Amaral pelos bons conselhos apoio e ajuda sempre que necessário, sendo ele e o Carlos Moreira os companheiros de grupo insubstituíveis ao longo destes dois anos, um muito obrigada. À Ana Francisca Amorim e à Vânia Alves pelos bons momentos e apoio. Ao Ângelo Fonseca pelos conselhos e ajuda sempre que necessário. Ao José Luís Fernandes e ao Rui Sineiro pela ajuda e conselhos que foram fulcrais para tomar algumas decisões durante a dissertação.

A todos vocês, um muito Obrigada!

RESUMO

Nos dias de hoje, os dispositivos móveis são objetos indispensáveis na vida das pessoas, tendo um forte impacto no dia-a-dia delas. Estes dispositivos integram várias tecnologias, as quais são constantemente exploradas em várias áreas, dentro delas o posicionamento no interior de edifícios.

Neste momento existem várias aplicações que determinam a posição de pessoas, objetos com grande precisão, mas em ambientes exteriores usando o sistema de localização GPS. Contudo estimar a posição de uma pessoa num ambiente interior com grande precisão é mais complexo, por isso muitos investigadores exploram cada vez mais esta área.

Normalmente, para estimar a posição no interior de edifícios pretende-se utilizar as infraestruturas instaladas neles e assim proporcionar soluções de baixo custo. Nesta situação, os sistemas de posicionamento baseados na técnica Wi-Fi *fingerprinting* são os que mais se destacam. Para a implementação desta técnica, normalmente são usados os dispositivos móveis das pessoas. As pessoas ao estarem dentro de um edifício, os dispositivos móveis conseguem detetar pontos de acesso Wi-Fi e recolhem dados sobre eles, juntamente com o respetivo nível de sinal detetado e enviam esses dados a um serviço que estima a localização, comparando as *fingerprints* com um mapa de rádio previamente construído.

O propósito desta dissertação centra-se em melhorar um sistema de posicionamento, já existente, baseado na técnica Wi-Fi *fingerprinting*, que posiciona tanto em ambientes interiores como exteriores. Então são descritas e implementadas soluções de melhoria do sistema. Também é apresentada uma solução para verificar a disponibilidade dos serviços do sistema de posicionamento, que consiste na implementação de uma ferramenta de monitorização.

Palavras-chave: Localização *indoor/outdoor*, posicionamento, técnica *fingerprinting*, *Wi-Fi*, redes sem fios, *web services*, aplicação móvel, Android, monitorização.

ABSTRACT

Nowadays mobile devices are indispensable objects in people's lives and they have a strong impact in their daily routines. These devices integrate several technologies, which are constantly explored in several areas, e.g. indoor positioning.

Currently there are several solutions that estimate the position of people, with great precision, in outdoor environments using the positioning system GPS. However, estimating the position of a person in an indoor environment with great precision is more complicated, so many researchers increasingly explored this area.

Normally, to estimate the indoor position the installed infrastructure is utilized therein and thus providing low-cost solutions. In this situation, the positioning systems based on fingerprint technique are the most distinct. To implement this technique, mobile devices of people are normally used. While individuals are inside buildings, mobile devices can detect Wi-Fi access points, and collect data on them, including the respective detected level signal, and sent the collected data to a service that estimates the location by comparing the fingerprints with a pre-built radio map.

The purpose of this dissertation focuses on improving an existing positioning system based on Wi-Fi fingerprint technique. This system works in indoor and outdoor environments. Solutions that improve the system are described in this document along with a presentation of the implemented solutions. It is also presented a solution to check availability of the positioning system services, consisting of a monitoring tool.

Keywords: Indoor/outdoor location, positioning, fingerprint technique, Wi-Fi, wireless networks, web services, mobile application, Android, monitoring.

ÍNDICE

ÍNDICE DE TABELAS

ÍNDICE DE ILUSTRAÇÕES

SIGLAS E ACRÓNIMOS

ADT	Android Developer Tools
AP	Access Point
App	Aplicação
BD	Base de Dados
DNS	Domain Name System
FTP	File Transfer Protocol
GSM	Global System for Mobile Communications
GPS	Global Positioning System
HTTP	HyperText Transfer Protocol
IP	Internet Protocol
kNN	k-Nearest-Neighbor
MAC	Media Access Control
NNTP	Network News Transfer Protocol
PHP	Hypertext Preprocessor
POP3	Post Office Protocol
RSSI	Received Signal Strength Indicator
SDK	Software Development Kit
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SVM	Support Vector Machine
UM	Universidade do Minho
Wi-Fi	Wireless Fidelity
XML	eXtensible Markup Language

1. INTRODUÇÃO

1.1. ENQUADRAMENTO

Nos últimos anos, o rápido desenvolvimento das tecnologias, mais precisamente os smartphones, permitem-lhes desempenhar um papel importante no nosso quotidiano. Os preços acessíveis dos smartphones fazem com que o número de utilizadores aumente radicalmente. As aplicações móveis dos smartphones podem facilitar e apoiar as atividades diárias dos seus utilizadores. Muitos dos smartphones integram o módulo GPS (*Global Positioning System*) que fornece as coordenadas geográficas da posição do utilizador usando sinais provenientes de uma rede de satélites. Contudo, esta função pode falhar quando utilizada em ambientes interiores, uma vez que os sinais GPS podem não se propagar através das estruturas dos edifícios. Assim, são necessárias técnicas de localização *indoor* adicionais, a fim de proporcionar um posicionamento mais preciso no interior dos edifícios (Boonsriwai & Apavatjrut, 2013). A localização *indoor* é cada vez mais explorada nas aplicações móveis, e existem já várias aplicações para determinar a posição dos utilizadores, mas utilizam o sistema de localização GPS. Os sistemas de posicionamento baseados nas técnicas Wi-Fi (*Wireless Fidelity*) *fingerprinting* são dos que mais se destacam no que diz respeito à exploração das infraestruturas habitualmente instaladas em edifícios. Esta é uma tecnologia de muita procura pois pode proporcionar soluções de baixo custo para o posicionamento dos dispositivos móveis e dos seus utilizadores. A técnica Wi-Fi *fingerprinting* é baseada num mapa de intensidade do sinal de rádio. Para se estimar a posição de um dispositivo é estabelecida uma relação entre as medições em tempo real do sinal e o conteúdo presente na base de dados (Fernandes, 2012).

1.2. OBJETIVOS

O objetivo deste projeto é resolver o problema de estimar a posição dos utilizadores no interior dos edifícios, e para isso será utilizada uma técnica que não necessite de qualquer infraestrutura extra, mas que consiga reutilizar a infraestrutura já existente no ambiente.

Para a execução deste projeto dispomos de um sistema de posicionamento colaborativo desenvolvido no âmbito de um trabalho de mestrado anterior, o qual inclui uma aplicação móvel Android e um sistema central de processamento e armazenamento de dados. O nosso trabalho passa por melhorar este sistema para que construa e mantenha os mapas de rádio de forma

dinâmica para ambientes interiores de grandes edifícios, ou conjuntos de edifícios, baseado na técnica Wi-Fi *fingerprinting*.

Este projeto tem como objetivos principais desenhar um mecanismo para otimização dos mapas de rádio e integrar este com o motor de posicionamento já existente. Ao mesmo tempo, deverá também ser desenhada uma estratégia para avaliar a credibilidade dos dados fornecidos pelos utilizadores através da aplicação móvel, para que possa ser construído um bom mapa de rádio do motor de posicionamento. Pretende-se ainda definir novas funcionalidades para a aplicação móvel de modo a atrair os utilizadores para uma maior utilização da aplicação. Por fim, deve ser encontrada uma solução que monitorize o servidor e garanta a sua disponibilidade permanente. Uma vez estas tarefas concluídas deverão ser implementadas e testadas.

1.3. ABORDAGEM METODOLÓGICA

A presente secção descreve a metodologia utilizada para dar suporte à elaboração desta dissertação. É também apresentada a estratégia de pesquisa utilizada para a recolha de documentação sobre a área de investigação.

1.3.1. METODOLOGIA

Para a realização desta dissertação, a abordagem escolhida foi a *Design Science*. Optou-se por esta abordagem por se tratar do desenvolvimento de um artefacto, neste caso a melhoria de um sistema de posicionamento em ambientes interiores.

O método *Design Science* é composto por cinco fases: consciencialização do problema, sugestão, desenvolvimento, avaliação e conclusão, como se pode observar na Ilustração 1.

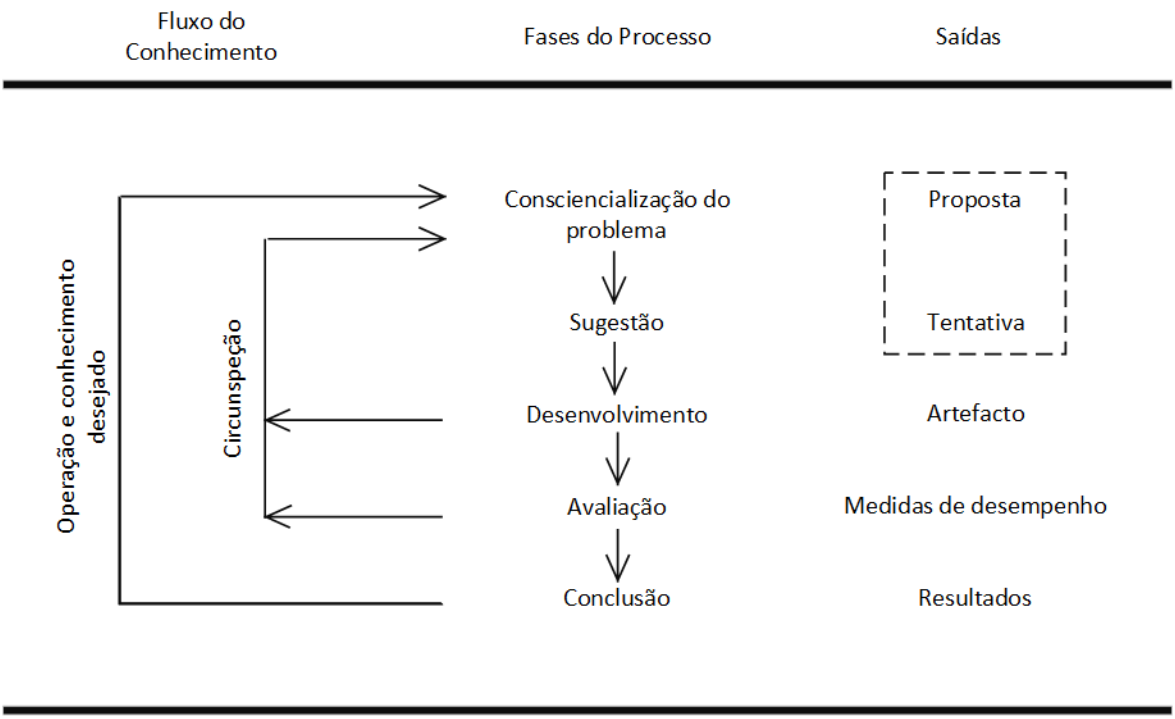


Ilustração 1 - Metodologia *Design Sciense* (adaptado de Järvinen, 2007).

Na primeira fase do problema é feita a familiarização sobre a área de estudo, neste caso a tecnologia de posicionamento em ambientes interiores, nomeadamente com técnicas de Wi-Fi *fingerprint,* para assimilar conhecimentos. O resultado desta fase é a descrição do capítulo 2 desta dissertação.

No passo seguinte é sugerida uma solução, neste caso foram sugeridas melhorias do sistema que se consideraram fundamentais.

A fase seguinte é o desenvolvimento da solução, que é a parte do trabalho que exige mais tempo, tendo como resultado o artefacto, nesta dissertação a implementação das melhorias sugeridas.

Na fase de avaliação é onde se valida o artefacto.

Por fim, a conclusão onde se determina a qualidade e impacto do artefacto produzido.

1.3.2. ESTRATÉGIA DE PESQUISA

O enorme volume de documentos científicos faz com que as pesquisas sejam um processo complexo. Por isso foram utilizados alguns critérios na pesquisa, para facilitar a procura de documentos relevantes, como datas, citações e a escolha da área do tema em questão.

Para a pesquisa destes documentos foram utilizadas ferramentas de pesquisa como Google Scholar[1], pois é dirigido aos universitários/científicos, e esta é capaz de dar um vasto número de resultados de artigos científicos, teses, livros, resumos, etc.; o RepositoriUM[2] porque contém já artigos de alunos e professores da universidade nesta área; o RCCAP[3], pois o acesso é livre e conseguiu-se ter acesso a milhares de documentos de caracter científico e académico, como revistas científicas, dissertações, conferências, etc., distribuídos por vários repositórios portugueses; o Web of Science[4] porque além da pesquisa habitual por ocorrência de palavras permite também a pesquisa de artigos relacionados e estabelece ligações entre artigos que citam outros; por fim, o B-on[5] pois disponibiliza acesso ilimitado aos textos integrais de artigos científicos e *ebooks online*.

Para os documentos que serviram de base para obter conhecimento na área, foram utilizadas as seguintes combinações de palavras-chave, apresentadas na Tabela 1.

Tabela 1 - Combinações de palavras-chave utilizadas na pesquisa no âmbito da dissertação.

Palavras-chave	*Keywords*
	Wi-Fi, fingerprint
Wi-Fi, posicionamento	*Wi-fi, positioning*
Técnicas de posicionamento no interior de edifícios	*Indoor positioning techniques*
Sistema de posicionamento no interior de edifícios	*Indoor positioning system*
	User feedback in indoor positioning
Sistema de localização no interior de edifícios	*Indoor Location System*

Após levantamento dos documentos foram selecionados os mais relevantes. Foram levados em conta critérios de seleção, como o título, resumo, principalmente o resumo porque contém informações essenciais, para concluir se é um artigo relevante para o tema de investigação. As

[1] https://scholar.google.pt/

[2] http://repositorium.sdum.uminho.pt/

[3] https://www.rcaap.pt/

[4] https://www.webofknowledge.com/

[5] http://www.b-on.pt/

palavras-chave também foi um critério tido em conta, visto refletirem os aspetos mais importantes dos documentos pesquisados.

1.4. ESTRUTURA DA DISSERTAÇÃO

A presente dissertação está dividida em oito capítulos. Neste primeiro capítulo é realizado um enquadramento ao tema da dissertação, são apresentados os objetivos que se pretendem alcançar, bem como uma descrição da abordagem metodológica adotada.

O segundo capítulo tem como objetivo apresentar uma análise ao contexto atual, que assenta em cinco partes. São analisadas algumas tecnologias que são utilizadas em posicionamento *indoor*, e quais as técnicas de localização existentes. É feita uma análise mais aprofundada à técnica *fingerprinting.* São descritos alguns sistemas de localização existentes em redes sem fios. Por fim é realizada uma análise aos sistemas colaborativos.

No terceiro capítulo são apresentados algumas ferramentas de monitorização e é escolhida qual a melhor ferramenta para monitorizar o sistema de posicionamento existente.

O quarto capítulo incide no estudo do problema da dissertação, onde é descrito o sistema de posicionamento existente e apresentada a arquitetura geral deste. É também descrita a abordagem do processo de posicionamento existente e descrito o processo da criação e manutenção do mapa de rádio. Por fim são identificados os problemas que se pretendem resolver nesta dissertação.

No quinto capítulo são delineadas algumas soluções para os desafios apresentados no capítulo anterior, são apresentadas estratégias para fomentar a interação com a aplicação móvel e também uma estratégia para avaliar a qualidade e credibilidades dos dados inseridos pelos utilizadores através da aplicação móvel.

O sexto capítulo descreve todo o processo da implementação das melhorias propostas no capítulo anterior e também o processo da implementação da ferramenta de monitorização escolhida.

No sétimo capítulo é apresentada a avaliação do sistema, expondo os testes realizados e os resultados obtidos destes mesmos testes.

Por fim, no capítulo oito são apresentadas as conclusões do trabalho desenvolvido e também delineadas algumas propostas para continuação da melhoria do sistema de posicionamento.

2. POSICIONAMENTO NO INTERIOR DE EDIFÍCIOS

Neste capítulo é apresentada uma breve descrição de algumas tecnologias e técnicas usadas para estimar a posição em ambientes *indoor*. É também elaborada uma descrição de alguns sistemas de localização que são referências nesta área.

2.1. TECNOLOGIAS SEM FIOS USADAS EM POSICIONAMENTO *INDOOR*

2.1.1. IDENTIFICAÇÃO POR FREQUÊNCIA DE RÁDIO

Identificação por frequência de rádio (RFID – *Radio Frequency IDentification*) é uma tecnologia de identificação que permite a leitura e escrita de dados através de sinais de rádio.

Um sistema RFID tem como seus componentes, leitores e *tags* RFID. Estas *tags* armazenam informação que pode ser lida pelos leitores. Normalmente estas *tags* são integradas num objeto, o que faz com que seja possível a sua identificação e rastreio. Existem dois tipos de *tags*, as passivas e as ativas. As *tags* passivas são mais leves e menos caras do que as ativas. Estas funcionam com a energia que é enviada, via rádio, pela antena do leitor. A antena da *tag* recebe essa energia utilizando parte dela para a alimentação da *tag* e a outra parte para o envio dos dados armazenados ao leitor, não necessitando de nenhuma fonte de alimentação. As *tags* ativas não necessitam ser alimentadas via rádio, pois elas contêm uma bateria que desempenha essa função. Estas estão constantemente a emitir sinais permitindo que a antena de um leitor as possa detetar (Ni, Liu, Lau, & Patil, 2004).

RFID é aplicada em áreas como proteção contra roubo ou deslocação não autorizada de equipamentos valiosos, é possível vigiar artigos eletronicamente, controlar o acesso de viaturas a instalações ou parques de estacionamento, controlar o acesso de pessoas a locais de acesso restrito, entre outras (Couto, 2003).

Esta poderia ser uma solução para a localização em ambientes *indoor.* Com implementação desta solução o utilizador transportaria consigo um leitor, por exemplo numa mochila, enquanto as *tags* estariam instaladas por todo o edifício. Normalmente as *tags* ativas estão no teto e as passivas no chão do edifício. Com esta técnica não só seria possível obter a localização absoluta do utilizador, mas também informações de deslocamento devido ao movimento (Sanpechuda & Kovavisaruch, 2008). Contudo seria preciso criar toda uma infraestrutura, que iria implicar custos

de montagem e equipamentos dedicados, e esse é um problema que se pretende resolver usando a infraestrutura existente no ambiente interior.

2.1.2. BLUETOOTH

Bluetooth é uma tecnologia sem fios que pode ser utilizada para comunicação de curto-alcance entre diferentes dispositivos. Quando dois dispositivos estão ao alcance um do outro, estes comunicam entre si e determinam se há alguma informação a ser transmitida. Durante uma comunicação inicial, é necessário criar uma relação entre dispositivos desconhecidos, este processo tem o nome de emparelhamento. Qualquer pessoa com um recetor apropriado pode intercetar transmissões via Bluetooth. A fim de reduzir este tipo de problemas, o Bluetooth suporta a autenticação de dispositivos, ou seja, dispositivos que partilham informações usam uma chave secreta partilhada ou a chave de ligação para autenticar um dispositivo ao outro (Dursch, Yen, & Shih, 2004).

Esta tecnologia poderia ser outra solução para o problema de localização em ambientes interiores, é uma tecnologia presente na maioria dos dispositivos móveis, cada dispositivo contém um identificador único que possibilita a sua identificação, e poderia ser utilizado para determinar a sua localização. No entanto, tal como a tecnologia RFID, seria necessário criar toda a infraestrutura que iria implicar custos e o que se pretende é resolver este problema usando a infraestrutura existente no edifício.

2.1.3. WI-FI

Wi-Fi é uma tecnologia sem fios, que usa ondas de rádio, para permitir a transferência de dados em alta velocidade em distâncias curtas.

A base de qualquer rede sem fios é um AP (*Access Point*) e a sua principal tarefa é transmitir um sinal que os dispositivos possam detetar e coordenar a comunicação entre os diversos nós da rede.

Para os dispositivos poderem conectar-se a um AP e juntarem-se a uma rede sem fios, estes devem estar equipados com os adaptadores de rede sem fios (Beal, 2015).

Cada vez mais há APs em locais públicos, como aeroportos, hotéis e cafés, oferecidos pelas respetivas entidades, e algumas cidades têm mesmo construído redes Wi-Fi de acesso gratuito por toda a cidade.

A popularidade do Wi-Fi tem crescido constantemente, esta tecnologia permite que redes locais (LANs) funcionem sem fios, tornando-se uma escolha popular para redes domésticas e empresariais. A tecnologia Wi-Fi também pode ser usada para fornecer acesso à Internet de banda larga sem fios a dispositivos, como smartphones, computadores, tablets e consolas (The Editors of Encyclopædia Britannica, 2014).

Como referido, esta é uma tecnologia que existe na maioria dos locais frequentados pelos utilizadores, visto já um elevado número de pessoas possuírem um smartphone capaz de se ligar a uma rede Wi-Fi, e estes já serem utilizadores assíduos desta tecnologia para acederem à maioria das aplicações móveis que necessitam de acesso à Internet. Logo será a tecnologia ideal para implementar num sistema de posicionamento em ambientes *indoor*, pois podemos aproveitar as infraestruturas existentes nesses locais.

2.2. TÉCNICAS DE LOCALIZAÇÃO

Como já visto anteriormente, o GPS não é bem-sucedido para determinar o posicionamento em ambientes interiores, onde o sinal pode ser fraco ou até nulo, afetando assim a acuidade da posição. Nos dias de hoje já existem técnicas para resolver o problema de posicionamento *indoor*. Estas podem ser organizadas em três tipos, triangulação, proximidade e análise de cenário (Brás, 2009). A seguir são apresentadas, sucintamente, algumas técnicas existentes para determinar a posição dos utilizadores em ambientes *indoor*.

2.2.1. TRIANGULAÇÃO

Triangulação é uma técnica que usa as propriedades geométricas de triângulos para estimar a localização do alvo. Tem duas derivações, uma delas a *lateration*, que calcula a posição de um dispositivo através da medição das distâncias a vários pontos de referência (Liu, Member, Darabi, Banerjee, & Liu, 2007). Esses pontos de referência são transmissores de sinal e a sua localização deve ser conhecida. A posição do utilizador pode ser calculada baseada na medição da distância entre o ponto de referência e a sua localização. Após essa medição é criado um círculo onde o raio tem precisamente essa mesma medição, o centro do círculo é o ponto de referência. Este procedimento deve ser feito também para os outros dois pontos de referência, depois de se ter os três círculos é possível saber a interceção entre eles, essa interceção é a posição do utilizador (Hightower & Borriello, 2001). Também pode ser baseada noutra técnica, *angulation*, são utilizados

ângulos para determinar a posição de um objeto. Em geral, num plano bidimensional são necessárias pelo menos duas medições de ângulos de dois pontos de referência, e a medida da distância entre os dois pontos de referência para a estimativa da localização do objeto. (Hightower & Borriello, 2001).

2.2.2. PROXIMIDADE

Na técnica de proximidade o objeto a localizar é descoberto pelo ponto de referência que estiver mais próximo, ou seja, ao sabermos a localização dos pontos de referência sabemos a localização do objeto. Este método pode ser obtido através de 3 processos: contacto físico, monitorização dos pontos de acesso sem fio e observação de sistemas de identificação automática.

Contacto físico consiste em detetar o toque físico entre o objeto a localizar e o ponto de referência. O ponto de referência pode ser baseado em sensores de pressão ou de toque. Ao se conhecer a posição do ponto de referência é possível saber a posição do objeto a localizar.

O método monitorização dos pontos de acesso sem fio consiste num dispositivo móvel detetar um ou mais pontos de acesso. Caso o dispositivo detete mais do que um ponto de acesso, ele usa o valor máximo de RSSI (*Received Signal Strength Indicator*) para escolher o ponto de acesso à qual se vai ligar.

Observação de sistemas de identificação automática é o método onde a partir das ações da pessoa se pode estimar a sua posição. Por exemplo, ao fazer um pagamento com um cartão de crédito, registos de telefones fixos, passar nas portagens, etc. Ou seja, uma pessoa que use o cartão num terminal multibanco é possível saber quando e onde este foi utilizado, podendo-se estimar a localização do utilizador (Hightower & Borriello, 2001).

2.2.3. ANÁLISE DE CENÁRIOS

Análise de cenários é outra técnica que consiste em observar um ambiente e verificar os seus padrões e a sua variação ao longo do tempo. Esta técnica pode ser conseguida a partir da análise de imagens obtidas a partir de câmaras de vídeo. Estas imagens são analisadas e processadas de maneira a retirar todas as informações importantes para descobrir o objeto a localizar. Mas, por exemplo, se se pretende localizar todas as pessoas dentro de um edifício, então define-se que estas tenham algumas características específicas, como todas as pessoas têm que ter vestido uma camisola com determinado símbolo estampado. Ao ser analisada a imagem o

algoritmo vai ter em conta estas características e assim que as encontrar, encontra também as pessoas (H. T. G. Pinto, 2009).

Esta técnica também pode ser conseguida através do método *fingerprint*, que consiste na análise das *fingerprints* eletromagnéticas do cenário. Após a análise há uma característica única ou um conjunto de características de um cenário, que diferencia um cenário de outros. Essas características são recolhidas e comparadas com os dados existentes na base de dados para cada cenário (Disha, 2013).

2.3. TÉCNICA FINGERPRINTING

Esta técnica é composta por duas fases, a fase de calibração (*offline*) e a de localização (*online*). Durante a fase de calibração, a intensidade do sinal RSSI dos pontos de acesso e as coordenadas de localização são recolhidas nos diferentes locais dos edifícios selecionados. Cada local deve conter uma lista dos valores de intensidade do sinal RSSI para todos os AP visíveis a partir desse local. Isto deve ser feito, pois a posição do utilizador/dispositivo móvel pode influenciar os valores medidos pelo ponto de acesso. Por exemplo, o dispositivo pode estar numa mochila, ou o próprio utilizador estar entre o dispositivo móvel e o ponto de acesso, e o valor da intensidade do sinal RSSI será menor do que se o utilizador tivesse o dispositivo móvel na direção do ponto de acesso sem nenhum obstáculo para atenuar o sinal (Fernandes, 2012). O objetivo desta operação é a construção da base de dados de *fingerprints* (mapa de rádio) que será utilizada na fase de localização.

Na fase de localização a *fingerprint* enviada pelo dispositivo móvel ao sistema de localização é comparado com o conjunto de dados armazenados no mapa de rádio para identificar os melhores pontos de referência e assim conseguir obter a localização do dispositivo (Disha, 2013).

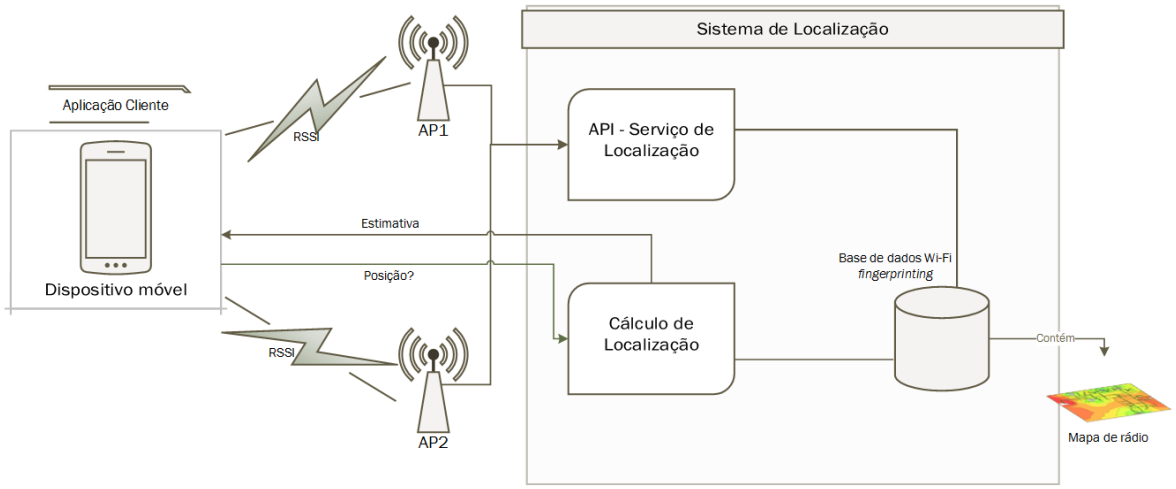


Ilustração 2 - Esquema geral da fase de localização da técnica *Fingerprinting* (Fernandes, 2012).

Na Ilustração 2 está representado um esquema geral da fase de localização. Nela pode ser visto que o sistema de localização contém a base de dados com as *fingerprints.* O dispositivo móvel solicita a sua posição enviando os seus dados, o sistema faz o cálculo da localização comparando a *fingerprint*, enviada pelo dispositivo, com as *fingerprints* contidas na base de dados e, por fim, concede ao dispositivo móvel uma estimativa da posição do utilizador (Fernandes, 2012).

Um dos desafios da técnica baseada no método *fingerprinting* é que os pontos de acesso podem ser movidos dos locais onde se encontram, ou até mesmo serem substituídos, e nestes casos iria ser necessário proceder novamente à calibração dos locais afetados.

Segundo Couto (2003) existem vários algoritmos, para determinar o posicionamento em ambientes interiores baseados no método *fingerprinting*, como métodos probabilísticos, kNN (*k-Nearest-Neighbor*), SVM (*Support Vector Machine*), redes neuronais artificiais, entre outros.

2.4. SISTEMAS DE LOCALIZAÇÃO EM REDES SEM FIOS

Temos visto que, para resolver problemas de localização em ambientes interiores, como universidades, prédios, casas, etc., os investigadores têm apostado na exploração dos sinais em redes Wi-Fi. As redes Wi-Fi têm-se tornado cada vez mais comuns, não só nos ambientes domésticos, mas também em locais públicos. A seguir são apresentados alguns sistemas de localização.

2.4.1. RADAR

Desenvolvido pela Microsoft Research, RADAR foi o primeiro sistema de posicionamento em ambientes interiores baseado na intensidade do sinal Wi-Fi, utilizando a técnica *fingerprinting.* No processo de determinar a posição e a localização dos dispositivos são necessárias duas fases, a *offline* e a *online*. Na fase *offline*, é recolhida e armazenada a informação temporal e o valor do indicador de intensidade do sinal recebido, sobre a área alvo (Fernandes, 2012). Na fase *online* são reportados sinais recebidos de cada estação, através de unidades móveis. Assim, é determinada a melhor correspondência entre as observações *online* e qualquer ponto do modelo *offline*. Por fim, a estimativa da localização é o ponto de localização que receber a melhor correspondência (Coelho, 2009).

2.4.2. EKAHAU

Este é um sistema de localização em tempo-real, desenvolvido no ano de 2000 na Finlândia. Considerado, nos dias de hoje, dos sistemas de localização mais precisos, apresenta um erro médio abaixo de 1 metro na localização de dispositivos móveis (Bisatto & Peres, 2009). Com este sistema, as posições dos dispositivos são atualizadas automaticamente e podem ser partilhadas com outros sistemas que requeiram esta informação. Tem também a vantagem de a localização ser processada através de um dispositivo central (Pereira, 2012).

2.4.3. HORUS

O sistema Horus funciona em duas fases, a fase *offline* onde é construído o mapa de rádio e é aplicado o processo de *clustering* nas localizações do mapa de rádio, seguindo-se um novo pré-processamento dos modelos de intensidade do sinal obtido. Na fase *online* é estimada a localização do utilizador com base na intensidade do sinal obtido a partir de cada ponto de acesso e no mapa de rádio construído na fase *offline*. O mapa de rádio armazena a distribuição de intensidade do sinal obtido a partir de cada ponto de acesso em cada local. Há dois modos de operação do sistema Horus, um usa distribuições não paramétricas e outro usa distribuições paramétricas. Os resultados experimentais mostram que este sistema pode adquirir uma precisão de 90% para um erro abaixo de 2,1 metros (Youssef & Agrawala, 2005).

2.4.4. REDPIN

Este é um sistema de posicionamento em ambientes interiores, baseado na conjunção da intensidade do sinal Wi-Fi, Bluetooth e GSM (*Global System for Mobile Communications*), este mostra a divisão do edifício em que o utilizador se encontra. É um sistema baseado em *fingerprints*, Redpin não fornece as coordenadas geográficas mas sim o nome ou o número da divisão. Este sistema permite ao utilizador inserir os nomes das divisões se necessário, e é capaz de adaptar-se às mudanças no ambiente interior, provocadas por exemplo, pela substituição dos pontos de acesso. Trata-se de uma aplicação concebida para dispositivos Android e iOS. As experiências iniciais mostraram que o sistema atinge uma taxa de sucesso de cerca de 90% na estimação da localização dos dispositivos.

Redpin emprega um mecanismo de posicionamento de duas fases. Se as *fingerprints* consistirem em apenas algumas medições, Redpin utiliza uma variante do algoritmo kNN, enquanto o algoritmo SVM é usada para *fingerprints* com grande número de medições (Redpin, 2008).

2.4.5. CONCLUSÃO

Na Tabela 2 é apresentada uma comparação entre os sistemas analisados. Nesta tabela está presente a informação relativa a tecnologia, algoritmo utilizados e o desempenho de cada um dos sistemas.

Tabela 2 - Comparação entre sistemas de localização (adaptado de Gu, Lo, Member, & Niemegeers, 2009).

Sistema	Tecnologia	Algoritmo	Desempenho
RADAR	WLAN (RSSI)	kNN	50% probabilidade de obter um erro à volta dos 4 metros
Ekahau	WLAN (RSSI)	*Lateration*	Apresenta erros de 1 metro
Horus	WLAN (RSSI)	Método probabilístico	Precisão de 90% dentro de 2,1 metros
Redpin	WLAN (RSS)	kNN	Taxa de sucesso

	Bluetooth GSM	SVM	de cerca de 90%.

2.5. SISTEMAS COLABORATIVOS

Neste nosso projeto está implementado um sistema colaborativo onde os utilizadores, sem que disso se apercebam, colaboram na construção do mapa de rádio. Desta forma, com a ajuda de todos os utilizadores, será mais rápido e fácil conseguir um maior número de *fingerprints* para o mapa de rádio.

A rápida evolução tecnológica dos dispositivos móveis têm vindo a criar novas oportunidades para os sistemas colaborativos. Graças às múltiplas tecnologias de conectividade, estes são capazes de enviar dados de forma simples, rápida e com custos relativamente baixos.

Cada vez mais os investigadores optam por incorporar colaboração nos sistemas de posicionamento em ambientes interiores, nomeadamente em abordagens baseadas em *fingerprint*, a fim de melhorar o desempenho do sistema. A seguir serão descritos alguns sistemas colaborativos de referência existentes nesta área.

Como já foi possível perceber anteriormente, o Redpin é um sistema onde os utilizadores podem inserir o nome das divisões onde se encontram.

Bhasker, Brown, & Griswold (2004) apresentam um método para dedução da localização assistida pelo utilizador com base na intensidade do sinal *wireless*. Os utilizadores ao não concordarem com informações que o sistema exibe podem fazer correções, por exemplo a localização do utilizador estar errada.

Gallagher, Li, Dempster, & Rizos (2010) investigam um método para utilizar o *feedback* do utilizador como uma forma de monitorização de alterações na infraestrutura Wi-Fi. É baseado num sistema de "pontos" dado a cada AP na base de dados. Sempre que um AP estiver desligado, o número de pontos associado a esse AP reduzirá gradualmente à medida que os utilizadores dão o seu *feedback*. Até que, eventualmente, é excluído da base de dados. Se um novo AP for instalado, o sistema irá detetá-lo e atualizará a base de dados com novas *fingerprints*.

É também proposto por Mahtab Hossain, Nguyen Van, & Soh (2010), uma técnica de *fingerprint* baseada em interpolação usando o *feedback* do utilizador, que não exige uma fase de treino exaustivo. Os autores defendem que o *feedback* dos utilizadores ajudam a afinar/melhorar

um sistema de posicionamento ainda numa fase inicial e que estes sistemas baseados no *feedback* dos utilizadores se adaptam muito bem a mudanças no ambiente. Defendem também que se os utilizadores forem bem comportados, a participação dos utilizadores finais pode realmente ajudar na construção de um sistema de posicionamento de forma incremental a partir do zero.

O Herecast (Paciga & Lutfiyya, 2004) é um sistema de localização que funciona em ambientes *indoor* e *outdoor*, baseado na tecnologia Wi-Fi. Este sistema para estimar a localização recolhe informação sobre o ponto de acesso Wi-Fi ao qual está conectado. Desta forma, é possível saber a localização do utilizador visto os pontos de acesso possuírem um endereço MAC (*Media Access Control*), que é um identificador único. O sistema contém uma base de dados que disponibiliza informação relativa à localização dos pontos de acesso como, país, província, cidade, rua, nome do edifício, andar e número do apartamento.

Sempre que um novo ponto de acesso é detetado por o dispositivo de um utilizador, este pode introduzir a informação relativa à localização do ponto de acesso ao qual está conectado. Esta informação é guardada na base de dados do sistema Herecast permitindo outros utilizadores utilizarem estas informações para saberem qual a sua localização.

O Molé (Ledlie et al., 2012) é um motor de localização móvel com recurso à infraestrutura Wi-Fi. Este é dividido em dois componentes cliente e servidor. O cliente tem uma interface do utilizador e em *background* faz leituras periódicas dos sinais Wi-Fi e estima o local atual às restantes aplicações do mesmo dispositivo. A interface do utilizador mostra ao utilizador a sua posição atual e alguns dados relativos às leituras como, número de pontos de acesso detetados e número de leituras usado para estimar o local. O servidor do sistema contém quatro componentes, *Map Server*, *Bind Database*, *Fingerprint Builder* e *Fingerprint Server*. Estes permitem os utilizadores introduzirem e acederem às *fingerprints* guardadas no componente *Fingerprint Server* do servidor. O *Map Server* recebe e reencaminha as leituras do local adicionadas pelos clientes para a *Bind Database*. E também é responsável por disponibilizar aos clientes as possíveis áreas em que estes se encontram, para estimar a área o cliente recolhe o endereço MAC de um ponto de acesso. O *Fingerprint Builer* consulta periodicamente a *Bind Database* para atualizar os locais com novos dados ou para introduzir novos locais.

3. FERRAMENTAS DE MONITORIZAÇÃO

Hoje em dia, as pessoas são cada vez mais dependentes das novas tecnologias. Com o aumento do número de aplicações e serviços, é exigida mais qualidade de serviço, mais requisitos de desempenho e, assim, a criação de mais e melhores funcionalidades. Contudo devemos pensar nos riscos que possam surgir, nas falhas ou anomalias inesperadas. De forma a corrigir estas falhas ou mesmo evitá-las é necessário monitorizar o sistema para garantir a máxima disponibilidade e também melhorar a qualidade do serviço. Neste caso, a disponibilidade é fundamental para garantir que os utilizadores tenham acesso a todas as funcionalidades da aplicação e não sejam constantemente impedidos de a utilizar por esta não estar disponível, ou com falhas.

A seguir serão apresentadas algumas ferramentas de monitorização identificadas durante o levantamento de ferramentas de monitorização.

3.1. OPENNMS

OpenNMS é uma ferramenta para gestão e monitorização de redes *open-source*. Este projeto foi iniciado em julho de 1999.

O OpenNMS tem várias funcionalidades. Dentro destas está a capacidade de monitorização de serviços, incluindo a realização regular de testes a todos os tipos de serviços que estão a ser prestados na rede. Durante a monitorização, verifica o estado dos serviços e caso haja anomalias são tomadas as devidas ações. Uma outra funcionalidade é a capacidade de criação de eventos e escolha de ações a tomar caso estes ocorram, os eventos podem gerar notificações através de métodos de *e-mail*. O OpenNMS é capaz de medir o desempenho através da recolha dos dados dos equipamentos da rede e dos seus serviços. O OpenNMS gera relatórios estatísticos em forma de gráficos de todos os dados recolhidos. Possibilita também a criação de tarefas para a adição automática de novos serviços, *hosts*, etc. Os ficheiros de configuração são escritos em linguagem XML (*eXtensible Markup Language*) (OpenNMS, 2014).

3.2. EUROTUX

O eurotux é uma ferramenta própria de monitorização, com base em tecnologias *open-source*, capaz de vigiar infraestruturas de rede, plataformas, serviços e aplicações.

Esta ferramenta é capaz de gerar relatórios de disponibilidade, acompanhando a evolução do desempenho de serviços e lançando alertas (por *e-mail*, SMS, etc.) que avisam para falhas nos sistemas.

Esta ferramenta é adaptável às necessidades específicas da empresa onde é instalado, apresenta uma visão detalhada dos recursos que estão a ser utilizados pela infraestrutura, e é capaz de detetar potenciais falhas ainda numa fase inicial, fazendo com que os tempos de resposta e os danos causados por essas falhas sejam reduzidos. (Eurotux Informática, 2013).

A ferramenta eurotux é capaz de monitorizar as infraestruturas de rede, incluindo supervisionar a disponibilidade, traçar gráficos de utilização recolhendo dados por SNMP (*Simple Network Management Protocol*), portas de routers/switches, interfaces de servidores, etc.

Quanto à monitorização de plataformas é capaz de monitorizar:

- Disponibilidade;
- CPU;
- Estatísticas de I/O;
- Utilização de memória;
- Espaço nas partições;
- Estado das ventoinhas, fontes de alimentação, temperatura;
- Utilização interfaces rede;
- Alertas para telemóveis, via SMS;
- SWAP;
- Monitorização em qualquer lugar através de aplicações móveis para Android e iOS.

O eurotux também é apto a monitorizar a disponibilidade de serviços como:

- HTTP, FTP, DNS;
- Base de dados MySQL, Oracle, SQL Server, PgSQL, etc;
- TomCat, JBoss, etc;

Além de todos estes benefícios o eurotux não exige a compra de nenhuma licença de software comercial (Eurotux, 2013).

3.3. ZABBIX

Zabbix é um software capaz de monitorizar a disponibilidade e o desempenho de servidores, aplicações web, base de dados, equipamentos de rede, entre outras. É *open-source* e não tem

custo. É capaz de monitorizar em tempo real dezenas de milhares de servidores, máquinas virtuais e dispositivos de rede, reunindo estatísticas precisas e dados de desempenho.

Com o Zabbix é possível verificar a disponibilidade e capacidade de resposta dos serviços padrão, tais como servidores de *e-mail* ou web, sem precisar instalar nenhum software nos dispositivos monitorizados. É também capaz de verificar se as bases de dados estão disponíveis e ainda monitoriza-las. É o caso do MySQL, PostgreSQL, Oracle e Microsoft SQL Server (Zabbix, 2011a).

Zabbix é também capaz de mandar mensagens via *e-mail*, SMS ou Jabber (protocolo XMPP). Além disso, tem a possibilidade de criar ações automáticas que podem corrigir problemas, como reiniciar um serviço, ou fazer um servidor de reposição através de IPMI (Zabbix, 2011b).

3.4. NAGIOS

Nagios é uma ferramenta de monitorização que permite identificar e resolver problemas de infraestrutura TI, fazendo com que os danos causados por esses problemas sejam reduzidos. É também capaz, de atenuar problemas futuros, e assim evitar que afetem os utilizadores finais (Nagios, 2014).

Nagios fornece (P. Pinto, 2010):

- Monitorização de servidores (Windows/*Unix), serviços (SMTP, POP3, HTTP, NNTP, PING, etc) e equipamentos ativos da rede;
- Monitorização de recursos (carga, disco, etc);
- Possibilidade de desenvolvimento, de *plugins* específicos capazes de dar resposta às necessidades de cada administrador. Estes *plugins* podem ser desenvolvidos nas mais diversas linguagens de programação/scripting como bash, PHP (*Hypertext Preprocessor*), Python ou Perl;
- Escalável, uma vez que os processos de monitorização são executados de forma paralela;
- Notificação (*e-mail*, SMS, etc.) quando é detetado um problema e para quando o mesmo é considerado resolvido;
- Capacidade de executar ações de resolução sobre os problemas quando estes são detetados pelo Nagios;
- Interface Web;
- Vários níveis de estado dos serviços monitorizados:

- UNKNOWN: Quando não é possível obter informação sobre o elemento monitorizado;
- OK: Quando o elemento monitorizado está a funcionar sem problemas;
- WARNING: Quando o elemento monitorizado está acima do valor definido para *Warning*;
- CRITICAL: Quando o elemento monitorizado está acima do valor definido para *Critical*, ou deixou de responder aos pedidos do servidor Nagios.

3.5. CONCLUSÃO

São inúmeras as soluções que resolvem parcialmente o problema da monitorização. Depois da análise apresentada fica claro que não há uma solução ideal, que se destaque de todas. Fica a cargo de cada administrador da rede escolher a solução que melhor se adapte às necessidades específicas deste.

Para verificar a disponibilidade do nosso servidor o Nagios pareceu ser o mais indicado, pois é uma ferramenta que reúne um maior nível de aceitação pelos utilizadores, é bem documentada que faz com que a sua instalação não seja difícil, possibilita o desenvolvimento de *plugins* específicos e envia notificações sempre que é detetado problemas e quando estes são resolvidos.

4. SISTEMA DE POSICIONAMENTO EXISTENTE

4.1. DESCRIÇÃO DO SISTEMA EXISTENTE

Como referido anteriormente, para o desenvolvimento deste projeto já dispomos de uma aplicação móvel Android e um sistema central de processamento e armazenamento de dados. Com esta aplicação, o utilizador pode registar-se, obter a sua localização, fazer pedidos de amizade a utilizadores, para que estes pertençam ao seu grupo de amigos, e obter informação relativa aos amigos, nomeadamente a sua localização mais recente. Relativamente à localização dos utilizadores, a aplicação disponibiliza os locais dos campi da Universidade do Minho ao utilizador e também os locais próximos ao utilizador quando este se encontra fora da área universitária, permitindo na mesma que o utilizador adicione novos locais ao sistema sempre que seja necessário, pertencendo ou não à Universidade do Minho.

O sistema inclui duas bases de dados, uma no dispositivo móvel e outra no servidor do sistema. A base de dados do dispositivo móvel armazena temporariamente os dados recolhidos pela interface Wi-Fi do dispositivo, que não são enviados para o servidor por falta de conectividade à Internet, até que o envio destes seja bem-sucedido. A base de dados do servidor do sistema guarda os dados que dão suporte à aplicação móvel e ao motor de posicionamento, ou seja, armazena os dados relativos aos utilizadores, o mapa de rádio do motor de posicionamento, os locais previamente inseridos e os locais que os utilizadores adicionam à medida que utilizam a aplicação. Com esta aplicação foi permitido envolver os utilizadores, sem que disso se apercebam, na construção dos mapas de rádio, muito embora esse objetivo seja claramente apresentado na descrição pública do sistema.

4.2. ARQUITETURA GERAL DO SISTEMA

Neste sistema o principal objetivo é a construção do mapa de rádio através da recolha da informação dos pontos de acesso Wi-Fi. Para esta recolha de informação é usado um serviço da aplicação móvel, que através da interface Wi-Fi dos dispositivos móveis, procede à recolha da informação dos pontos de acesso, que estão na vizinhança do dispositivo.

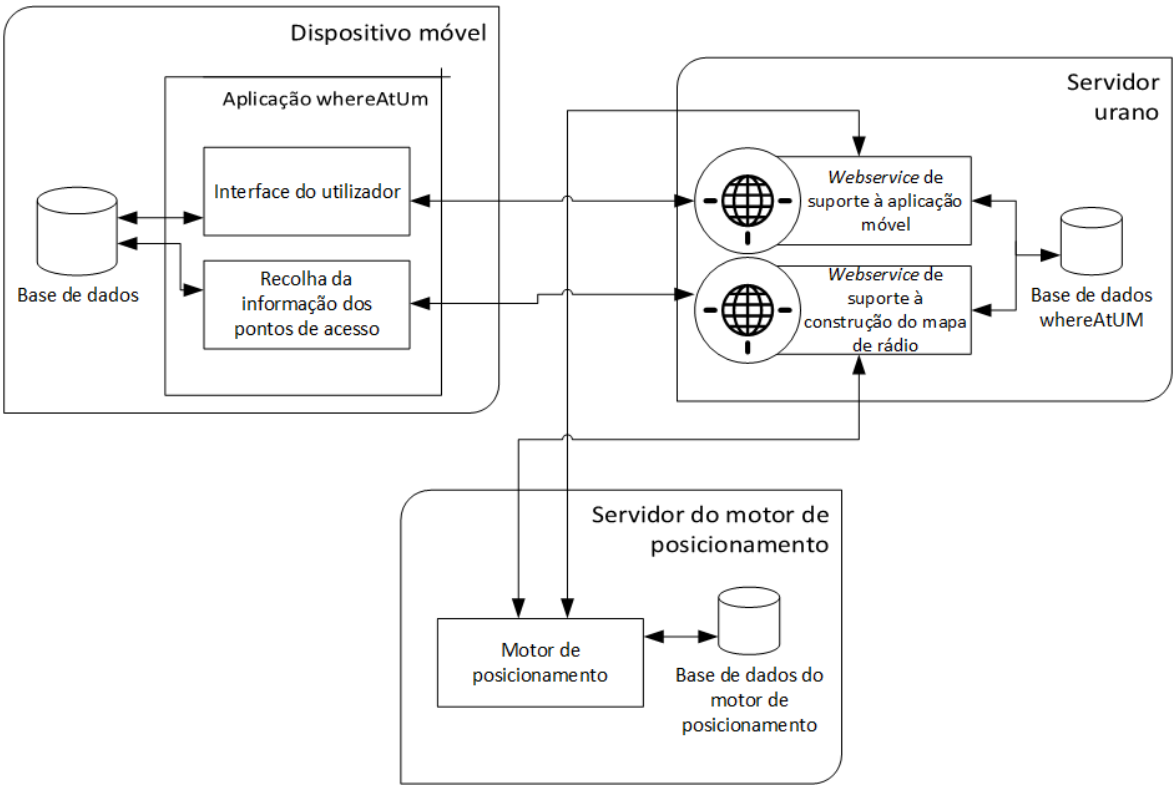


Ilustração 3 - Arquitetura geral do sistema existente (Matos, 2014).

Na Ilustração 3 está representada a arquitetura do sistema existente. Pode ser verificado que a aplicação móvel é capaz oferecer as funcionalidades necessárias ao utilizador e também recolher os dados dos pontos de acesso Wi-Fi através da interface Wi-Fi do dispositivo móvel. Compete ao *web service* de suporte à aplicação móvel ser capaz de disponibilizar todas as informações da base de dados para que a aplicação funcione corretamente. O *web service* de suporte à construção do mapa de rádio é responsável por receber e processar os novos locais adicionados por utilizadores e as *fingerprints* recolhidas pelos dispositivos móveis. O servidor do motor de posicionamento contém outro mapa de rádio (base de dados do motor de posicionamento) e ajuda a responder a pedidos de localização sempre que o mapa de rádio da base de dados "whereatum" não consegue estimar a localização.

4.3. DESCRIÇÃO DA ABORDAGEM ATUAL DO PROCESSO DE POSICIONAMENTO

O processo de localização do utilizador é realizado sempre que o servidor recebe uma *fingerprint* enviada pela aplicação móvel. Através de um algoritmo de estimação, a *fingerprint* recebida é comparada com as *fingerprints* anotadas na base de dados e por fim é obtida a

localização do utilizador através de um processo de *ranking*. Nesta secção é descrito detalhadamente todo esse processo.

Função "fingerprint"

Quando o dispositivo recolhe uma nova *fingerprint*, esta *fingerprint* recebida será comparada com as já armazenadas, para que a aplicação seja capaz de estimar a localização do utilizador.

Esta função começa por obter o identificador do utilizador do dispositivo. De seguida é obtida a última localização conhecida do utilizador. Caso a *fingerprint* recebida possua uma data mais antiga que a última localização conhecida do respetivo utilizador[6], esta é comparada com as *fingerprints* anotadas de maneira a tentar associar a *fingerprint* recebida com um dos locais conhecidos, sendo posteriormente armazenada na tabela "fingerprintsHistory". Se a *fingerprint* recebida for mais recente que a última localização conhecida do utilizador, e se o intervalo de tempo entre as duas for inferior a 1 minuto, o sistema assume que o utilizador se encontra no mesmo local da última localização conhecida do utilizador. Se existir uma última localização conhecida do utilizador mas o intervalo de tempo entre esta e a *fingerprint* recebida for superior a 1 minuto, a *fingerprint* recebida será comparada com as *fingerprints* anotadas para se tentar estimar uma localização, para isso é usada a Função "fingerprintsRankingV2", que será explicada mais tarde. Caso a Função "fingerprintsRankingV2" retorne uma resposta positiva, ou seja uma localização válida, a última localização conhecida do utilizador é alterada para a localização estimada, e o atributo "isCurrent" da tabela "lastKnownLocation" é colocado a 1 (este atributo estando a 1 indica que é a última localização conhecida do utilizador). Esta informação é guardada na base de dados, na tabela "fingerprintsHistory" (para guardar na base de dados é usada a Função "saveFingerprintInDB"). Caso a *fingerprint* não corresponda a nenhum local conhecido, a última localização conhecida do utilizador é atualizada para desconhecida colocando o atributo "isCurrent a 0, a *fingerprint* é guardada na base de dados e posteriormente enviada para o motor

[6] Este caso acontece pois sempre que um dispositivo móvel recolhe *fingerprints*, por vezes, por algum motivo este não consegue enviar de imediato ao servidor, então estas *fingerprints* são guardadas numa base de dados da aplicação móvel para mais tarde, assim que seja possível, esta enviar os dados recolhidos ao servidor.

de posicionamento (através da função "httpPOST"), para assim que possível o motor de posicionamento verificar se conhece estas localizações.

Função "isCurrent"

Esta função recebe como atributos o *idUser* e o *timestamp* e verifica na base de dados, na tabela "lastKnownLocation" se a *fingerprint* recebida é mais antiga que as *fingerprint* da última localização conhecida do utilizado.

Função "saveFingerprintInDB"

Esta função serve para guardar uma *fingerprint* de um dispositivo específico na base de dados. Recebe como parâmetros o *idUser*, *device*, *timestamp*, *aps*, *idPlaces* e *executionTime*. Começa por inserir a nova *fingerprint* na base de dados na tabela "fingerprintsHistory" e depois insere os APs na tabela "apsFingerprintsHistory" associados às respetivas *fingerprints*.

Função "placessearch"

Esta função obtém os locais próximos ao utilizador, fora da Universidade do Minho. Recebe como atributos as coordenadas da posição do utilizador. Começa por obter todos os lugares da base de dados, onde o atributo "idType" é igual a 8 e as coordenadas não estejam a vazio. De seguida calcula a distância entre coordenadas (onde recorre à Função "vincentyGreatCircleDistance") e são obtidos os locais com distância menor a 150 (Matos, 2014).
Feito isto obtém os locais de Foursquare usando as coordenadas GPS e funde-se com os resultados da base de dados, excluindo os duplicados.

Função "places"

É nesta função que é feito todo o processo de introdução dos locais inseridos pelos utilizadores na aplicação. Recebe como parâmetros a *fingerprint* e a localização. Começa por verificar se os locais existem, caso existam faz uma atualização, caso não existam insere os novos locais. Por fim, insere a *fingerprint* e os APs.

Também é nesta função que são obtidos os locais pertencentes à Universidade do Minho.

Função "calculateFingerprintsDistance"

Esta função recebe como parâmetros duas *fingerprints* e calcula a distância entre elas para se saber se as duas *fingerprints* pertencem ao mesmo local. A distância é definida pela soma da diferença absoluta entre os valores RSSI de cada ponto de acesso, dividida pelo número total de pontos de acesso detetados.

É necessário fazer este cálculo, pois a maioria das vezes quando são recolhidas duas *fingerprints* no mesmo local, as *fingerprints* apresentam valores RSSI diferentes para os mesmos pontos de acesso. Por vezes acontece mesmo um dispositivo móvel detetar um ponto de acesso e outro estando no mesmo local não o conseguir detetar. Isto acontece devido a vários fatores como, as pessoas estarem em movimento, obstáculos entre o dispositivo e o ponto de acesso, dispositivos móveis com diferentes características a nível da placa de rede Wi-Fi., entre outros (Matos, 2014).

No caso do sistema tiver que comparar duas *fingerprints* por exemplo, registadas por dispositivos móveis diferentes, e um deles tiver detetado um ponto de acesso que o outro não tenha detetado, o sistema assume um valor RSSI pré-definido de -109 (Matos, 2014), e usa esse valor para conseguir calcular a diferença entre o valor RSSI presente na outra *fingerprint*, obtendo assim a distância entre as duas *fingerprints*.

Função "fingerprintsRankingV2"

Esta função recebe como parâmetro uma *fingerprint* e é feita a comparação desta *fingerprint* recebida com as *fingerprints* anotadas para estimar a localização do utilizador.

As *fingerprints* anotadas são escolhidas por ordem de data mais recentes e são apenas selecionadas 10 *fingerprints* anotadas por local. Após a seleção é calculada a distância entre as *fingerprints* (através da Função "calculateFingerprintsDistance"). Posteriormente é ordenada uma lista de *fingerprints* por distância e verifica se a primeira está abaixo do valor *threshold*. O valor de *threshold* é pré-definido a 17.5, este valor define a distância limite entre duas *fingerprints*. Isto é, se a distância calculada entre as duas *fingerprints* for inferior a 17.5, o sistema concluí que as duas *fingerprints* pertencem ao mesmo local e é retornado o identificador desse mesmo local.

Função "vincentyGreatCircleDistance"

Nesta função é calculada a distância entre as coordenadas dos locais da base de dados e as coordenadas do local do utilizador. Recebe como parâmetros os dois pares de coordenadas.

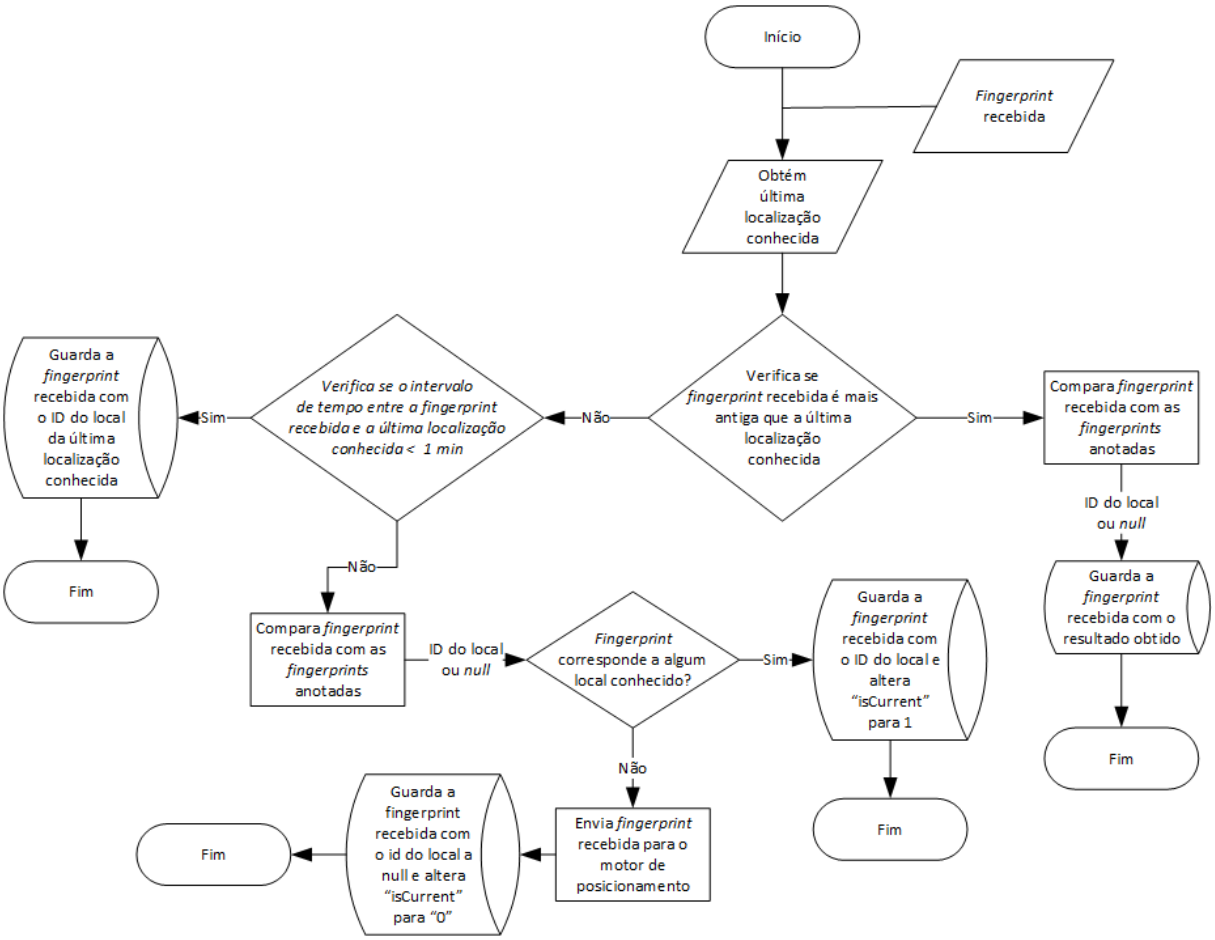


Ilustração 4 - Processo existente para localização dos utilizadores.

Na Ilustração 4 pode ser visto um esquema que apresenta todo o processo existente para localização dos utilizadores, explicado atrás na função "*fingerprint*".

4.4. CRIAÇÃO E MANUTENÇÃO DO MAPA DE RÁDIO

Um dos objetivos deste projeto é a criação e manutenção do mapa de rádio para grandes edifícios, ou conjuntos de edifícios. Esta é uma tarefa morosa e que requer um grande esforço, no entanto com a execução desta tarefa estima-se que os resultados da localização *indoor* se tornem mais precisos. Para facilitar a resolução deste problema é necessário que a aplicação móvel seja utilizada por um elevado número de utilizadores. À medida que os utilizadores usam a aplicação

encontram locais conhecidos por ela, mas também desconhecidos. Quando a aplicação exibe um local desconhecido, o utilizador pode colaborar para o crescimento do mapa de rádio, introduzindo num formulário da aplicação informação sobre o local onde se encontra. À informação do local dada pelo utilizador é associada uma *fingerprint* recolhida no mesmo instante. Por fim, estes dados recolhidos são enviados para o servidor do sistema que processa e armazena o novo local na base de dados.

4.5. DESCRIÇÃO DO PROBLEMA

Segundo Pendão (2012) surgem vários desafios na utilização de sistemas colaborativos, pois são muito dependentes dos utilizadores. Com isto, também se torna muito complicado confirmar a veracidade nos dados inseridos pelos utilizadores. No entanto, assomam outros desafios como segurança, privacidade e angariação de novos utilizadores. Por isso será também importante resolver estes problemas para que os utilizadores não percam o interesse na aplicação.

Como já referido anteriormente, o utilizador tem um papel fulcral neste sistema, logo quanto mais utilizadores utilizarem a aplicação mais depressa o mapa de rádio do sistema crescerá. Para isso é preciso soluções para manter e angariar novos utilizadores que usem a aplicação, com alguma regularidade, e ajudem com o seu contributo.

Quanto à integridade dos dados inseridos pelos utilizadores, estes podem ser adulterados por pessoas maliciosas ou, então, os utilizadores introduzirem dados falsos ou incorretos, colocando desta forma em risco a credibilidade do nosso sistema de posicionamento. Logo é preciso encontrar uma solução que nos ajude a verificar a credibilidade dos dados inseridos pelos utilizadores.

O objetivo desta dissertação é conseguir resolver os problemas anteriormente apresentados.

5. PROPOSTAS DE MELHORIA DO SISTEMA

5.1. ESTRATÉGIA PARA FOMENTAR A INTERAÇÃO COM A APLICAÇÃO

Para o nosso sistema ter sucesso, é necessário que a aplicação móvel seja utilizada por um elevado número de utilizadores. Surge aqui o desafio de manter os utilizadores interessados em não deixar de utilizar a aplicação, e também angariar novos utilizadores. Criação de novidades como novas funcionalidades pode captar a atenção e o interesse dos utilizadores experimentarem, gostarem e passarem a ser utilizadores assíduos da aplicação.

5.1.1. FUNCIONALIDADE DE ENVIO E RECEÇÃO DE MENSAGENS

Segundo um estudo da Deloitte (2014), as pessoas optam cada vez mais por serviços de mensagens instantâneas para comunicarem entre si. Por isso, surgiu a ideia de se integrar um serviço de mensagens instantâneas na aplicação já existente. Os utilizadores já tendo acesso à localização dos seus amigos a partir da aplicação, faria todo o sentido que um utilizador ao se aperceber que um amigo está por perto, este pudesse contactá-lo para um possível encontro, etc. Ou até mesmo os utilizadores passarem a utilizar a aplicação simplesmente para comunicarem com os amigos.

Para implementação desta funcionalidade pretende-se utilizar o serviço gratuito GCM (*Google Cloud Messaging*). A escolha recaiu sobre este serviço por este apresentar grandes vantagens, como tratar de todos os aspetos relacionados com a fila de mensagens para envio e entrega de mensagens ao destinatário certo, ser grátis e a sua implementação ser fácil. Estes pontos contribuíram para a escolha deste servidor como mecanismo a implementar na aplicação desenvolvida.

5.1.1.1. Google Cloud Messaging

O GCM é um serviço gratuito fornecido pela Google, que permite o envio e receção de dados entre dispositivos Android (Android, 2015). O servidor GCM envia mensagens a um dispositivo Android, para que este não esteja constantemente a consultar o servidor com o intuito de verificar se há novas mensagens.

Uma implementação do mecanismo de notificação *push*, fornecido pelo GCM, inclui um servidor de conexão GCM, uma aplicação servidor que interage com o servidor de conexão e uma

aplicação cliente. Na Ilustração 5 está apresentada a arquitetura GCM e como todos os componentes envolvidos interagem.

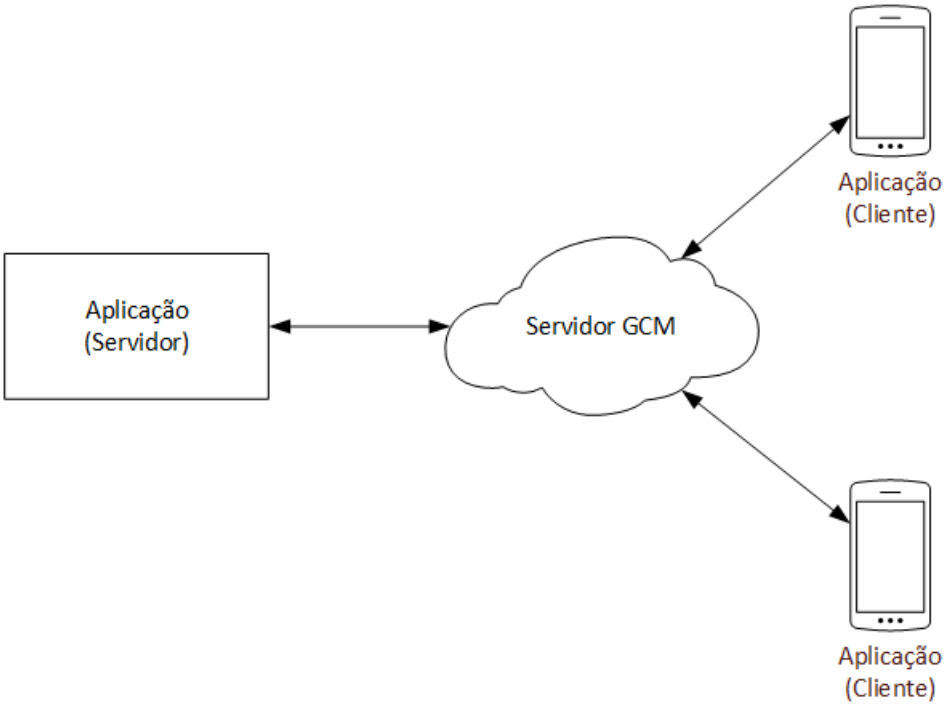


Ilustração 5 - Componentes GCM (Queirós, 2014).

O papel do componente da aplicação do servidor é enviar mensagens para o servidor de conexão GCM.

O servidor de conexão GCM é o componente que envia as mensagens recebidas da aplicação do servidor para as aplicações cliente registadas.

A aplicação cliente tem que estar registada no servidor GCM e assim obter um ID de registo, que é partilhado com a aplicação do servidor, só assim a aplicação cliente recebe as mensagens através do servidor GCM (Queirós, 2014).

Em suma, sempre que a aplicação do servidor tivesse que enviar uma mensagem a um ou vários dispositivos, esta enviará a mensagem juntamente com o ID do registo GCM dos dispositivos destinatários ao servidor GCM. O servidor GCM ao receber os dados encarregava-se de enviar as mensagens, através de uma notificação *push,* que apareceria nos dispositivos móveis destinatários.

5.1.2. CRIAÇÃO DE ALERTAS PARA QUESTIONAR OS UTILIZADORES

Para induzir a interação dos utilizadores com a aplicação pretende-se que os utilizadores sejam atraídos. Assim, foi pensada uma forma de questionarmos os utilizadores acerca da fiabilidade das localizações que a aplicação lhes transmite. Os utilizadores são fundamentais para o sucesso deste projeto, além de se envolverem no sistema e colaborarem na construção do mapa de rádio, agora com a implementação destas notificações vão poder colaborar na correção deste.

Assim é pretendido que os utilizadores deem o seu *feedback* acerca da localização mencionada pela aplicação, como também incentivar o utilizador a definir novas localizações. Desta forma, os utilizadores ao darem *feedback* às questões da aplicação estarão a contribuir para a recolha de novas *fingerprints* e corrigir possíveis *fingerprints* erradas. Esta será mais uma maneira de fazer o sistema crescer. Contudo a implementação de alertas exige um certo cuidado, pois se por um lado cativa a interação do utilizador, por outro o recurso excessivo aos mesmos pode tornar-se incomodativo, o que conduziria ao surgimento de resistência por parte dos utilizadores, o que deve ser levada em conta.

5.2. ESTRATÉGIA PARA AVALIAR A QUALIDADE E CREDIBILIDADE DOS DADOS FORNECIDOS PELOS UTILIZADORES NA APLICAÇÃO MÓVEL

O nosso sistema, como já referido anteriormente, é um sistema colaborativo, ou seja, que permite os utilizadores inserirem informação através da aplicação móvel neste caso o nome do local onde se encontram. Uma das principais preocupações neste tipo de sistemas é avaliar a qualidade e credibilidade dos dados inseridos pelos utilizadores. Estes podem ser mal-intencionados ou até mesmo negligentes contribuindo com informações erradas, prejudicando a qualidade e credibilidade do nosso mapa rádio.

Assim, a solução poderia passar pela utilização de um mecanismo de reputação para tentar impedir que as informações erradas introduzidas pelos utilizadores prejudiquem o nosso sistema de localização dos utilizadores em ambientes interiores.

Segundo Rebocho (2013), os sistemas colaborativos que têm como objetivo impedir o mau comportamento dos utilizadores podem utilizar os mecanismos de reputação. Os mecanismos de reputação têm como finalidade recolher, partilhar e guardar as opiniões e experiências dos utilizadores sobre empresas, produtos, serviços, outros utilizadores, etc. Desta forma é possível decidir quais os utilizadores em que se podem confiar e tentar evitar que os menos confiáveis

influenciem prejudicialmente o sistema, reduzindo as suas reputações para que a sua opinião não tenha tanto impacto.

No entanto os mecanismos de reputação têm várias vulnerabilidades sendo que os torna alvos fáceis a ataques, como classificações falsas, dificuldade para obter *feedback* negativo, e ausência de incentivo aos utilizadores para que eles contribuam com as suas classificações (Josang & Golbeck, 2009).

Para o nosso sistema o objetivo não é que os utilizadores saibam em quem possam confiar, mas sim fortalecer a qualidade dos dados inseridos por eles, para decidir quais as *fingerprints* mais confiáveis.

Como já referido anteriormente, o sistema de posicionamento apresentado por Mahtab Hossain et al. (2010), utiliza o *feedback* dos utilizadores, este é obtido quando os utilizadores indicam a sua posição ao sistema. Os autores defendem que o *feedback* dos utilizadores é importante, pois ajuda a melhorar o desempenho do sistema de posicionamento. No entanto têm noção que há utilizadores mal-intencionados e que muitas vezes têm como objetivo sabotar o sistema de posicionamento. Estes utilizadores são considerados prejudiciais para o sistema, pois contribuem com informações incorretas que em vez de ajudar a melhorar a precisão do sistema de posicionamento poderiam fazer com que o erro de posicionamento fosse ainda maior. Assim apresentam uma forma para filtrar estes *feedbacks* duvidosos, que consiste em sempre que um utilizador afirmar que está num determinado local, esta informação é associada a um grau de credibilidade. Para calcular este fator de credibilidade foram considerados *n feedbacks* de utilizadores "bem comportados" e utilizados como amostras para obter um "*RoC*[7] *profile*". A partir deste perfil o sistema consegue filtrar quais os *feedbacks* prejudiciais para o sistema (Mahtab Hossain et al., 2010).

Assim, para garantir a credibilidade dos dados inseridos pelos utilizadores, decidiu-se que seria necessária novamente a opinião de todos os utilizadores. Com a criação da nova funcionalidade na aplicação móvel (secção 5.1.2), onde o utilizador é questionado se concorda, ou não, com o local que a aplicação diz este estar, será possível obter o *feedback* dos utilizadores sobre as suas localizações. Esta funcionalidade não só incentivará o utilizador a interagir mais com a aplicação mas também ajudará a recolher o *feedback* dos utilizadores podendo assim,

[7] *Region of Confidence*

posteriormente, ser decidido quais as *fingerprints* que os utilizadores concordam mais e vice-versa, ou seja quais as mais e menos credíveis.

Por fim, o sistema de posicionamento não terá em conta as *fingerprints* que contenham um grande número de *feedback* negativo, o que fará com que o sistema seja mais preciso na estimação da localização dos seus utilizadores.

6. IMPLEMENTAÇÃO DAS MELHORIAS NO SISTEMA

Neste capítulo é apresentada toda a implementação das melhorias propostas no capítulo.

6.1. APLICAÇÃO MÓVEL

6.1.1. FERRAMENTAS, PLATAFORMA DE DESENVOLVIMENTO

Para proceder à implementação das novas funcionalidades na aplicação móvel foi necessária a instalação de algum *software*. Assim, procedeu-se à instalação da ferramenta Eclipse ADT (*Android Developer Tools*) para auxiliar no desenvolvimento da aplicação e do pacote de ferramentas Android SDK (*Software Development Kit*) que auxilia a desenvolver aptidões para a plataforma Android.

6.1.2. COMPONENTES DA APLICAÇÃO MÓVEL

Como já referido na secção 4.2, a aplicação móvel é capaz de interagir com o utilizador e de recolher dados dos pontos de acesso Wi-Fi através da interface Wi-Fi do dispositivo móvel. A aplicação móvel tem como componentes *Activities* e *Fragments* e *Services*.

As *Activities* são atividades ou ações da aplicação móvel. Todas as aplicações móveis Android têm uma ou mais *Activities* que normalmente interagem com os utilizadores através de uma interface do utilizador.

Os *Fragments* servem para representar uma parte da interface do utilizador em particular. Estes são executados dentro de uma atividade principal e cada fragmento pode ter um *layout*[8] específico.

Os *Services* são geralmente utilizados para realizar tarefas em *background*, podem ter a sua execução agendada e não dependem de nenhuma ação do utilizador.

A seguir são apresentadas as principais atividades, fragmentos e serviços que compõem a aplicação móvel:

- Register - Esta atividade é responsável por permitir que o utilizador se registe na aplicação.
- Login - Esta atividade é responsável por permitir que o utilizador entre na sua conta e tenha acesso às suas informações.

[8] Controla o formato do ecrã da aplicação móvel e a aparência da imagem da interface do utilizador.

- MainActivity - É a atividade principal, e é sobre esta atividade que funcionam os vários fragmentos da aplicação.
- MainScreen - Este fragmento é o responsável pela apresentação das informações listadas no menu "*Home*" da aplicação, como a localização do utilizador e as localizações dos amigos.
- FriendsScreen - Fragmento responsável por listar todos os amigos do utilizador, fazer pedidos de amizade, ver pedidos de amizade enviados e recebidos.
- UserSettings - É a atividade responsável por permitir que os utilizadores modifiquem as configurações da aplicação.
- AlarmHelper - Classe responsável por programar um alarme que execute periodicamente o serviço de recolha de dados.
- WifiScan - Este é um serviço iniciado pela classe "*AlarmHelper*", responsável por iniciar e receber a recolha de dados dos pontos de acesso Wi-Fi detetados na vizinhança do dispositivo. Após receção dos dados, o serviço envia-os para o servidor; se não houver ligação à Internet, o serviço armazena os dados recebidos na base de dados SQLite do dispositivo móvel para envio posterior.
- ChatActivity - Esta é a atividade responsável por permitir que o utilizador envie mensagens e mostra o histórico das mensagens recebidas e enviadas.
- BroadcastReceiver - É responsável por receber os dados recolhidos dos pontos de acesso Wi-Fi, receber as mensagens enviadas por outros utilizadores e receber as notificações enviadas pelo servidor, pedindo ao utilizador para confirmar o local em que se encontra.

6.1.3. ECRÃS E FUNCIONALIDADES

Nesta secção são apresentados os ecrãs das novas funcionalidades implementadas. Contudo é possível observar todos os ecrãs existentes na aplicação móvel no Apêndice B (Ilustração 21, Ilustração 22 e Ilustração 23).

6.1.3.1. Ecrã Principal

Quando o utilizador inicia a aplicação, este é redirecionado para o ecrã principal. Neste ecrã o utilizador já pode ver uma lista com os amigos, com a respetiva localização de cada um. Os amigos estão listados por data de atualização da localização mais recente, a intensidade da cor do

círculo verde visível na Ilustração 6 altera-se consoante a data da última atualização da localização do utilizador. O utilizador também consegue observar a sua própria localização e tem um menu onde consegue aceder à lista de amigos e pedidos de amizade (enviados e recebidos), no submenu "*Friends*", entre outras funcionalidades. Todas estas funcionalidades da antiga versão da aplicação são apresentadas em Matos (2014). Nesta nova versão da aplicação móvel é possível enviar mensagens através do *icon* de *chat* apresentado na ilustração a seguir.

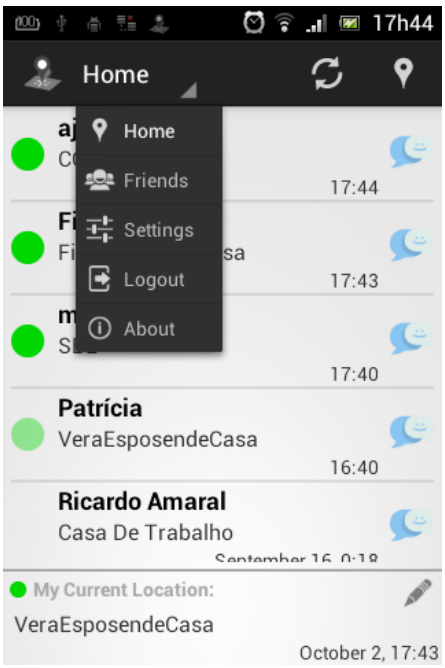


Ilustração 6 - Ecrã Principal.

6.1.3.2. Ecrã de Mensagens

O utilizador para enviar uma mensagem a algum dos seus amigos deve clicar no *icon* de *chat* ao lado do nome do amigo que escolheu enviar uma mensagem. Ao pressionar o *icon* de *chat* o utilizador é levado ao ecrã de mensagens (Ilustração 7). Neste ecrã, o utilizador pode enviar as mensagens ao amigo escolhido no ecrã principal, e ver todas as mensagens recebidas e enviadas.

Ilustração 7 - Ecrã de Mensagens.

O utilizador quando recebe uma mensagem de algum dos seus amigos é notificado, na barra de notificações do dispositivo móvel (Ilustração 8). O utilizador ao pressionar a notificação é redirecionado ao ecrã de mensagens.

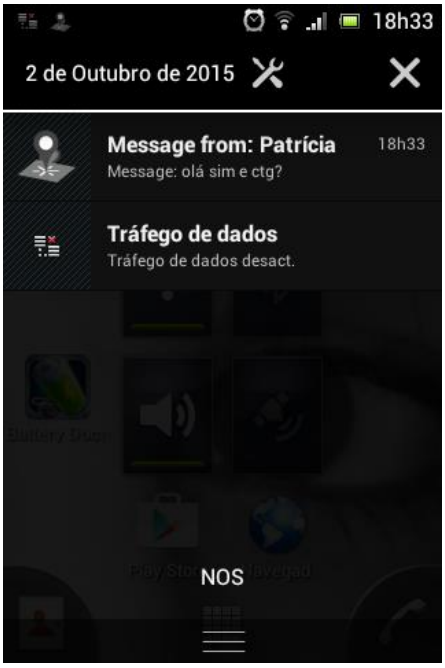


Ilustração 8 - Notificação de nova mensagem recebida.

6.1.3.3. Ecrã de Notificação questionando o utilizador (localização conhecida)

Sempre que o sistema enviar uma notificação ao utilizador questionando se este está no local que a aplicação exibe, a aplicação exibe uma caixa de diálogo, como exemplo temos a Ilustração 9, o utilizador tem três opções de resposta, caso pressione que não está no local exibido pela caixa de diálogo, é levado ao ecrã onde pode inserir a localização que está naquele preciso momento.

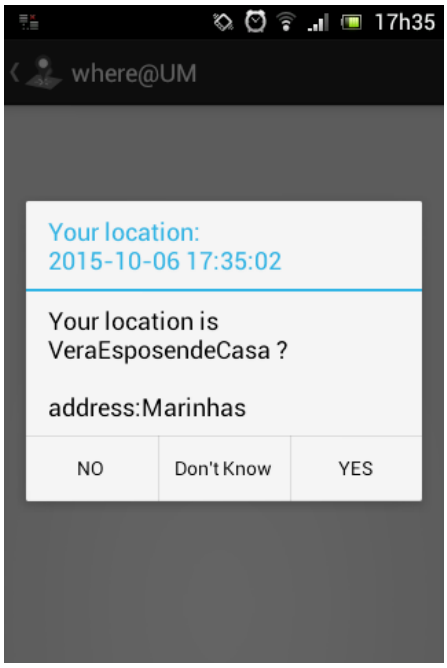


Ilustração 9 - Notificação questionando o utilizador quanto à sua localização conhecida.

6.1.3.4. Ecrã de Notificação questionando o utilizador (localização desconhecida)

Quando o utilizador está num local desconhecido, por vezes a aplicação notifica-o questionando-o se pode inserir a sua localização, temos como exemplo a Ilustração 10. O utilizador tem duas opções de resposta, e caso escolha que quer inserir a sua localização, o utilizado é levado ao ecrã onde pode inserir a sua localização naquele preciso momento.

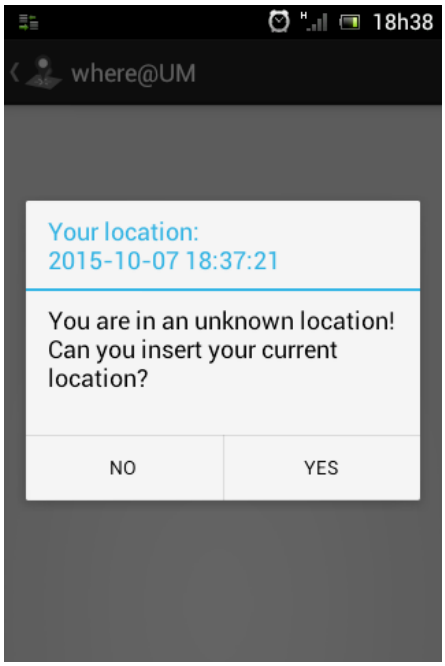


Ilustração 10 - Notificação questionando o utilizador quanto à sua localização desconhecida.

6.1.4. FUNCIONALIDADE DE ENVIO E RECEÇÃO DE MENSAGENS

Nesta secção é explicada toda a implementação da funcionalidade de envio e receção de mensagens. Como explicado na secção 5.1.1, foi usado o serviço GCM do Google. Para isso, na aplicação móvel foram criados novos componentes para suportar este serviço.

6.1.4.1. Registo no GCM

Para que seja possível utilizar o GCM como serviço de mensagens é necessário que os dispositivos móveis estejam registados neste serviço. A aplicação existente já estava a usar este serviço para notificar os utilizadores quando recebem pedidos de amizade de outros utilizadores. Quando um novo utilizador se regista na aplicação Where@UM, a aplicação realiza de imediato o registo do dispositivo móvel no serviço GCM. Assim, para o serviço de envio e receção de mensagens foram reutilizados os identificadores de registo GCM dos utilizadores, guardados na base de dados *"*whereatum*"* na tabela *"*devices*"*.

6.1.4.2. BroadcastReceiver

Foi implementado um *BroadcastReceiver* para receber as mensagens que são enviadas pelos utilizadores amigos, vindas do servidor de conexão GCM, e disponibilizá-las no ecrã de mensagens.

6.1.4.3. SQLite

Para a persistência das mensagens na aplicação móvel, ou seja para que o utilizador tivesse acesso a todas as mensagens recebidas e enviadas de cada utilizador, foi criada uma base de dados SQLite. A base de dados implementada (Ilustração 11) contém uma entidade que permite guardar o nome do utilizador amigo ("person_name"), o identificador de registo GCM ("person_device_id"), a mensagem ("person_chet_mensagem"), se a mensagem foi enviada ou recebida ("to_or_from") e por fim a data que a mensagem foi enviada (timestamp).

Ilustração 11 - Entidade da base de dados que dá suporte ao serviço de mensagens.

Na Ilustração 12 está representado o funcionamento do serviço de envio de mensagens. O utilizador remetente escreve uma mensagem, esta é enviada para o *web service*, este envia a mensagem para o servidor GCM que é responsável por entregar a mensagem ao destinatário correto.

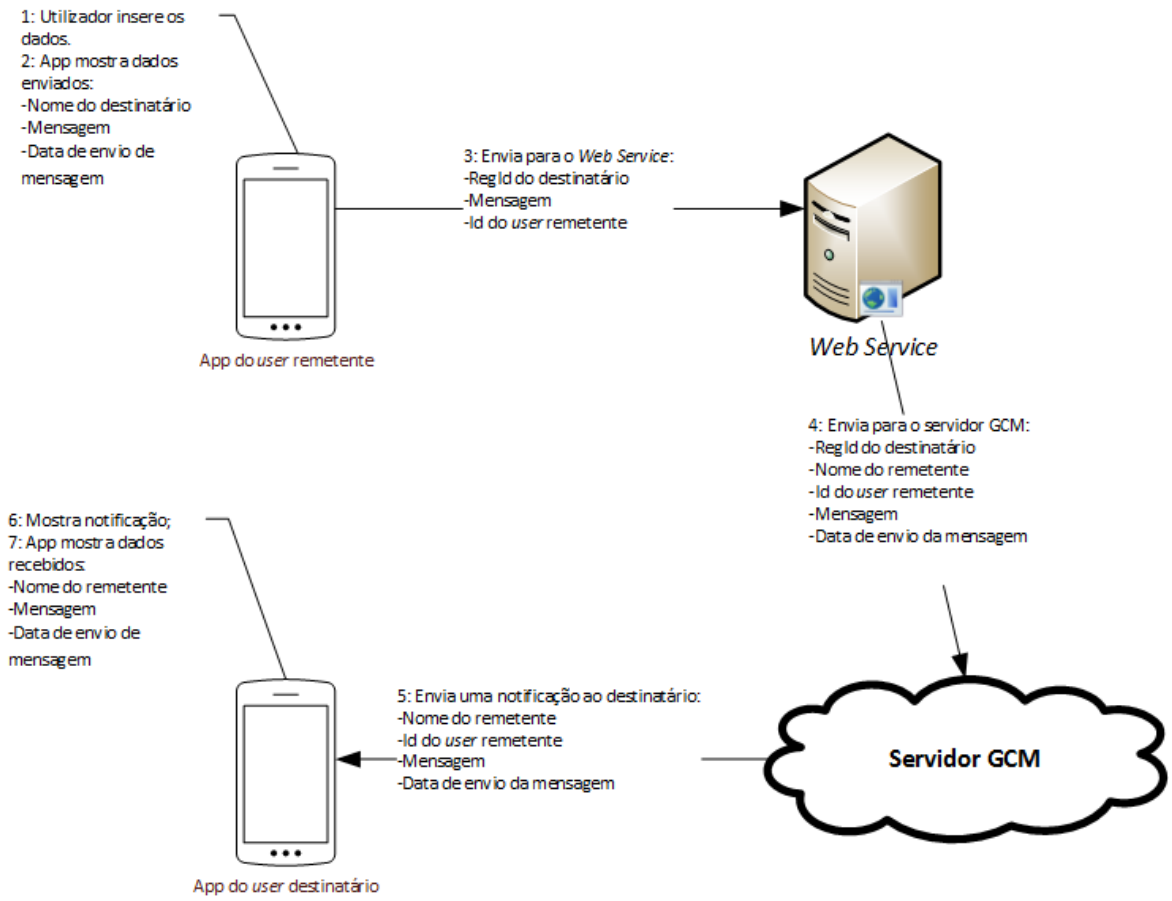


Ilustração 12 - Funcionamento do serviço de envio e receção de mensagens.

6.1.5. IMPLEMENTAÇÃO DOS ALERTAS PARA QUESTIONAR OS UTILIZADORES

6.1.5.1. Notificação para questionar o utilizador sobre a sua localização

Por vezes a localização que a aplicação nos indica não é a mais correta, isto acontece por vários motivos, como questões relacionadas com limitações do *hardware*, ou até mesmo com a possibilidade dos utilizadores se enganarem a inserir a sua localização. No entanto, para salvaguardar esta questão o utilizador tem a possibilidade de corrigir a sua localização manualmente.

Para questionar o utilizador sobre a sua localização, este será notificado através de uma caixa de diálogo com a sua localização. Este terá que verificar se concorda, ou não, com a localização estimada pela aplicação. Esta notificação tem os seguintes atributos:

- Questão sobre se a localização do utilizador coincide com a registada na aplicação.

- Data e Hora.
- Respostas possíveis: “Sim”, “Não” ou “Não Sei”.

Restrições

- O utilizador não deve ser notificado se a sua localização for “desconhecida”.
- Os utilizadores só devem receber este tipo de notificações nos dias úteis das 9 horas às 22 horas e aos fim-de-semanas das 11 horas às 23 horas, para que não se torne um incómodo para o utilizador.
- Estas notificações não devem ser enviadas mais do que uma vez por dia nem mais do que duas vezes por semana, para desta forma não ser demasiado intrusivo.
- Estas respostas devem ser guardadas numa nova tabela “feedbackUsers” da base de dados. Deverá ser guardada a resposta (Sim, Não ou Não sei), o id do utilizador, id da pergunta, hora a que a notificação foi enviada, hora que o utilizador respondeu e id do local que apareceu na questão.
- A caixa de diálogo se não obtiver uma resposta ao fim de 5 minutos deverá desaparecer, pois se o utilizador responder à questão muito mais tarde a sua resposta pode já não ser válida.

Tabela 3 - Cenários possíveis às respostas dadas pelos utilizadores referente à sua localização.

Pergunta	**Data em que o alerta é enviado** **Confirma que está no LID3?**	
Respostas Possíveis		
“Sim”	**“Não”**	**“Não sei”**
Caso o utilizador responda “Sim”, a resposta juntamente com os outros atributos serão guardados na tabela “feedbackUsers” na base de	Caso a resposta do utilizador seja “Não”, este será redirecionado para um ecrã específico, onde poderá inserir a localização onde se encontra (será o mesmo ecrã	Caso o utilizador responda “Não sei”, a resposta juntamente com os outros atributos serão guardados na tabela “feedbackUsers” na

dados.	de quando é inserido um novo local). Neste caso a resposta "Não" juntamente com os outros atributos serão guardados na tabela "feedbackUsers" na base de dados.	base de dados.

Para implementação desta funcionalidade, foi também usado o serviço da GCM, cada vez que a localização do utilizador é atualizada, o algoritmo representado na Ilustração 13 é processado. No final, se todas as verificações forem bem-sucedidas, é enviada a questão ao servidor GCM e este encarrega-se de notificar o utilizador respetivo. Para isto foi necessário criar um *BroadcastReceiver*, para receber os dados vindos do servidor GCM, e mostrar na aplicação móvel.

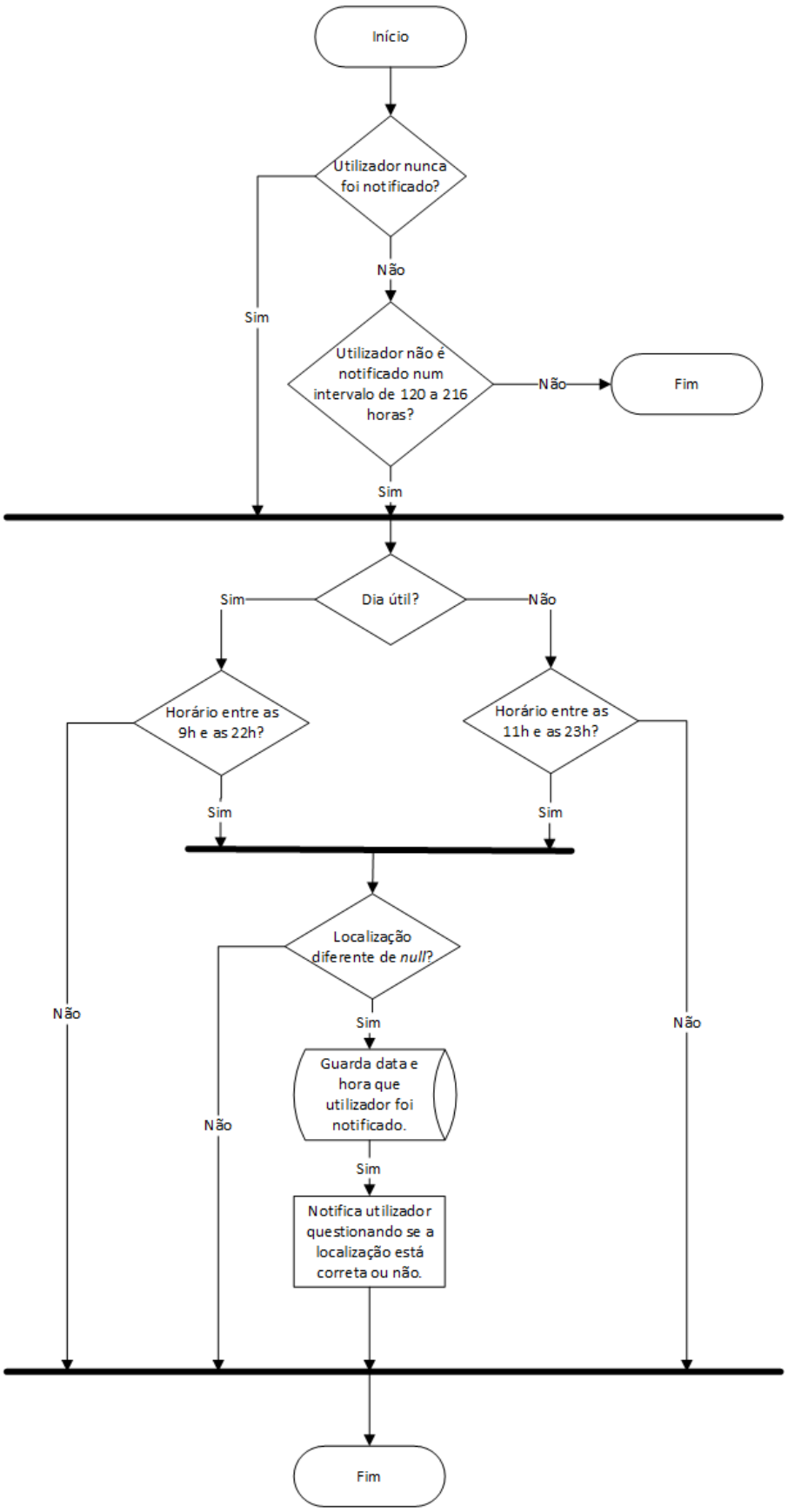


Ilustração 13 - Algoritmo do processo de notificar os utilizadores questionando se a sua localização está correta

6.1.5.2. Notificação aos utilizadores para inserirem um novo local

Quando o sistema não conhece o local em que o utilizador se encontra, a aplicação indica que o utilizador se encontra numa zona desconhecida, no sistema "Unknown".

Neste caso, a aplicação notificará o utilizador, através de uma caixa de diálogo, alertando-o de que está numa localização desconhecida, e incentivando este a inserir a sua localização. Esta notificação tem os seguintes atributos:

- Alerta para uma localização desconhecida e pedido para que atualize a sua localização.
- Data e Hora.
- Respostas possíveis: "Sim" e "Não".

Restrições

- O utilizador só deve ser notificado se a sua localização for desconhecida.
- Os utilizadores só devem receber este tipo de notificações nos dias úteis das 9 horas às 22 horas e aos fim-de-semanas das 11 horas às 23 horas, para que não se torne um incómodo para o utilizador.
- Estas notificações não devem ser enviadas mais do que uma vez por dia nem mais do que duas vezes por semana, para desta forma não ser demasiado intrusivo.
- A caixa de diálogo, se não obtiver uma resposta ao fim de 5 minutos deverá desaparecer, pois se o utilizador responder à notificação muito mais tarde a sua resposta pode já não ser válida.

Tabela 4 - Cenários possíveis às respostas dadas pelos utilizadores referente à sua localização desconhecida.

Pergunta	Data em que o alerta é enviado Encontra-se num local desconhecido, deseja inserir um novo local?
Respostas Possíveis	
"Sim"	"Não"

Caso o utilizador responda “Sim”, este será redirecionado para um ecrã específico, onde poderá inserir a localização onde se encontra (será o mesmo ecrã de quando é inserido um novo local). Neste caso a resposta “Não” juntamente com os outros atributos serão guardados na tabela “feedbackUsers” na base de dados.	Caso o utilizador responda “Não”, a resposta juntamente com os outros atributos serão guardados na tabela “feedbackUsers” na base de dados.

Para a implementação desta notificação foi usado o mesmo método descrito na subsecção 6.1.5.1, usando o mesmo *BroadcastReceiver*. Na Ilustração 14 está representado todo o processo para o envio da notificação ao utilizador. Caso todas as verificações sejam bem-sucedidas, o utilizador é notificado.

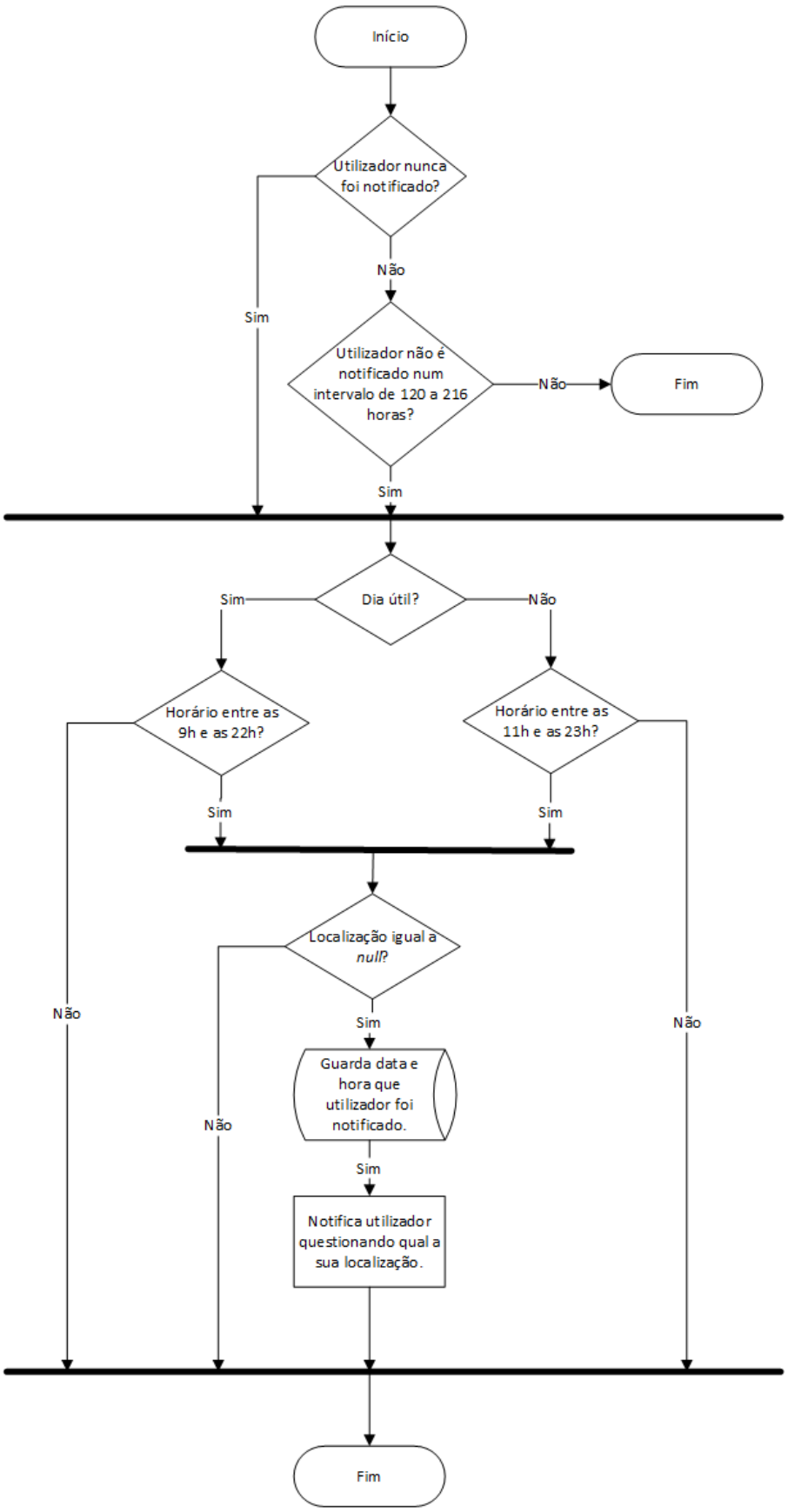


Ilustração 14 - Algoritmo do processo de notificar os utilizadores quando estão numa localização desconhecida

6.1.6. IMPLEMENTAÇÃO DA ESTRATÉGIA PARA ATRIBUIR CREDIBILIDADE ÀS *FINGERPRINTS*

Como referido na secção 5.2, o utilizador será questionado quanto à sua localização, de forma a recolher o *feedback* dele sobre as suas localizações.

O utilizador ao ser questionado tem três possibilidades de resposta: "Sim", "Não", "Não Sei". Sempre que um utilizador responder a uma questão, a resposta é guardada, na tabela "feedbackUsers", juntamente com o identificador da *fingerprint* anotada, que por ser a mais semelhante (processo realizado na Função "fingerprintsRankingV2"), determinou a localização atual do utilizador. Com esta solução os utilizadores vão poder ajudar a definir se as *fingerprints* anotadas pelos utilizadores são credíveis.

Para determinar a credibilidade e qualidade da *fingerprint* anotada, o sistema atribui "*likes*" e *"unlikes"* às *fingerprints* da tabela "fingerprints" da base de dados, isto é, se um utilizador não concordar com a localização estimada pela aplicação, é contabilizado à respetiva *fingerprint* um *"unlike"*, caso contrário é contabilizado um *"like"*.

Para conseguir implementar esta solução a base de dados teve de sofrer alterações. Na tabela "fingerprints" foram adicionados mais dois campos, "*likes*" e "*unlikes*". Estes dois novos campos têm como função contabilizar os *"likes"* e *"unlikes"* dados pelos utilizadores, através do processo acima descrito. Os dois campos são inicializados a zero, para todas as *fingerprints* anotadas na tabela "fingerprints". Consoante os utilizadores forem respondendo, as respostas "Não" ou "Sim" serão contabilizadas no campo "*unlikes*" ou "*likes*", respetivamente (Ilustração 15). A tabela "fingerprintsHistory" da base de dados, também teve de sofrer alterações, pois as *fingerprints* que eram enviadas automaticamente pelo dispositivo móvel, sempre que eram submetidas ao processo de comparação de *fingerprints* (Função "fingerprintsRankingV2") para lhe atribuir um local, eram guardadas nesta tabela, no entanto não era guardado a *fingerprint* anotada que deu origem à atribuição do local escolhido. De maneira a ser possível os utilizadores darem o *feedback* sobre as *fingerprints* era necessário que fosse guardado o identificador da *fingerprint* anotada que deu origem ao local. Para isso foi criado um novo campo "fingerprintSelected" na tabela "fingerprintsHistory" para guardar o identificador da *fingerprint* anotada.

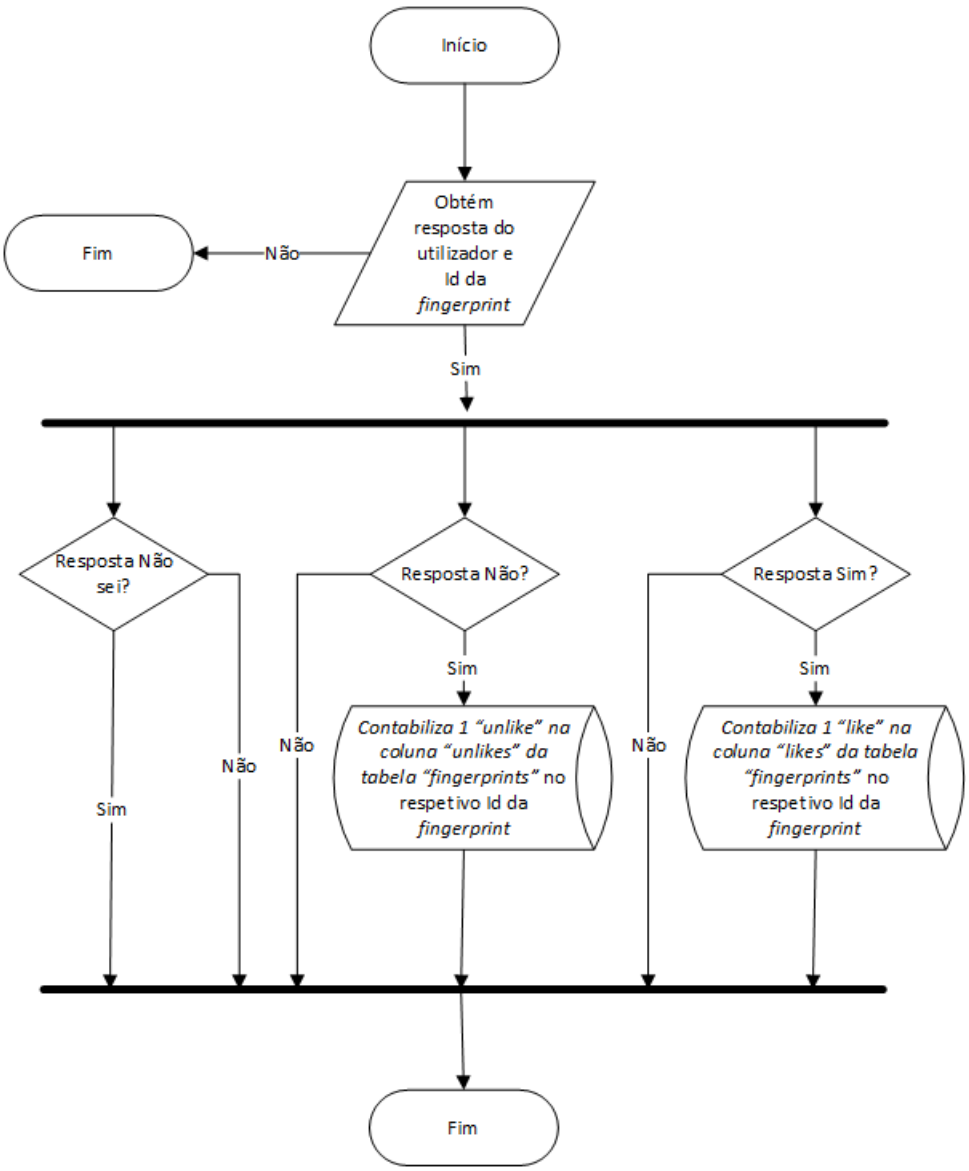


Ilustração 15 - Processo de atribuir credibilidade aos dados inseridos pelos utilizados.

Este processo será essencial para melhorar o processo de estimar a posição do utilizador, para que a aplicação não atribua localizações incorretas. Assim, para estimar a posição dos utilizadores a *fingerprint* não só será comparada com as 10 *fingerprints* mais recentes anotadas por local, mas também passará a ter em conta a credibilidade das *fingerprints* anotadas.

Para calcular a credibilidade de uma *fingerpritnt* anotada (C) serão tidos em conta o total dos "*likes*" (L) e dos "*unlikes*" (U) dessa mesma *fingerprint.* Então, para determinar a credibilidade poderá ser utilizada a seguinte função de cálculo:

$$C = ½ * (1 + (L - U) / (L + U + 1))$$

Se,

$L=U$, então $C=0,5$

$L>U$, então $C>0,5$

$L<U$, então $C<0,5$

Desta forma C toma sempre valores entre 0 e 1, sendo que se C tende para 0 a *fingerprint* anotada não tem nenhuma credibilidade, mas se C tende para 1 a *fingerprint* anotada tem uma excelente credibilidade e é de total confiança.

Assim, o processo de comparação de *fingerprints* para estimar a posição do utilizador realizado na Função "fingerprintsRankingV2" deverá ser alterado. Logo, em vez de serem selecionadas as 10 *fingerprints* anotadas por local ordenadas por data mais recente, a função passará a selecionar as 10 *fingerprints* anotadas por local ordenadas por o valor de credibilidade (C) mais alto. No entanto as *fingerprints* com C<0,5 devem ser rejeitadas, só devem ser selecionadas *fingerprints* com C≥0,5 no processo de comparação de *fingerprints*.

As *fingerprints* com C=0,5 devem ser tidas em conta no processo de comparação de *fingerprints*, pois o utilizador tem a possibilidade de inserir novos locais na aplicação móvel, as *fingerprints* recolhidas nesses locais numa fase inicial não terão qualquer *feedback* do utilizador, no entanto não significa que estejam erradas, mas para que elas consigam ter o *feedback* do utilizador precisam de ser selecionadas no processo de comparação de *fingerprints* para conseguir estimar a localização do utilizador e este conseguir dizer se concorda ou não com a sua localização.

6.2. SERVIDOR

Como referido na secção 4.2 o sistema é constituído por um servidor, que tem como objetivo receber as *fingerprints* enviadas pela aplicação Android, através da interface Wi-Fi do dispositivo móvel. Como já referido anteriormente, este servidor já se encontrava desenvolvido no ínicio deste trabalho de mestrado, no entanto para a implementação de novas funcionalidades foi necessário fazer alterações nos *web services* REST e na base de dados. A linguagem de programação de desenvolvimento dos *web services* é o PHP. Para a comunicação da base de dados com os *web services* é utilizada uma classe de linguagem PHP.

6.2.1. WEB SERVICES

Como já explicado na secção 4.2, existem dois *web services*, um que dá suporte à aplicação móvel e outro que dá suporte à construção do mapa de rádio.

O *web service* que dá suporte à construção do mapa de rádio contém um conjunto de funções que estão apresentadas na secção 4.3. Durante a implementação das melhorias este não precisou de sofrer grandes alterações, apenas na Função "fingerprintsRankingV2" ffoi necessário não apenas retornar o local que o utilizador está, mas também retornar qual a *fingerprint* que deu origem à escolha daquele local. A função "saveFingerprintInDB" também foi alterada para guardar não só os campos que já guardava mas também o novo campo retornado da Função "fingerprintsRankingV2" (esta decisão é explicada melhor na secção 6.2.2). O *web service* que dá suporte à aplicação móvel sofreu algumas alterações e foram acrescentadas novas funções para dar suporte às novas funcionalidades da aplicação móvel. Cada função tem um objetivo e estas funções são explicadas a seguir.

Função "userslogin"

Esta é a função que recebe os dados (*e-mail* e *password*) inseridos pelos utilizadores na aplicação móvel e verifica se estão corretos, permitindo, ou não, o utilizador entrar na sua conta.

Função "usersgcm"

Esta é a função responsável por receber o identificador de registo no serviço GCM e o guardar na base de dados na tabela "devices".

Função "usersRegid"

Sempre que um utilizador deseja enviar uma mensagem a outro utilizador (amigo) a aplicação móvel solicita a esta função o identificador de registo no serviço GCM referente ao amigo a que pretende enviar uma mensagem.

Função "usersmessages"

Esta é a função responsável por receber a mensagem juntamente com o identificador de registo no serviço GCM do destinatário e o identificador do utilizador que enviou a mensagem. Envia

estes dados para o servidor do GCM (invocando a função "send_push_notifications") que se certificará de enviar a mensagem ao utilizador destinatário.

Função "users"

Função responsável por receber os dados de registo de um utilizador, inseridos na aplicação móvel e guardar na base de dados, também é responsável por remover uma conta sempre que o utilizador assim o quiser.

Função "usersnickname"

Função responsável por alterar o *nickname* de um utilizador na base de dados sempre que o utilizador assim o solicitar na aplicação móvel.

Função "userspassword"

Esta função é responsável por atribuir uma nova *password* a um utilizador que assim o tenha solicitado.

Função "userspermission"

Função responsável por alterar na base de dados o estado da partilha da localização de um utilizador com os seus amigos.

Função "usersrequest"

Esta função é a responsável por inserir na base de dados os pedidos de amizade efetuados por um utilizador e listar todos os pedidos pendentes de um utilizador.

Função "userscancel"

Função responsável por cancelar um pedido de amizade emitido pelo próprio utilizador.

Função "usersapprove"

Esta função é responsável por aprovar um pedido de amizade enviado por outro utilizador.

Função "usersdeny"

Esta função é responsável por cancelar um pedido de amizade enviado por outro utilizador.

Função "usersfriends"

Função responsável por listar todos os utilizadores amigos de um utilizador.

Função "usersunfriend"

Sempre que um utilizador decida remover um amigo da sua lista de amigos é invocada esta função.

Função "userscheckins"

Função responsável por obter a informação da última localização conhecida do utilizador e dos seus amigos.

Função "sendquestion1"

Função responsável por enviar uma notificação aos utilizadores questionando se concordam com a localização exibida pela aplicação.

Função "sendquestion2"

Esta função é responsável por enviar uma notificação aos utilizadores alertando que estão numa localização desconhecida, e perguntando se desejam inserir a sua localização atual.

Função "sendresponseQ1"

Sempre que um utilizador responder a uma questão, enviada pela função "sendquestion1", esta função é invocada e tem como função guardar na base de dados, na tabela "feedbackUsers", as respostas emitidas pelos utilizadores e atualizar os campos "likes" e "unlikes" da tabela "fingeprints".

Função "sendresponseQ2"

Quando um utilizador responde a uma questão enviada pela função "sendquestion2", esta é invocada e tem como função guardar na base de dados as respostas emitidas pelos utilizadores.

6.2.2. BASE DE DADOS

A Ilustração 16 esquematiza o modelo físico da base de dados "whereatum" que dá suporte aos *web services*. Os *web services* consultam todas as informações a partir da base de dados e também inserem novas informações nesta.

Para usufruir das novas funcionalidades propostas foi necessário efetuar alterações na base de dados.

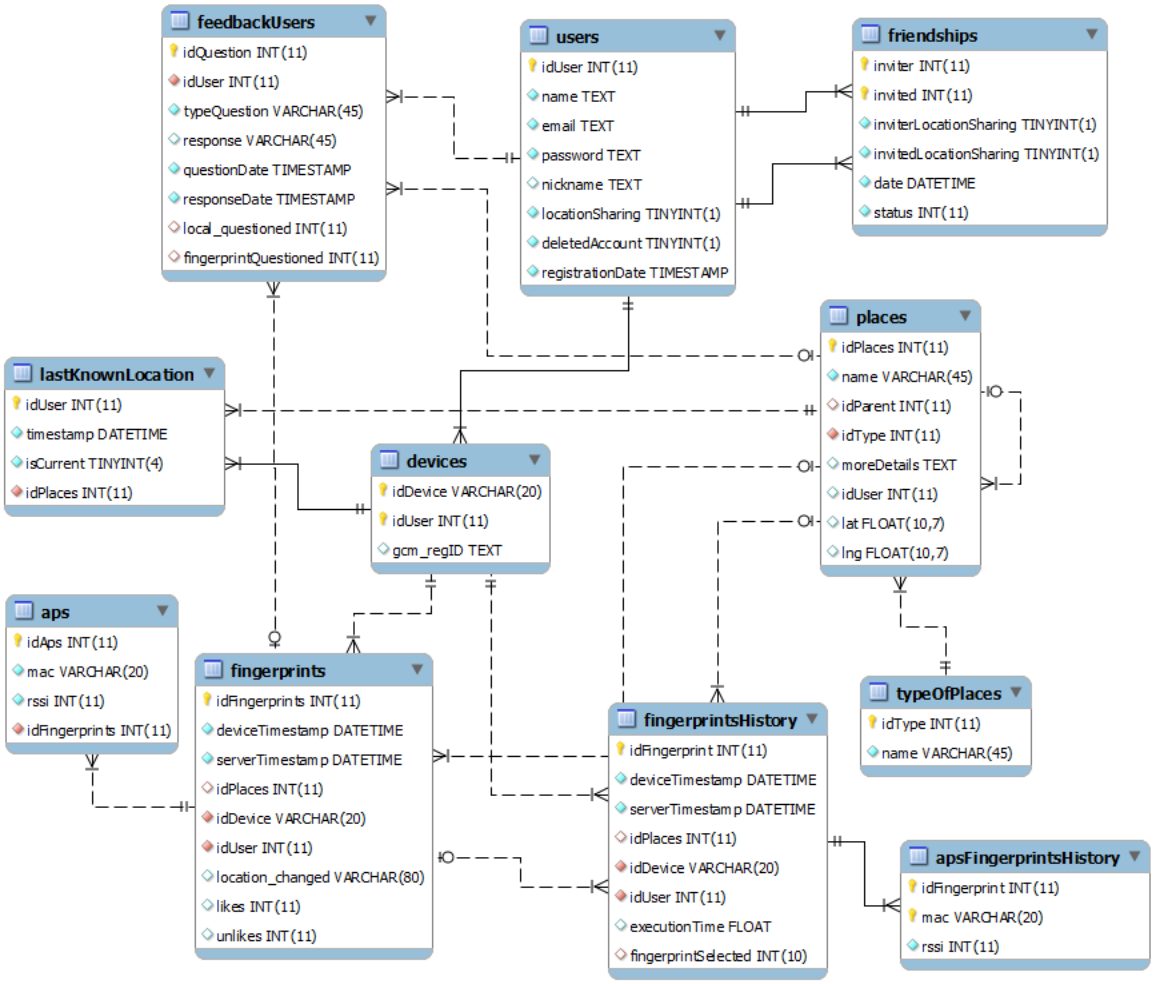


Ilustração 16 - Modelo físico da base de dados "whereatum".

As tabelas contidas no modelo da Ilustração 16 desempenham as funções descritas a seguir.

- Tabela "users" - Esta tabela tem como função guardar os dados relativos aos utilizadores que se registam na aplicação. Guarda também informação sobre o estado da conta do utilizador, caso o utilizador elimine a conta os dados do utilizador não são eliminados da base de dados. A aplicação também permite desativar a partilha de localização com os

utilizadores amigos, essa informação também é guardada nesta tabela. Por fim, foi criado um novo campo nesta tabela onde é guardada a data de registo do utilizador, uma vez esta informação não estar a ser guardada achou-se por bem passar a guardar a data de registo do utilizador, para possíveis consultas mais tarde.

- Tabela "friendships" - Esta tabela contém informação sobre a relação entre utilizadores. Guarda o identificador do utilizador que faz um pedido de amizade, o identificador do utilizador convidado e o estado da relação, que pode ser pendente, terminada ou confirmada. A tabela tem mais dois atributos que neste momento não estão a ser utilizados mas que poderão ser utilizados em operações futuras.
- Tabela "devices" - Sempre que um utilizador se regista na aplicação é guardado nesta tabela o endereço MAC da interface Wi-Fi do dispositivo móvel do utilizador e o identificador do registo efetuado no serviço GCM.
- Tabela "lastKnownLocation" - Nesta tabela é guardada a última localização conhecida dos utilizadores.
- Tabela "fingerprints" - Esta tabela armazena as *fingerprints* anotadas pelos utilizadores através da aplicação. Esta tabela foi alterada e adicionados três novos campos, "location_changed", "likes"e "unlikes". Sempre que um utilizador tem a iniciativa de alterar a sua localização na aplicação móvel por algum motivo, por exemplo o utilizador decide alterar a sua localização por não concordar com o local que a aplicação diz este estar. Caso esta situação aconteça, não só são guardados todos os dados relativos à *fingerprint* anotada pelo utilizador, mas também é guardado o identificador do local que estava anteriormente (antes do utilizador proceder à alteração do local) no campo "local_changed". Permitindo assim perceber se uma *fingerprint* anotada pelo utilizador foi uma nova *fingerprint* ou uma correção de uma *fingerprint* existente. Os dois últimos campos foram adicionados para guardar os "*likes*" e "*unlikes*" respetivos a cada *fingerprint* anotada, atribuídos pelos utilizadores.
- Tabela "aps" - Sempre que é adicionada uma nova *fingerprint* na tabela "fingerprints" é guardada na tabela "aps" os dados dos APs que o dispositivo móvel detetou naquele momento; guarda o endereço MAC e a intensidade do sinal RSSI e o identificador da *fingerpritnt* que foi adicionada naquele momento.
- Tabela "fingerprintsHistory" - Nesta tabela são guardadas todas as *fingerprints*, não anotadas, enviadas automaticamente pelos dispositivos móveis dos utilizadores, sempre

que estes estejam com o Wi-Fi ligado. O sistema ao receber uma *fingerprint*, esta é submetida a um processo de comparação de *fingerprints* para se tentar estimar o local correspondente. No entanto não era guardado a *fingerprint* que deu origem à atribuição do local escolhido. De maneira a ser possível os utilizadores darem o *feedback* sobre as *fingerprints,* foi criado um campo ("fingerprintSelected") que guarda o identificador da *fingerprint* que deu origem ao local escolhido no processo de comparação de *fingerprints*.

- Tabela "apsFingerprintsHistory" - Sempre que é adicionada uma nova *fingerprint* na tabela "fingerprintsHistory" é guardada na tabela "apsFingerprintsHistory" os dados dos APs que o dispositivo móvel detetou naquele momento, guarda o endereço MAC e a intensidade do sinal RSSI e o identificador da *fingerpritnt* que foi adicionada naquele momento.
- Tabela "places" - Esta tabela guarda os locais inseridos pelos utilizadores através da aplicação móvel.
- Tabela "typeOfPlaces" - Esta tabela contém todos os tipos de locais existentes na tabela "places", como país, cidade, universidade, edifício, etc., a sua função é auxiliar a tabela "places" a armazenar os locais.
- Tabela "feedbackUsers" - Esta é uma nova tabela, criada para guardar as opiniões dos utilizadores emitidas pela aplicação móvel sobre os locais. Esta tabela tem 8 campos que guardam o identificador da questão, identificador do utilizador que respondeu à questão, tipo de questão, isto é, se é uma questão para confirmar o local onde o utilizador se encontra ou para inserir um novo local quando o utilizador está num local desconhecido. Contém também a resposta dada pelo utilizador, a data que o servidor enviou a questão, a data em que o utilizador respondeu à questão, o identificador do local que foi questionado e por fim o identificador da *fingerprint* que deu origem à estimação do local do utilizador. A partir da informação desta tabela é possível calcular os "*likes*" e "*unlikes*" das *fingerprints* anotadas, através das respostas dadas pelos utilizadores e atualizar os campos "likes" e "unlikes" da tabela "fingerprints"

6.3. IMPLEMENTAÇÃO DA FERRAMENTA DE MONITORIZAÇÃO NAGIOS

Após o levantamento de algumas aplicações existentes foi possível concluir que o Nagios é o mais indicado para a monitorização do sistema de posicionamento, pois é uma ferramenta que reúne um maior nível de aceitação pelos utilizadores e é bem documentada. Depois de recolhidas

algumas opiniões positivas de profissionais da área, verificou-se a satisfação em relação à ferramenta, o que reforçou essa escolha.

6.3.1. INSTALAÇÃO DO NAGIOS

Para a instalação do Nagios, foi necessário saber quais os seus pré-requisitos.

6.3.1.1. Pré-Requisitos

Os pré-requisitos do Nagios são bastante simples, o sistema funciona em qualquer computador com Linux que cumpra os seguintes pré-requisitos de *software* (A. R. R. Pinto & Coutinho, 2012):

- Servidor Apache;
- PHP;
- Bibliotecas GD;
- Compiladores GCC (incluindo g++).

Para a instalação do Nagios foi usada uma máquina com o Windows 10. Nesta máquina foi instalado o Oracle VM VirtualBox Manager, para ser possível criar uma máquina virtual com o sistema operativo CentOS 7.

Depois da instalação e configuração do CentOS 7 procedeu-se à instalação de todos os pré-requisitos. Feito isto, foi feito a instalação do Nagios e os *plugins*.

O projeto Nagios e os *pulgins* foram obtidos no site http://www.nagios.org/download/. Depois da instalação do Nagios, a sua interface *web* ficou disponível no endereço http://localhost/nagios/. Nesta interface *web* tem-se acesso a várias funcionalidades do Nagios, como ver o estado dos serviços, reagendar verificação do estado dos serviços, etc. No Anexo A pode ser vista com mais detalhe toda a instalação e configuração do Nagios.

6.3.2. MONITORIZAÇÃO DO SISTEMA DE POSICIONAMENTO

Para se proceder à monitorização do sistema de posicionamento, foi configurado o servidor a monitorizar, uma máquina Linux, neste caso urano.dsi.uminho.pt. Todos os ficheiros de configuração estão no caminho /etc/nagios/objects, esses ficheiros foram configurados com os dados da máquina Linux a monitorizar. Nestes ficheiros foi definido também o intervalo de tempo

que o Nagios vai verificar a disponibilidade do serviço da máquina Linux, que deve ser a cada intervalo de 5 minutos. Também foi definido o grupo de contactos a ser notificado caso haja problemas.

Posteriormente foi adicionado um *script* específico para verificar a disponibilidade do serviço e da base de dados, o *script* check_service_nagios.php. A localização deste ficheiro encontra-se numa pasta específica do nagios. Este script tem como função comunicar com o script, check_service.php, localizado na máquina urano.dsi.uminho.pt. O *script* check_service.php é uma página php, que consiste em pedir o número de utilizadores que estão registados na base de dados, se o número de utilizadores for diferente de *null* ou 0 a base de dados está disponível, se for *null* ou 0 a base de dados está indisponível. Este resultado vai ser retornado à página check_service_nagios.php que vai avaliar a resposta e caso obtenha uma resposta diferente de *null* ou 0, o Nagios exibe que o estado do serviço é "OK", ou seja, está disponível. Caso contrário exibe que o estado do sistema é "*CRITICAL*", ou seja, não está disponível, se assim for os administradores são alertados via *e-mail*. Os *scripts* criados podem ser vistos com mais detalhe no Anexo A.

Na Ilustração 17 está apresentado o funcionamento geral da solução de monitorização descrito atrás.

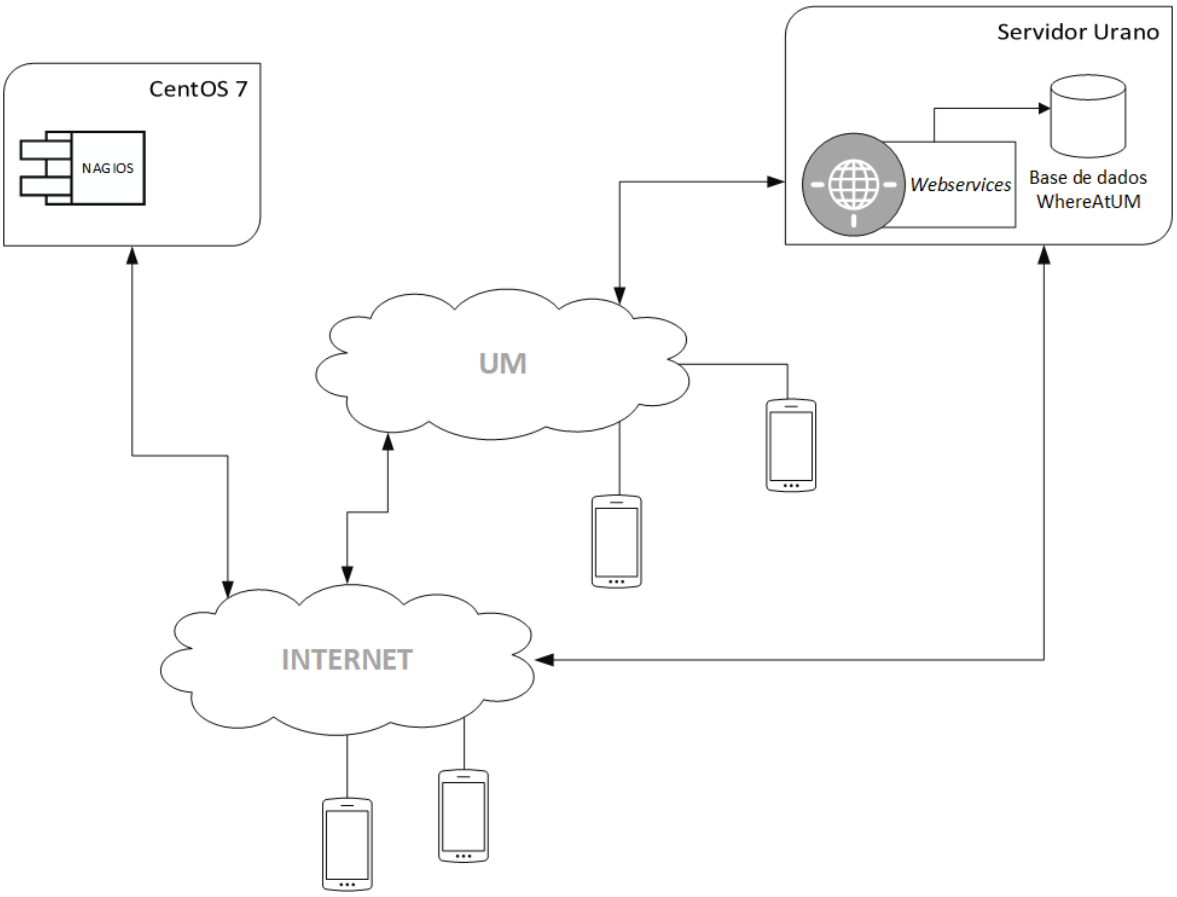


Ilustração 17 - Esquema geral do funcionamento da solução de monitorização.

Como referido anteriormente, a ferramenta de monitorização foi apenas implementada numa máquina virtual, entretanto a intenção era migrar a solução para um servidor da UM, tal não aconteceu por o servidor não ter sido disponibilizado em tempo útil. Contudo houve a noção que esta solução ao ser transferida para um servidor na rede da UM, caso houvesse algum problema com a rede, o alerta do Nagios não seria enviado, logo não se seria notificado mal o problema ocorresse. Assim surgiu a ideia de se criar outro *script* que consistia em obter quantos utilizadores estavam ativos naquele dia, até à hora que o *script* era executado. Este *script* era executado duas vezes ao dia, a primeira às 11 horas e a segunda às 23 horas, cada vez que era executado emitia um alerta via *e-mail*. Caso não se recebesse os *e-mails* às horas referidas atrás era certo que algo se passava com a rede da UM, e que a solução de monitorização não estaria a funcionar.

Porém havia um problema o Nagios só notifica quando algo não está bem, logo colocou-se o *script* a retornar ao Nagios um resultado "*WARNING*", e assim ele passou a notificar. Só que enquanto há um problema, neste caso um "*WARNING*", enquanto o problema não for resolvido o Nagios está constantemente a enviar alertas via *e-mail*, para que o administrador resolva o quanto antes o problema.

Visto a segunda solução não funcionar como pretendido, optou-se por instalar uma interface *web* chamada Thruk que tem mais funcionalidades que a interface original da Nagios. Com a instalação do Thruk foi possível agendar o envio de relatórios via *e-mail*, sobre a máquina que está a ser monitorizada. Então a solução anterior foi eliminada e passaram a ser enviados relatórios 2 vezes ao dia, às 11 horas e às 23 horas. Caso estes relatórios não cheguem nas horas previstas, o administrador responsável por manter o bom funcionamento da rede da UM, suporia que algo não estaria bem com a rede da UM, e que a solução de monitorização não estaria a funcionar. A instalação do Thruk pode ser vista com mais detalhe no Anexo B.

Na Ilustração 18 está representado detalhadamente o estado dos serviços de todos os *hosts* que estão a ser monitorizados, como a hora da última verificação, informação do estado, etc.

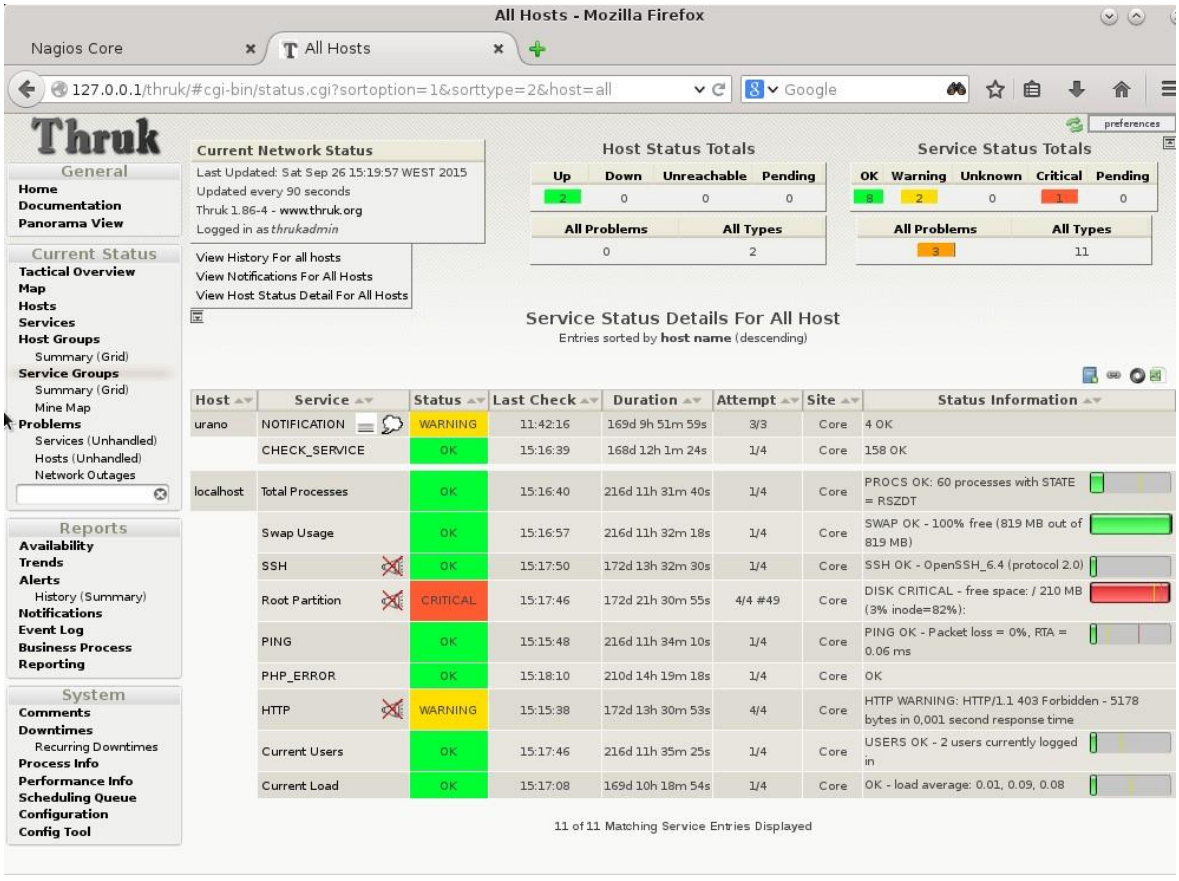


Ilustração 18 - Detalhes do estado dos serviços de todos os *hosts.*

Na Ilustração 19 pode ser visto os dados do relatório que é enviado para o *e-mail* dos administradores, duas vezes ao dia. Também é neste menu que dá para criar novos relatórios e editar os relatórios existentes.

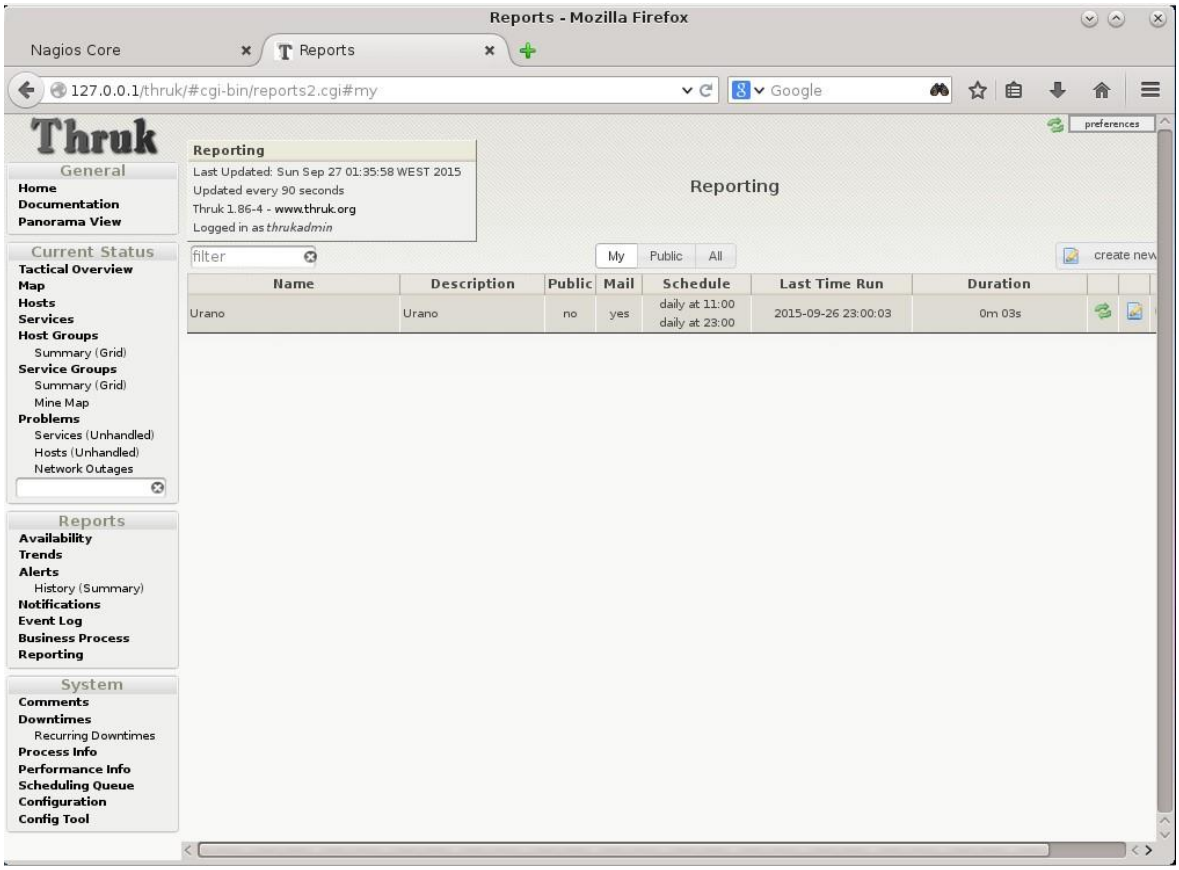


Ilustração 19 - Detalhes dos relatórios criados.

7. TESTES E RESULTADOS

Este capítulo apresenta os testes efetuados tendo como objetivo validar os resultados obtidos para a aplicação móvel e para a solução de monitorização.

7.1. RESULTADOS DOS TESTES EFETUADOS À APLICAÇÃO MÓVEL

De modo a testar a aplicação móvel foram efetuados alguns testes que consistiam na troca de mensagens entre dois dispositivos, para perceber e visualizar o comportamento da aplicação móvel. Após a realização destes testes foi possível descobrir falhas que a aplicação apresentava e proceder à resolução das mesmas. A seguir são apresentados os resultados obtidos em todos os testes realizados.

7.1.1. TESTES AO SERVIÇO DE ENVIO E RECEÇÃO DE MENSAGENS

Teste 1

Tabela 5 - Resultados do primeiro teste ao serviço de envio e receção de mensagens.

Verificação	Descrição	Resultado
Aceder à aplicação.	Verificar se a aplicação arranca sem problemas.	Passou
Listagem das informações.	Verifica se a aplicação é capaz de exibir as informações no ecrã principal da aplicação.	Passou
Aceder ao ecrã de mensagens.	O utilizador tenta aceder ao ecrã de envio de mensagens.	Passou
Enviar mensagem.	O utilizador escreve uma mensagem para o utilizador destinatário e clica no botão enviar.	Passou
Verificar se notificações de novas mensagens são enviadas.	O utilizador verifica se recebe notificações ao receber uma nova mensagem de outro	Falhou

	utilizador (estado da aplicação: encerrada).	
Verificar se notificações de novas mensagens são enviadas.	O utilizador verifica se recebe notificações ao receber uma nova mensagem de outro utilizador (estado da aplicação: encerrada).	Falhou
Verificar se a mensagem recebida aparece no ecrã de mensagens.	Verificar se a mensagem recebida é guardada no ecrã das mensagens do respetivo utilizador remetente (estado da aplicação: encerrada).	Falhou
Verificar se a mensagem recebida aparece no ecrã de mensagens.	Verificar se a mensagem recebida é guardada no ecrã das mensagens do respetivo utilizador remetente (estado da aplicação: iniciada).	Falhou
Listagem das mensagens enviadas e recebidas.	Verificar se é possível observar um histórico de mensagens recebidas e enviadas a um determinado utilizador, no ecrã de mensagens.	Falhou

No primeiro teste (Tabela 5) a maioria das verificações falharam, isto deveu-se ao facto do registo do endereço IP (*Internet Protocol*) do servidor que envia as mensagens para o servidor da GCM, estar errado. Este problema fazia com que as mensagens enviadas nunca chegassem aos destinatários. Então procedeu-se à correção do endereço IP do servidor, no seu registo no servidor GCM.

Teste 2

Tabela 6 - Resultados do segundo teste ao serviço de envio e receção de mensagens.

Verificação	Descrição	Resultado
Aceder à aplicação.	Verificar se a aplicação arranca sem problemas.	Passou
Listagem das informações.	Verifica se a aplicação é capaz de exibir as informações no ecrã principal da aplicação.	Passou
Aceder ao ecrã de mensagens.	O utilizador tenta aceder ao ecrã de envio de mensagens.	Passou
Enviar mensagem.	O utilizador escreve uma mensagem para o utilizador destinatário e clica no botão enviar.	Passou
Verificar se notificações de novas mensagens são enviadas.	O utilizador verifica se recebe notificações ao receber uma nova mensagem de outro utilizador (estado da aplicação: encerrada).	Passou
Verificar se notificações de novas mensagens são enviadas.	O utilizador verifica se recebe notificações ao receber uma nova mensagem de outro utilizador (estado da aplicação: iniciada).	Passou
Verificar se a mensagem recebida aparece no ecrã de mensagens.	Verificar se a mensagem recebida é guardada no ecrã das mensagens do respetivo utilizador remetente (estado da aplicação: encerrada).	Falhou
Verificar se a mensagem recebida aparece no ecrã de	Verificar se a mensagem recebida é guardada no ecrã	Passou

mensagens.	das mensagens do respetivo utilizador remetente (estado da aplicação: iniciada).	
Listagem das mensagens enviadas e recebidas.	Verificar se é possível observar um histórico de mensagens recebidas e enviadas a um determinado utilizador, no ecrã de mensagens.	Falhou

Após ter sido corrigido o problema descrito anteriormente foi realizado um novo teste (Tabela 6). No entanto a aplicação continuava a falhar em algumas verificações, para tentar resolver o problema foi realizada uma análise ao código-fonte da aplicação móvel, para identificar e tentar resolver os "*bugs*" que impediam a aplicação móvel funcionar corretamente. Após a análise foram feitas algumas alterações no código-fonte da aplicação móvel.

Teste 3

Tabela 7 - Resultados do terceiro teste ao serviço de envio e receção de mensagens.

Verificação	Descrição	Resultado
Aceder à aplicação.	Verificar se a aplicação arranca sem problemas.	Passou
Listagem das informações.	Verifica se a aplicação é capaz de exibir as informações no ecrã principal da aplicação.	Passou
Aceder ao ecrã de mensagens.	O utilizador tenta aceder ao ecrã de envio de mensagens.	Passou
Enviar mensagem.	O utilizador escreve uma mensagem para o utilizador destinatário e clica no botão enviar.	Passou
Verificar se aplicação envia	O utilizador verifica se a	Passou

notificação à receção de uma nova mensagem.	aplicação é capaz de enviar uma notificação ao receber uma mensagem de outro utilizador (estado da aplicação: encerrada).	
Verificar se aplicação envia notificação à receção de uma nova mensagem.	O utilizador verifica se a aplicação é capaz de enviar uma notificação ao receber uma mensagem de outro utilizador (estado da aplicação: iniciada).	Passou
Verificar se a mensagem recebida aparece no ecrã de mensagens.	Verificar se a mensagem recebida é guardada no ecrã das mensagens do respetivo utilizador remetente (estado da aplicação: encerrada).	Falhou
Verificar se a mensagem recebida aparece no ecrã de mensagens.	Verificar se a mensagem recebida é guardada no ecrã das mensagens do respetivo utilizador remetente (estado da aplicação: iniciada).	Passou
Listagem das mensagens enviadas e recebidas.	Verificar se é possível observar um histórico de mensagens recebidas e enviadas a um determinado utilizador, no ecrã de mensagens.	Passou

Após algumas alterações no código-fonte, para tentar corrigir os "*bugs*" detetados anteriormente, foi possível verificar num terceiro teste (Tabela 7) que continuava a persistir uma falha. Posteriormente foram feitas várias tentativas de resolução do problema mas sem sucesso.

7.1.2. TESTES AOS ALERTAS ENVIADOS QUESTIONANDO OS UTILIZADORES

Teste 1

Tabela 8 - Resultados do primeiro teste à funcionalidade de recolher o *feedback* dos utilizadores.

Verificação	Descrição	Resultado
Aceder à aplicação.	Verificar se a aplicação arranca sem problemas.	Passou
Verificar se pode notificar o utilizador	Verificar se o utilizador nunca foi notificado, ou se já passou o tempo necessário para ser notificado novamente.	Passou
Verificar se é dia útil ou se é fim-de-semana	Verificar o dia de semana para saber em que horário o sistema deve/pode notificar o utilizador.	Passou
Verificar se o utilizador pode ser notificado	Verificar se o utilizador está numa hora do dia ideal para ser notificado.	Passou
Verificar se há localização conhecida	Verificar se a localização do utilizador é desconhecido ou conhecida	Passou
Verificar se a notificação é exibida	Verificar se o utilizador é questionado sobre a sua localização, exibida pela aplicação, estar correta ou não.	Falhou
Receber *feedback* do utilizador	Verificar se o *feedback* do utilizador está a ser guardado na base de dados.	Falhou
Atribuir "*likes*" e "*unlikes*"	Verificar se o sistema está a atribuir "*likes*" e "*unlikes*" às	Falhou

	fingerprints baseado nas respostas dos utilizadores.	
Listagem das informações.	Verifica se a aplicação é capaz de exibir as informações no ecrã principal da aplicação.	Falhou

Numa primeira fase, após o desenvolvimento desta funcionalidade, algumas verificações falharam (Tabela 8). Sempre que se tentava iniciar a aplicação, ela deixava de responder acabando por encerrar. Para resolver este problema foi revisto o código-fonte da aplicação e efetuadas alterações.

Teste 2

Tabela 9 - Resultados do segundo teste à funcionalidade de recolher o *feedback* dos utilizadores.

Verificação	Descrição	Resultado
Aceder à aplicação.	Verificar se a aplicação arranca sem problemas.	Passou
Verificar se pode notificar o utilizador	Verificar se o utilizador nunca foi notificado, ou se já passou o tempo necessário para ser notificado novamente.	Passou
Verificar se é dia útil ou se é fim-de-semana	Verificar o dia de semana para saber em que horário o sistema deve/pode notificar o utilizador.	Passou
Verificar se o utilizador pode ser notificado	Verificar se o utilizador está numa hora do dia ideal para ser notificado.	Passou
Verificar se há localização conhecida	Verificar se a localização do utilizador é desconhecido ou conhecida	Passou

Verificar se a notificação é exibida	Verificar se o utilizador é questionado sobre a sua localização, exibida pela aplicação, estar correta ou não.	Passou
Receber *feedback* do utilizador	Verificar se o *feedback* do utilizador está a ser guardado na base de dados.	Passou
Atribuir "*likes*" e "*unlikes*"	Verificar se o sistema está a atribuir "*likes*" e "*unlikes*" às *fingerprints* baseado nas respostas dos utilizadores.	Passou
Listagem das informações.	Verifica se a aplicação é capaz de exibir as informações no ecrã principal da aplicação.	Passou

Depois de efetuadas as alterações foi possível verificar num segundo teste (Tabela 9) que as verificações eram todas efetuadas com sucesso.

7.2. RESULTADOS DOS TESTES EFETUADOS À SOLUÇÃO DE MONITORIZAÇÃO

A solução de monitorização por ter sido apenas implementada numa máquina virtual, instalada numa máquina pessoal, não permitiu efetuar testes quanto ao seu desempenho a monitorizar os serviços do sistema de posicionamento. No entanto foi possível verificar se a solução implementada está a funcionar como pretendido (Tabela 10).

Teste

Tabela 10 - Resultados do teste à solução de monitorização.

Verificação	Descrição	Resultados
Verificar envio de relatórios	Verificar se são enviados dois relatórios sobre o estado do	Passou

	sistema monitorizado, duas vezes ao dia.	
Verificar disponibilidade do serviço	Verificar a cada 5 minutos a disponibilidade do serviço.	Passou
Envio de *e-mails*	Verificar se é enviado um *email* sempre que é detetado um problema nos serviços.	Passou

8. CONCLUSÕES E TRABALHO FUTURO

8.1. CONCLUSÕES

Esta dissertação focou-se em melhorar um sistema de posicionamento existente. Para o desenvolvimento desta dissertação foram estudas tecnologias e técnicas utilizadas para estimar a localização em ambientes interiores. Foram também estudados alguns sistemas de localização existentes. Estes estudos vieram a potenciar uma familiarização com esta área e de que forma era possível otimizar o mecanismo para estimar a posição do utilizador no sistema de posicionamento. Ao longo do desenvolvimento da dissertação foi possível perceber que por vezes o sistema de posicionamento não estava disponível, o que impedia que a aplicação móvel funcionasse corretamente. Por isso foi decidido também fazer um pequeno estudo sobre as ferramentas de monitorização existentes e implementar uma solução que nos permitisse perceber sempre que o sistema de posicionamento ficasse indisponível.

Depois de todo o estudo desenvolvido foram delineadas algumas propostas de melhoria do sistema de posicionamento e implementadas. Foi desenvolvido um serviço de envio e receção de mensagens na aplicação móvel de modo a atrair e cativar a atenção dos utilizadores, para que estes utilizem a aplicação regularmente. Foram também criados alertas para potenciar a interação do utilizador com a aplicação móvel e ao mesmo tempo recolher o *feedback* dos utilizadores relativamente à sua localização, e, posteriormente, poder avaliar a credibilidade das *fingerprints* anotadas pelos utilizadores. Por fim foi também encontrada uma solução capaz de verificar a disponibilidade do sistema de posicionamento.

Em suma, pode-se então concluir que a maioria dos objetivos propostos foram alcançados, embora a implementação da estratégia para atribuir credibilidade às *fingerprints* não tenha sido terminada por completo, falta completar o último passo que consiste em alterar o processo de estimação da localização para a *fingerprint* não ser só comparada com 10 *fingerprints* mais recentes anotadas por local, mas passe também a ter em conta os "*likes*" e "*unlikes*" atribuídos a cada *fingerprint* anotada. A solução de monitorização foi implementada e testada apenas numa máquina virtual, a intenção era migrar a solução para um servidor da UM, no entanto o servidor não foi disponibilizado em tempo útil, impedindo a sua implementação.

8.2. TRABALHO FUTURO

O sistema de posicionamento poderá ter no futuro diversos aperfeiçoamentos, nomeadamente no que diz respeito a novas funcionalidades na aplicação móvel, visto os utilizadores serem fulcrais para o sucesso deste sistema de posicionamento, estes necessitam de novidades para se manterem interessados. Uma funcionalidade poderia passar pela possibilidade do utilizador poder criar grupos de amigos com a partilha da localização configurável para cada um. Possibilidade do utilizador criar alertas por áreas (*geofencing*), os utilizadores seriam alertados sempre que entrassem ou saíssem de uma área específica e também quando os seus amigos entrassem nessa área. Desta forma os utilizadores seriam alertados sempre que um amigo estivesse por perto e, por exemplo, com a funcionalidade das mensagens conseguiriam contactar esse amigo e combinarem encontrarem-se para conversarem. Os utilizadores poderiam achar esta funcionalidade engraçada e com isso serem utilizadores frequentes da aplicação móvel. Integrar a aplicação com as redes sociais, ou seja, capacidade de fazer login com o facebook para que os utilizadores não tivessem que perder muito tempo em registarem-se na aplicação. Incorporar os mapas dos campi da Universidade do Minho, identificando a posição do utilizador no mapa, integrar a aplicação com o Google Maps para quando o utlizador estiver fora dos campi da universidade seriam igualmente funcionalidades que iriam certamente captar a atenção do utilizador.

Seria igualmente importante terminar a implementação da estratégia de atribuir credibilidade às *fingerprints* anotadas pelos utilizadores e testar esta solução.

Assim que for disponibilizado um servidor era também importante implementar a solução proposta para monitorizar não só o sistema de posicionamento, mas também outros sistemas igualmente importantes.

REFERÊNCIAS BIBLIOGRÁFICAS

Android, D. (2015). Google Cloud Messaging | Desenvolvedores Android. Obtido 26 de Fevereiro de 2015, de http://developer.android.com/google/gcm/gcm.html

Beal, V. (2015). Wi-Fi. Obtido 19 de Fevereiro de 2015, de http://www.webopedia.com/TERM/W/Wi_Fi.html

Bhasker, E. S., Brown, S. W., & Griswold, W. G. (2004). Employing user feedback for fast, accurate, low-maintenance geolocationing. *Proceedings - Second IEEE Annual Conference on Pervasive Computing and Communications, PerCom*, (Section 2), 111–120. doi:10.1109/PERCOM.2004.1276850

Bisatto, A. P., & Peres, A. (2009). Localização de Estação Sem Fio Utilizando Trilateração. *XII Seminário Intermunicipal de Pesquisa*, 1–19.

Boonsriwai, S., & Apavatjrut, A. (2013). Indoor WIFI localization on mobile devices. *2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, 1–5. doi:10.1109/ECTICon.2013.6559592

Brás, L. P. M. (2009). *Desenvolvimento de Sistemas de Localização Indoor de Baixo Consumo*, Mestrado em Engenharia Electrónica e Telecomunicações, Departamento de Informática, Universidade de Aveiro.

Coelho, F. (2009). *Computação Ubíqua para Aplicações em Saúde*, Mestrado Integrado em Engenharia Electrotécnica e de Computadores, Faculdade de Engenharia da Universidade do Porto.

Couto, C. (2003). *Identificação por Radiofrequência Carlos Couto*. Guimarães. Obtido de http://www.dei.isep.ipp.pt/~qtdei/RFID_300403.pdf

Deloitte. (2014). *Short Messaging Services versus Instant Messaging: Value versus volume* (Vol. D). London.

Disha, A. (2013). A Comparative Analysis on indoor positioning Techniques and Systems. *International Journal of Engineering Research and Applications*, *3*(2), 1790–1796.

Dursch, A., Yen, D. C., & Shih, D. H. (2004). Bluetooth technology: An exploratory study of the analysis and implementation frameworks. *Computer Standards and Interfaces*, *26*, 263–277. doi:10.1016/j.csi.2003.12.005

Eurotux. (2013). *eurotux - Sistema de Monitorização*.

Eurotux Informática. (2013). Sistemas de Monitorização — Eurotux. Obtido 6 de Fevereiro de 2015, de http://eurotux.com/produtos-e-solucoes/produtos-eurotux/sistemas-de-monitorizacao

Fernandes, J. P. da F. (2012). *Localização em Redes Wi-Fi*, Mestrado em Engenharia Informática, Escola de Engenharia, Universidade do Minho.

Gallagher, T., Li, B., Dempster, A. G., & Rizos, C. (2010). Database updating through user feedback in fingerprint-based Wi-Fi location systems. *2010 Ubiquitous Positioning Indoor Navigation and Location Based Service, UPINLBS 2010*. doi:10.1109/UPINLBS.2010.5654329

Gu, Y., Lo, A., Member, S., & Niemegeers, I. (2009). A Survey of Indoor Positioning Systems for Wireless Personal Networks. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, *11*(1), 13–32.

Hightower, J., & Borriello, G. (2001). Location Sensing Techniques. *IEEE Computer magazine*, (August), 1–8. doi:10.1109/MOBHOC.2007.4428622

Järvinen, P. (2007). Action research is similar to design science. *Quality and Quantity*, *41*(1), 37–54. doi:10.1007/s11135-005-5427-1

Josang, A., & Golbeck, J. (2009). Chanllenges for robust trust and reputation systems. *5th International Workshop on Security and Trust Management (STM 2009)*, (September), 1–12.

Ledlie, J., Park, J., Curtis, D., Cavalcante, A., Camara, L., Costa, A., & Vieira, R. (2012). Molé: a scalable, user-generated WiFi positioning engine. *Journal of Location Based Services*, *6*(2), 55–80. doi:10.1080/17489725.2012.692617

Liu, H., Member, S., Darabi, H., Banerjee, P., & Liu, J. (2007). Survey of Wireless Indoor Positioning Techniques and Systems, *37*(6), 1067–1080.

Mahtab Hossain, a. K. M., Nguyen Van, H., & Soh, W. S. (2010). Utilization of user feedback in indoor positioning system. *Pervasive and Mobile Computing*, *6*(4), 467–481. doi:10.1016/j.pmcj.2010.04.003

Matos, D. A. da S. (2014). *where@UM - Aplicação móvel de posicionamento*, Mestrado em Engenharia de Comunicações, Escola de Engenharia, Universidade do Minho.

Nagios. (2014). Nagios Overview. Obtido 7 de Fevereiro de 2015, de http://www.nagios.org/about/overview/

Ni, L. M., Liu, Y., Lau, Y. C., & Patil, A. P. (2004). LANDMARC: Indoor Location Sensing Using Active RFID. *Wireless Networks*, *10*, 701–710. doi:10.1023/B:WINE.0000044029.06344.dd

OpenNMS. (2014). O que é OpenNMS - OpenNMS. Obtido 6 de Fevereiro de 2015, de http://www.opennms.org/wiki/What_is_OpenNMS

Paciga, M., & Lutfiyya, H. (2004). Herecast : An Open Infrastructure for Location- Based Services Using WiFi. *Access*, *9*.

Pendão, C. G. (2012). *Recolha de Dados de Movimento em Dispositivos Móveis Pessoais*, Mestrado em Engenharia de Comunicações, Escola de Engenharia, Universidade do Minho. Obtido de http://hdl.handle.net/1822/25757

Pereira, D. R. B. (2012). *UbiShare - Partilha Interativa de Conteúdos em Ambientes Ubíquos*, Mestrado em Sistemas e Tecnologias de Informação para as Organizações, Escola Superior de

Tecnologia e Gestão de Viseu, Instituto Politécnico de Viseu.

Pinto, A. R. R., & Coutinho, M. A. da S. (2012). *Inventariação e Monitorização de Sistemas e Redes*, Licenciatura em Engenharia Informática, Escola Superior de Tecnologia e de Gestão, Instituto Politécnico de Bragança.

Pinto, H. T. G. (2009). *Desenvolvimento de Modelos de Localização para Tecnologias de Redes sem Fios*, Mestrado em Engenharia Electrotécnica e de Computadores, Escola de Ciências e Tecnologia, Universidade de Trás-os-Montes e Alto Douro.

Pinto, P. (2010). Nágios – Monitorização Open Source. Obtido 29 de Janeiro de 2015, de http://pplware.sapo.pt/tutoriais/networking/nagios-%e2%80%93-monitorizacao-open-source/#comments

Queirós, R. (2014). *Desenvolvimento de Aplicações Profissionais em Android*. (F.-E. Informática, Ed.).

Rebocho, N. M. G. (2013). *Reputação e Recomendação como Serviços para Transportes Públicos*, Mestrado em Engenharia Informática e de Computadores, Instituto Superior de Engenharia de Lisboa.

Redpin. (2008). Redpin - Indoor Positioning para o resto de nós. Obtido 3 de Janeiro de 2015, de http://redpin.org/

Sanpechuda, T., & Kovavisaruch, L. (2008). A review of RFID localization: Applications and techniques. *5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, ECTI-CON 2008*, *2*, 769–772. doi:10.1109/ECTICON.2008.4600544

The Editors of Encyclopædia Britannica. (2014). Wi-Fi. Obtido 19 de Fevereiro de 2015, de http://www.britannica.com/EBchecked/topic/1473553/Wi-Fi

Youssef, M., & Agrawala, A. (2005). The Horus WLAN location determination system. *Proceedings of the 3rd international conference on Mobile systems, applications, and services - MobiSys '05*, 205–218. doi:10.1145/1067170.1067193

Zabbix. (2011a). Zabbix. Obtido 6 de Fevereiro de 2015, de http://www.zabbix.com/monitor_everything.php

Zabbix. (2011b). Zabbix. Obtido 6 de Fevereiro de 2015, de http://www.zabbix.com/proactive_monitoring.php

APÊNDICE A

Para continuar com a implementação das melhorias à aplicação móvel foi decidido migrar todo o sistema de posicionamento para um novo servidor, para minimizar o impacto aos utilizadores. Assim foi criado um ambiente de desenvolvimento e outro para produção.

Plano de migração do Servidor Linux para o Servidor Windows

Passo 1: Criar conta utilizador no servidor Windows e obter credenciais de acesso do servidor Linux
Primeiramente deverá ser criada uma conta no novo servidor e serem obtidas as credenciais do servidor Linux, para se puder ter acesso às máquinas e fazer tudo que seja necessário para a migração do servidor Linux.

Passo 2: Instalação do *software* no servidor Windows

MySQL 5.5.29

Apache 2.4.10

PHP 5.6.4

Passo 3: Fazer a exportação da BD (base de dados)1 e importar no servidor Windows (2)
Neste passo será feito o *backup* da base de dados do servidor Linux (1) e deverá ser restaurada no servidor 2.

Passo 4: instalar o serviço 2 no servidor 2

Passo 5: Criar uma versão da app (aplicação) (app 2) que fale com o novo servidor; igual à app 1, mas que fala com um servidor em http://where.dsi.uminho.pt em vez de http://urano.dsi.uminho.pt/whereatum

Passo 6: Testar a aplicação 2 no servidor 2
Nesta fase devem ser escolhidos alguns utilizadores para testar a nova aplicação no servidor Windows de maneira a perceber se tudo está a funcionar bem e se não há falhas.

Passo 7: Verificar se o servidor 1 consegue comunicar com a BD do servidor 2

Passo 8: Parar o serviço 1 no servidor 1, e o serviço 2 no servidor 2

Passo 9: Copiar a BD 1 para a BD 2 (outra vez)

Passo 10: Alterar o serviço 1 para falar com a BD 2

Passo 11: Ativar o serviço 1 e o serviço 2

Passo 12: Verificar que o serviço 1 comunica bem com a BD 2 (usando a app 1)

Passo 13: Verificar o tipo de *logs* que estão a ser feitos no Apache do Linux, e verificar o acesso a esses *logs* (importante para o passo 10)

Passo 14: Colocar aplicação 2 no Google Play

A aplicação terá de ser colocada no Google Play para que todos os utilizadores consigam atualizar as suas aplicações. Isto gera automaticamente notificações para que os utilizadores atualizem a sua app.

Passo 15: Notificar utilizadores da aplicação 1 para mudarem para a nova aplicação

Nesta fase os utilizadores devem ser incentivados a instalar a nova versão da aplicação e avisados por *e-mail* de que esta será descontinuada num prazo de uma semana; argumentar que a nova versão corrige alguns bugs; esta notificação deve ser feita logo que a aplicação esteja disponível no Google Play.

Passo 16: Processo para identificar os utilizadores da aplicação 1

De maneira a conseguirmos identificar quais os utilizadores que ainda não instalaram a nova aplicação, os *logs* do Apache do servidor Linux deverão ser explorados e verificar se ainda há pedidos que estejam a ser feitos através do servidor Linux, e assim conseguir identificar quais os utilizadores que ainda utilizam a primeira versão da aplicação.

Passo 17: Notificar novamente os utilizadores que não atualizaram a sua versão da aplicação

Os utilizadores que sejam identificados ainda a utilizar a versão 1 da aplicação, durante a semana limite de migração da aplicação, deverão ser notificados novamente, no último dia do prazo, para instalarem a nova versão se não deixarão de ter serviço.

Passo 18: Descontinuar aplicação 1

Nesta etapa a aplicação 1 será interrompida, os utilizadores deixarão de ter acesso ao serviço, após uma semana da primeira notificação aos utilizadores.

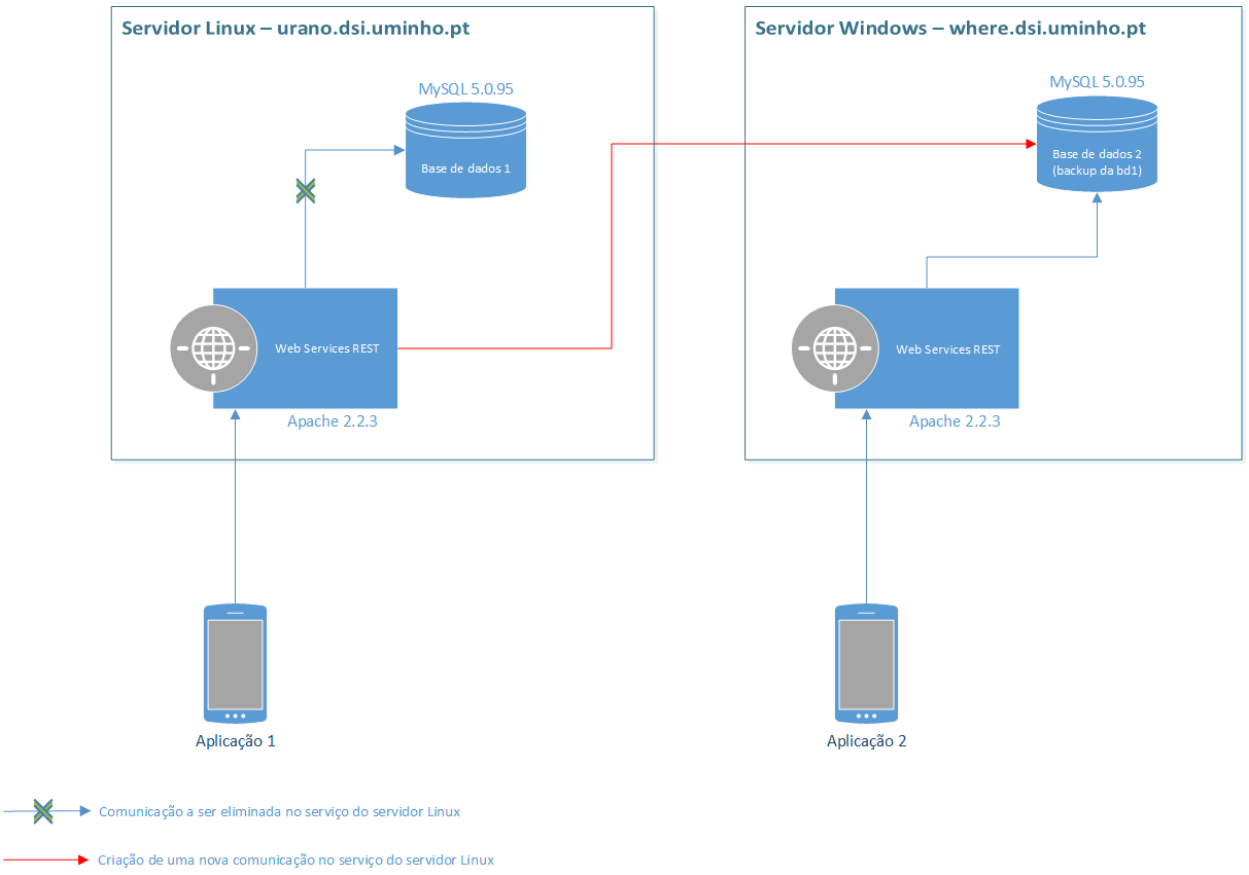


Ilustração 20 - Esquema de migração do servidor Linux para o Servidor Windows.

Na Ilustração 20 está representado um esquema que simboliza todos os passos acima descritos.

Passo 19: Tornar servidor Linux num ambiente de desenvolvimento

Após a descontinuação da aplicação 1, o servidor Linux deverá ser preparado de maneira a tornar-se num ambiente de desenvolvimento:

- Parar o serviço 1
- Mudar os ficheiros da pasta whereatum para whereatumdev
- Alterar o serviço 1 para falar com a BD1
- Ativar o serviço 1

APÊNDICE B

Aplicação WHERE@UM

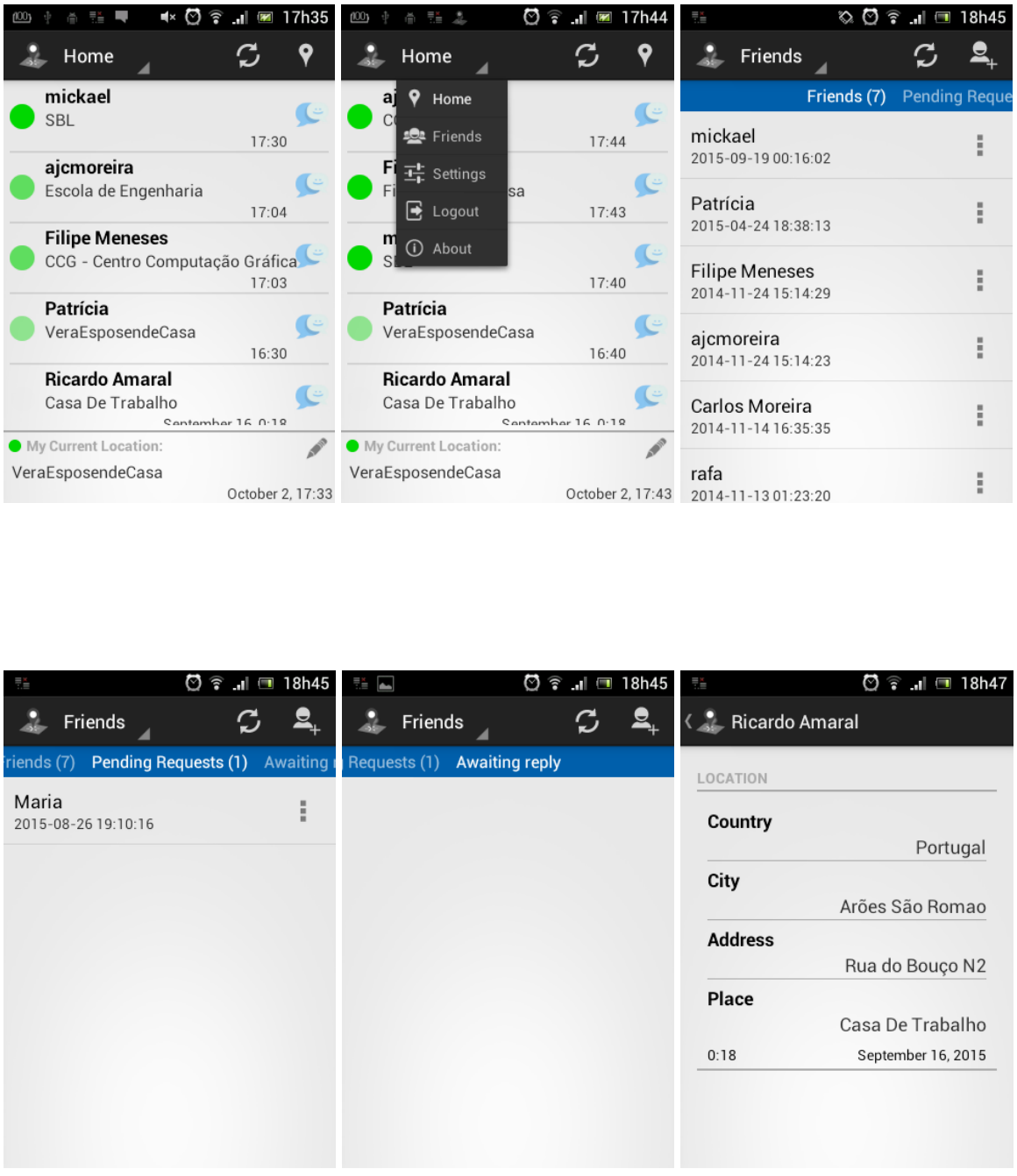


Ilustração 21 - Ecrãs da aplicação (1).

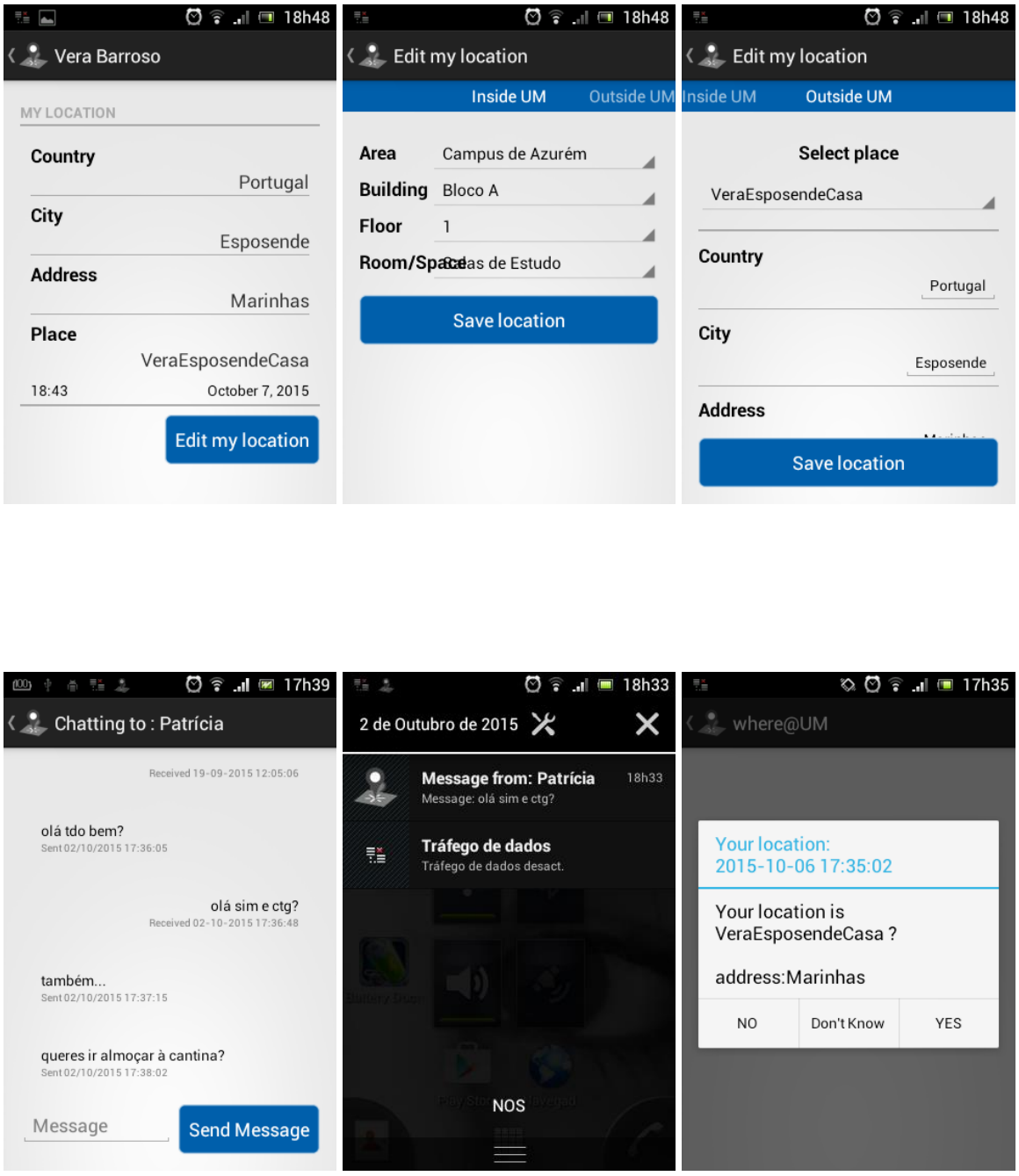


Ilustração 22 - Ecrãs da aplicação (2).

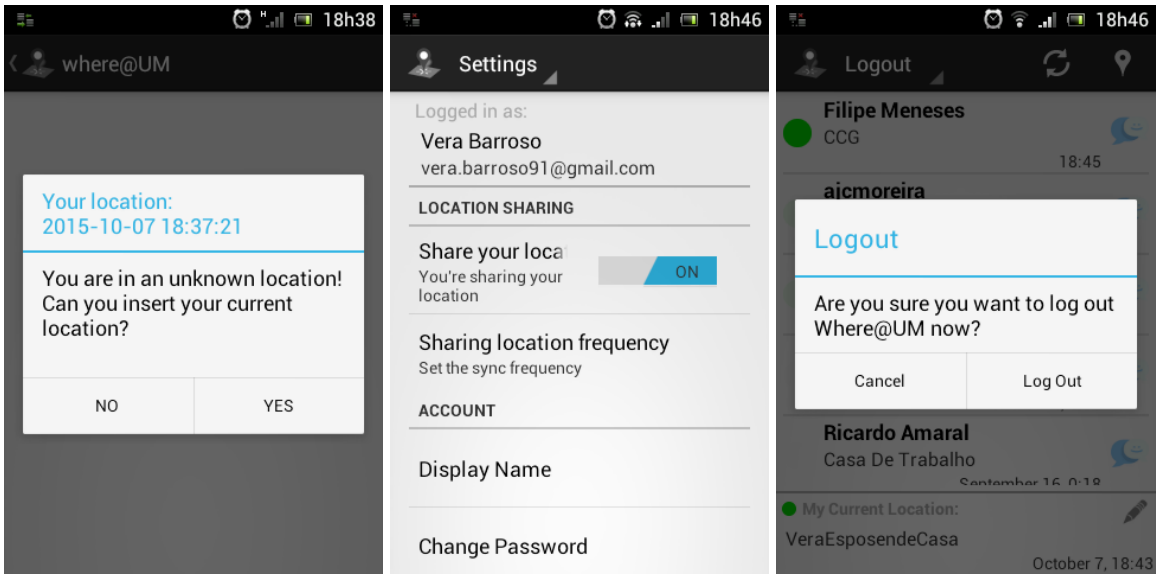


Ilustração 23 - Ecrãs da aplicação (3).

ANEXO A

Instalação e configuração do Nagios

Instalação do Nagios

1. Os comados a seguir foram executados como *root*.

su –l

yum update

2. Para a instalação dos pré-requisitos foram instalados os seguintes pacotes.

yum install httpd php

yum install gcc glibc glibc-comon

yum install gd gd-devel

yum install perl make

3. Criação do grupo nagcmd para permitir comandos externos, enviados pela interface web. Os utilizadores Nagios e Apache devem fazer parte do grupo.

/usr/sbin/groupadd nagcmd

/usr/sbin/usermod -a -G nagcmd nagios

/usr/sbin/usermod -a -G nagcmd apache

4. A seguir será criado uma pasta para armazenar os ficheiros temporariamente

mkdir ~/downloads

cd ~/downloads

5. Neste passo foi feito o download do Nagios e dos plugins. Versões Nagios 4.0.4 e Plugins 2.0.3

wget http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-4.0.4.tar.gz

wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz

6. Compilação e instalação do Nagios

Comandos para extrair

tar xzf nagios-4.0.4.tar.gz

cd nagios-4.0.4.tar.gz

Comandos para compilar e instalar

./configure –with-command-group=nagcmd

make all

make install

make install-init

make install-config

make install-commandmode

7. Configuração da interface web

make install-webconf

Criação do utilizador nagiosadmin com password nagiosadmin

htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

Reiniciar o apache

service httpd restart

Compilação e instalação dos plugins

cd ~/downloads

tar xzf nagios-plugins-1.4.11.tar.gz

cd nagios-plugins-1.4.11

./configure –with-nagios-user=nagios –with-nagios-group=nagios

make

make install

8. Iniciar o Nagios

Configuração do Nagios, para que este inicie automaticamente quando o sistema iniciar:

chkconfig –add nagios

chkconfig nagios on

chkconfig httpd on

Verificar o arquivo de configuração:

/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

service nagios start

9. Modificar as configurações SELinux

O CentOS vem com o SELinux (Security Enhanced Linux) instalado e configurado no modo "Enforcing" por padrão. Isso pode resultar num "*Internal Server Error*" (Erro Interno do Servidor). Assim esse modo foi alterado com o comando:

setenforce 0

Alteração permanente, alteração do ficheiro config e deixar a linha SELINUX=disabled

gedit /etc/selinux/config

10. Configuração do Firewall

Libertar a porta 80 no firewall

/sbin/iptables -I INPUT -p tcp –dport 80 -j ACCEPT

/etc/init.d/iptables save

/etc/init.d/iptables restart

11. Para aceder à Interface Web foi usado o seguinte endereço no *browser*.

http://localhost/nagios/

Configuração do Nagios

1. Criação da página php para fazer pedido à base de dados, check_service.php (este ficheiro foi escrito numa máquina Linux, para não dar problemas). Este ficheiro deve na máquina urano.dsi.uminho.pt/whereatumdev.

```php
<?php
	$connect = new mysqli('127.0.0.1','a58666','44574', 'whereatum');
	if($connect -> connect_error){
		print_r("null");
	}

	$query="SELECT * FROM users";
	$res= $connect->query($query);
	$r=$res->num_rows;
	if($res->num_rows > 0){
		print_r($r);
	}
	else{
		print_r("null");
	}

?>
```

2. Criação do script específico check_service_nagios.php para a monitorização do serviço, este ficheiro comunicará com o ficheiro php da máquina urano, e retornará a resposta dessa máquina ao Nagios.

```
#!/usr/bin/php
<?php

$dados=http_build_query(array(
	''=>''
	''=>''
));

$contexto=stream_context_create(array(
	'http'=>array(
		'method'=>'POST',
		'content'=>$dados,
		'header'=>"Content-type: application/x-www-form-urlencoded\r\n" . "Content-Length: " . strlen($dados) . "\r\n",
	)
));

$resposta=file_get_contents('http://urano.dsi.uminho.pt/whereatumdev/check_service.php', null, $contexto);

$resposta2=(int)$resposta;


if(($resposta2!=null) || ($resposta2 !=0)){
	echo ($resposta2);
	echo ("	");
	echo "OK";
	exit(0);
}
else{
	echo "CRITICAL";
	exit(2);

}
?>
```

3. Colocação do script check_service_nagios.php na seguinte localização /usr/local/nagios/libexec
4. Criação da página notification.php, que verifica o número de utilizadores ativos do dia em que é executado.

```
#!/usr/bin/php
<?php

date_default_timezone_set('GMT');
$data_hora=date('Y-m-d H:i:s');
$data_inicio=date('Y-m-d 00:00:00');


$connect=new mysqli('127.0.0.1', 'a58666', '44574', 'whereatum');

if($connect->connect_error){
      print_r("null");
}

$sql="SELECT idUser FROM fingerprintsHistory WHERE serverTimestamp >='$data_inicio' AND serverTimestamp <=
'$data_hora' GROUP BY idUser";
$result=$connect->query($sql);


while($linha=$result->fetch_array()){
      $linha['idUser'];
}

echo $result->num_rows;

?>
```

5. Criação do script específico check_service_nagios2.php para a monitorização do serviço, este ficheiro comunicará com o ficheiro notification.php da máquina urano, e retornará a resposta dessa máquina ao Nagios.

```
#!/usr/bin/php
<?php

$dados=http_build_query(array(
      ''=>''
      ''=>''
));

$contexto=stream_context_create(array(
      'http'=>array(
            'method'=>'POST',
            'content'=>$dados,
            'header'=>"Content-type: application/x-www-form-urlencoded\r\n" . "Content-Length: " . strlen($dados)
. "\r\n",
      )
));

$resposta=file_get_contents('http://urano.dsi.uminho.pt/whereatumdev/notification.php', null, $contexto);

$resposta2=(int)$resposta;

if(($resposta2!=null) || ($resposta2 !=0)){
      echo ($resposta2);
      echo ("     ");
      echo "OK";
      exit(1);
}
else{
      echo "CRITICAL";
      exit(2);

}
?>
```

6. Colocação do script check_service_nagios2.php na seguinte localização /usr/local/nagios/libexec
7. Dar permissões ao ficheiro check_service.php

cd /usr/local/nagios/libexec

chown nagios:nagcmd check_service.php

chmod +x check_service.php

8. Definição do comando em commands.cfg

cd usr/local/nagios/etc/objects

gedit commands.cfg

```
# 'check_service_nagios' command definition
define command{
        command_name check_service_nagios
        command_line php $USER1$/check_service_nagios.php
        }


# 'notification' command definition
define command{
        command_name notification
        command_line php $USER1$/check_service_nagios2.php
        }
```

9. Definição do *host* e do serviço no ficheiro, host.cfg

cd usr/local/nagios/etc/objects

gedit host.cfg

```
define host{
        use linux-server;
        host_name urano;
        alias urano;
        address urano.dsi.uminho.pt;
}


define service{
        use     generic-service ;
        host_name       urano
        service_description     CHECK_SERVICE
        check_command   check_service_nagios
        check_interval  5;
        register        1
        active_checks_enabled   1
        retry_interval  1;
        max_check_attempts      4;
        contact_groups  admins;
}

define service{
        use     generic-service ;
        host_name       urano
        service_description     NOTIFICATION
        check_command   notification
        check_period    HorarioNotificacoes
        check_interval  720;
        notification_interval   720
        contact_groups  admins;
}
```

10. Definição dos contactos no ficheiro contacts.cfg

Introduzidos os vários *e-mails* a serem avisados.

```
define contact{
    contact_name    nagiosadmin;
    use generic-contact;
    alias   Nagios Admin;
    email   vera.barroso91@gmail.com
    }

define contact{
    contact_name    nagiosadmin2;
    use generic-contact;
    alias   Nagios Admin;
    email   meneses@dsi.uminho.pt
    }

define contact{
    contact_name    nagiosadmin3;
    use generic-contact;
    alias   Nagios Admin;
    email   adriano@dsi.uminho.pt
    }
```

Definição dos vários membros.

```
define contactgroup{
    contactgroup_name   admins
    alias   Nagios Administrators
    members nagiosadmin, nagiosadmin2, nagiosadmin3
    }
```

ANEXO B

Instalação do Thruk

wget http://download.thruk.org/pkg/v1.88-2/rhel7/x86_64/thruk-1.88-2.rhel7.x86_64.rpm

yum localinstall *thruk-1.88-2.rhel7.x86_64.rpm*

http://127.0.0.1/thruk

Username thrukadmin

Password thrukadmin

Instalar mk-livestatus

Instalar os pré-requisitos C++, libc6-dev and libstdc++6-dev.

yum install rsync graphviz php-mbstring php-gd php-pdo php gcc-c++ make libc6-dev libstdc++6-dev

Download do mk-livestatus

wget http://mathias-kettner.com/download/mk-livestatus-1.2.6b12.tar.gz

tar xvfz mk-livestatus-1.2.6b12.tar.gz

Compilar e instalar o mk-livestatus

cd mk-livestatus-1.2.6b12

./configure –with-nagios4

make -j 8

make install

Depois de instalado com sucesso, livestatus.o foi criado em /usr/local/lib/mk-livestatus/ e unixcat foi criado em /usr/local/bin.

Alterar ficheiro de configuração Nagios, em /usr/local/nagios/etc/nagios.cfg

broker_module=/usr/local/lib/mk-livestatus/livestatus.o /usr/local/nagios/var/rw/live

debug=1

event_broker_options=-1

touch /usr/local/nagios/var/rw/live

chmod 0660 /usr/local/nagios/var/rw/live

systemctl restart nagios

Verificar output mk-livestatus

echo 'GET hosts' | unixcat /usr/local/nagios/var/rw/live

Alteração do ficheiro do Apache

(pois houve erro do Apache)

gedit /etc/httpd/conf/httpd.conf

Alterado:

<IfModule dir_module>

DirectoryIndex index.html

</ IfModule>

Por:

<IfModule dir_module>

DirectoryIndex index.html index.php

</ IfModule>

Reiniciação do Apache

systemctl restart httpd

Erro Permissão negada

Edição do ficheiro /usr/local/nagios/etc/cgi.cfg alterar 'authorized_for_configuration_information' que está definido para ser 'nagiosadmin' para 'authorized_for_configuration_information = *'.

/etc/thruk/cgi.cfg

Configuração do ficheiro thruk_local.conf

Alterar o ficheiro no caminho /etc/thruk/thruk_local.conf.

```
use_frames          = 1
statusmap_default_type=circle
<Component Thruk::Backend>
    <peer>
        name   = Core
        type   = livestatus
        <options>
            peer          = /usr/local/nagios/var/rw/live
            resource_file = /usr/local/nagios/etc/resource.cfg
       </options>
       <configtool>
            core_conf      = /usr/local/nagios/etc/nagios.cfg
            obj_check_cmd  = /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
            obj_reload_cmd = /etc/rc.d/init.d/nagios reload
       </configtool>
    </peer>
</Component>
```