Sistemas de Computação Móvel e Ubíqua (2016/17)

Home Security System

Ricardo Ribeiro, nº 42754 Vanessa Lopes, nº 42708

1. Introdução

O projeto desenvolvido assenta num sistema de segurança doméstico que deteta a entrada de um estranho em casa.

O dispositivo deverá ser colocado numa zona a definir do lado de dentro da habitação e junto à porta principal da mesma. Para efeito do mesmo inclui dois sensores ultrassónicos, que detetam a passagem de uma pessoa na distancia de alcance do sensor que comunicam a variação ao Arduíno, o qual envia um sinal de alerta pelo servidor para a *app* a informar os utilizadores que alguém entrou ou saiu de casa.

A par deste sensor é incluído um sensor de *RFID* que permite autenticar a pessoa que está a passar pela entrada principal da casa. A comunicação com este último sensor poderá ser feita através de um cartão *RFID* ou de um sensor *RFID* colocado nas chaves e que interagem com o que está integrado no sistema.

Desta forma, com os anteriores dispositivos integrados no Arduíno é possível, além de detetar qualquer passagem pela zona escolhida, a creditação do individuo que o está a fazer. Este processo será possível, seja através da app quando o utilizador entra na mesma rede do Arduíno, seja através do sensor *RFID* que cobre os casos em que o utilizador não consegue usar a app.

2. Visão Geral

2.1. Arquitetura do sistema

A arquitetura do sistema seguirá um modelo cliente-servidor em que o cliente será implementado na aplicação mobile e o servidor na parte ubíqua, sendo este servidor um *WifiRestServer*.

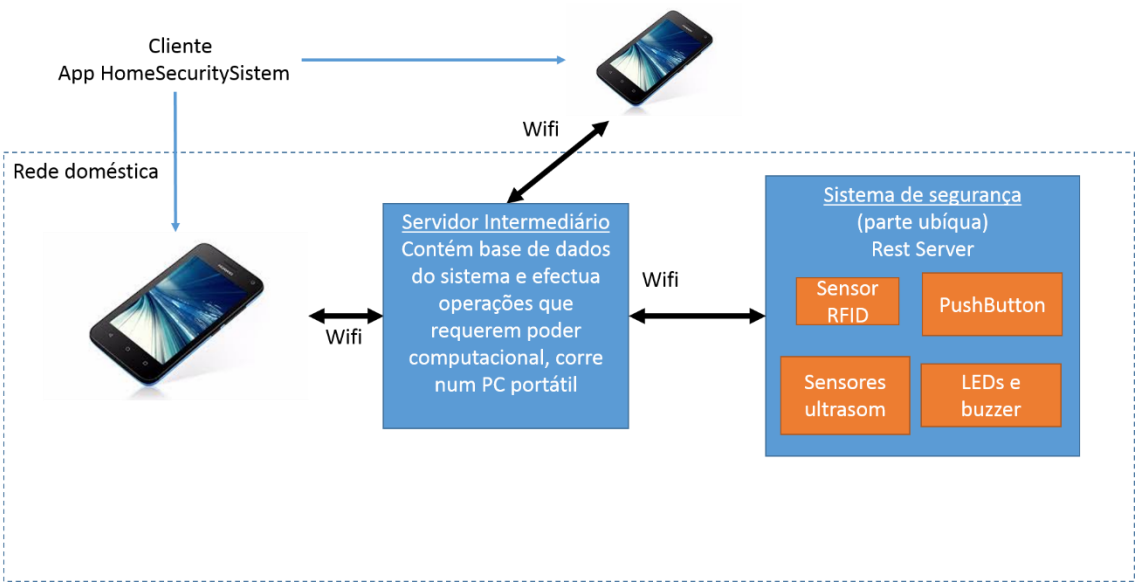

Figure 1: Sistema desenvolvido

A interação do sistema com o exterior decorre através de sensores conectados à placa de Arduíno, mais especificamente 1 *Sensor* de *RFID*, 2 Sensores de Ultrasom e 1 *PushButton*, que recebem inputs do exterior sendo que, por outro lado, estão presentes 2 *LEDs* e 1 *Buzzer* que se manifestam face a esses inputs recebidos.

O *firmware* é responsável pela gestão dos dados recebidos do meio envolvente e por retornar o devido output para as componentes atuadoras do sistema. Nesta perspetiva, foi necessário recorrer a uma entidade intermédia, a aplicação, que permite a interação dos utilizadores que estão autenticados no sistema e que responde a pedidos do servidor e manda as respetivas respostas aos mesmos.

3. Servidor intermediário entre a Aplicação móvel e o sistema ubiquo

Para efeitos de escalabilidade, adicionou-se à arquitetura do sistema um servidor intermédiario que tem a responsabilidade de guardar os dados (TAGs de cartões, MAC adress de telemóveis e entradas e saidas, assim como difundir os eventos gerados pelos sensores às suas apps clientes e sendo este servidor cliente da parte ubiqua, a existencia deste servidor deve-se a permitir a escalabilidade do sistema, assim como poder garantir o poder computacional que o sistema precisa sem transpor esse factor para as entidades extremas, quer seja a parte ubiqua ou a aplicação android.

Numa primeira fase o cliente deverá autenticar-se com a aplicação ou mediante um cartão RFID que abrange casos excecionais, em que o utilizador não possui dispositivo móvel que garanta a sua autenticação e surjam casos de falsos positivos a distorcer a situação.

4. Aplicação Android

A aplicação para android conforme especificado na parte da arquitetura consistirá na implementação do cliente do sistema.

De forma a suportar o seu correto funcionamento e a apresentação dos utilizadores foi necessário encontrar uma solução que comunicasse com o servidor a fim de pedir as listas a apresentar na *App* mas, ao mesmo tempo, não sobrecarregasse o servidor com pedidos constantes e que ainda assim recebesse todas as atualizações do utilizador quando algum membro fosse adicionado ao sistema ou face a uma entrada ou saída de casa.

4.1. Serviço de conexão com o servidor

Para suporte de pedidos e respostas foi necessário estabelecer um serviço que corresse continuamente e cujas mensagens fossem trocadas em *REST* e mediante a serialização dos dados para objetos *JSON*. Por sua vez, esse serviço precisou de ser acedido pela atividade principal da aplicação para distribuição dos dados às restantes

componentes da *App*. De forma a responder às várias necessidades foram procuradas as soluções que mais se enquadrassem.

4.1.1. *Volley*

Volley é uma biblioteca *HTTP* que estabelece ligação à rede entre uma aplicação *Android* e o servidor. A sua grande vantagem está associada à sua gestão dos pedidos ao servidor. Quando o cliente pretende executar um pedido ao mesmo, envia o devido pedido especificado.

Cada um dos pedidos executados pela aplicação é colocado numa *requestQueue*, sendo cada um executado de cada vez. Como a biblioteca usa um sistema de cache, os pedidos apenas são efetivamente feitos ao servidor caso a informação não esteja contida em cache – *cache miss* – sendo assim cada qual encaminhado como um simples pedido HTTP. Caso contrário, na presença dos dados desejados – *cache hit* – os mesmos são retornados diretamente à aplicação sem recorrer ao servidor. Desta forma, além de requerer uma implementação bastante simples, o *Volley* diminui a carga do servidor que não precisa de atender tantos pedidos, ao mesmo tempo que poupa recursos à aplicação, recursos esses escassos.

De notar que a biblioteca em questão apenas permite tratar as situações de pedidos ao servidor. Casos como a propagação das atualizações do servidor aos diversos clientes não pode ser tratada com recurso ao mesmo.

4.1.2. *PubNub*

Para tratar dos dados que o servidor precisa de mandar aos diversos clientes, que todos os utilizadores estariam interessados em receber, faria sentido que a aplicação implementasse uma estratégia *Publish*/*Subscribe*, com o servidor a publicar as atualizações das listas em específicos canais, de interesse a todos os clientes que estariam desta a desempenhar um papel de subscritores desses canais, recebendo as respetivas listas atualizadas. Desta forma, recorreu-se à *messaging* API do *PubNub* que utiliza um modelo de *Publish/Subscribe* para difusão de dados em tempo real e sinalização de

dispositivos que permitem estabelecer e manter conexões persistentes com qualquer dispositivo e enviar dados para públicos globais em menos de 1/4 de segundo. Considerou-se por isso adequado o seu uso no projeto.

4.2. Estrutura da Aplicação

Como foi referido anteriormente, recorreu-se a um serviço que comunicasse com o servidor, de forma a poder gerir a troca de dados entre a app e o servidor e uso das funcionalidades seguidamente descritas.

4.2.1. Funcionalidades:

- Registo de habitantes (*Settings*).
- Subscrever ao servidor (*Settings*).
- Acionar o alarme(*Settings*).
- Alternar modo silencioso e modo não silencioso
- Listagem de:
    - Residentes (*Members*).
    - Residentes em casa (*At home*).
    - Entradas e saídas dos residentes da casa (*History*).
- Autenticação através do sensor GPS do telemóvel ou por presença na mesma rede (*Authenticate*).

Através da *ActivityMainMenu* é criada uma instancia deste serviço e armazenados os dados atualizados. Cada um dos fragmentos da atividade, apresentados como cada uma das *Tabs* da página inicial, recorre a estes dados para apresentarem as diversas listas de interesse ao utilizador da aplicação.

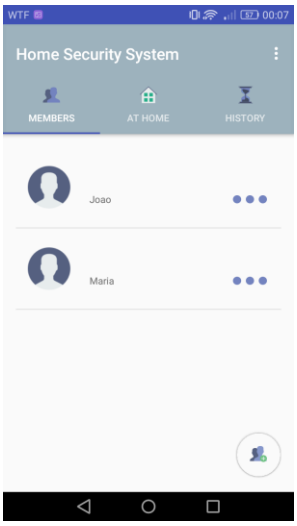

Figure 2: Tab com lista de membros.

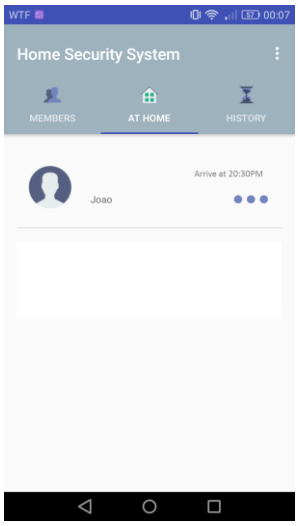

Figure 3: Tab com lista de membros em casa.

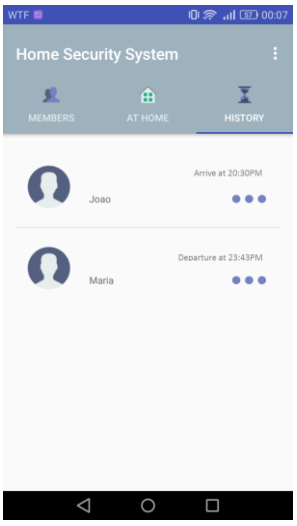

Figure 4: Tab com histórico de todos os membros.

Por sua vez são também disponibilizadas as funções de adicionar e editar um membro. A interface em ambos os casos é bastante semelhante. Além dos campos de Nome, Email e Telefone tem também um campo de *RFID* que associa uma *Tag* ao utilizador distinguindo-o unicamente. Um dos pormenores que se considerou fundamental incluir, foi a possibilidade de personalizar a *Tag* em questão de acordo com as preferências do utilizador. Assim, quando o utilizador seleciona o botão *NEXT* na figura à esquerda ou o botão X na figura à direita a aplicação é redireciona para a página de edição da *Tag* do cartão *RFID*.

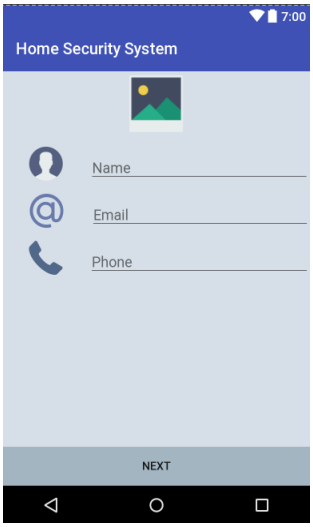

Figure 5: Página de Adicionar Membro

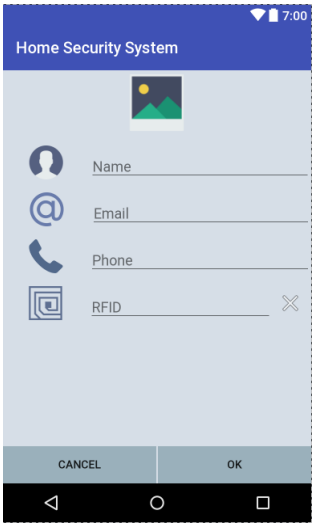

Figure 6 Página de Editar Membro:

A página referente ao RFID é apresentada seguidamente em 2 estados diferentes que expõem dois estados do botão de Escrita/Leitura, isto é, quando indicada a informação *Place the card near the phone TO WRITE*, é sugerida a escrita de uma *Tag* à escolha para identificação do próprio. Quando indicada a *Tag* e o cartão é aproximado do telefone a mesma é guardada automaticamente no cartão de *RFID*. De notar que para tal funcionar, o sensor do telefone para identificação de cartões *NFC* (que inclui *RFID*), deve estar selecionado.

Caso premido o botão TO WRITE, o seu valor muda para TO READ e o utilizador pode aproximar o cartão para ler o conteúdo do mesmo (figura 5).

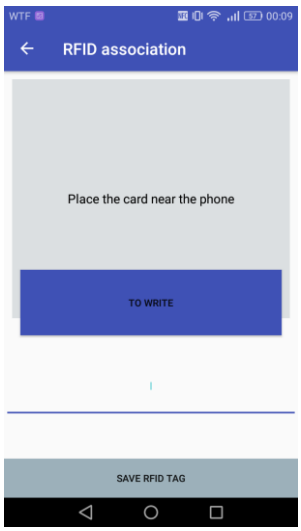

Figure 7: Página de associação de RFID no estado de Escrita para cartão

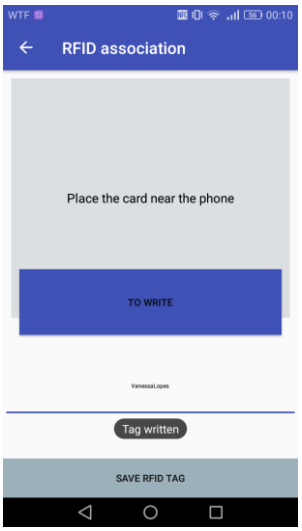

Figure 8: Notificação de escrita para cartão.

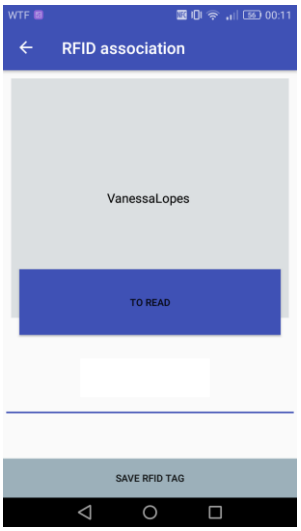

Figure 9: Página de associação de RFID no estado de Leitura do cartão.

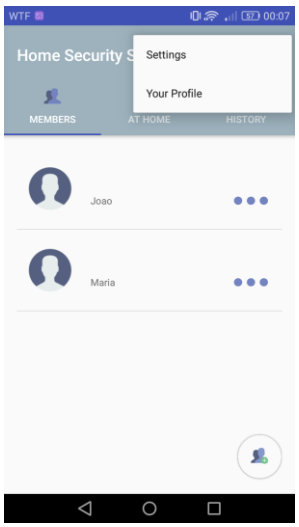

Figure 10: Menu para seleção das definições da aplicação ou da edição do perfil do dono do dispositivo

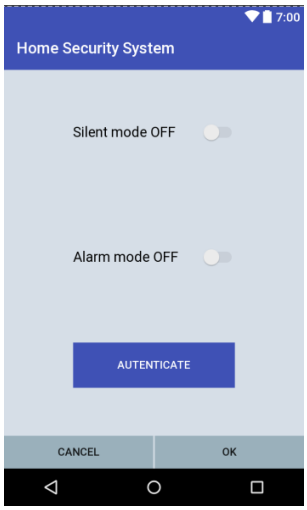

Figure 11: Página das definições do sistema

A página de definições permite que o utilizador configure uma das configurações do Arduíno, mais especificamente o modo silencioso, que inibe o *buzzer* de fazer ruído. Por outro lado, permite ativar o modo alarme, que permite acionar o alarme. É ainda permitido ao utilizador autenticar-se através da aplicação, premindo o botão *AUTHENTICATE*. Isto faz com que seja enviado ao servidor a *Tag* do membro a autenticar.

Quando o utilizador prime a opção OK, todas as definições aqui estabelecidas são enviadas ao servidor sob a forma de objeto JSON.

A página de perfil do dono da aplicação é semelhante à figura 6 que apresenta a página de definições de um utilizador. Assim, é também possível que o mesmo altera a sua *Tag*, entre outros dados.

Todo o sistema apresentado funciona para a família da habitação seja para os membros que têm a aplicação, como para os que, mesmo não a tendo, conseguem usar o cartão *RFID* para se autenticar. Destaca-se por isso a possibilidade de ler e escrever de e para o cartão *RFID* e a possibilidade de poder criar um novo membro de qualquer dispositivo com a aplicação. Torna-se assim flexível e eficiente a associação e autenticação dos utilizadores do sistema.

5. Sistema Ubíquo

A componente ubiqua é constituida por um microprocessador *Arduino UNO Wifi* além de um conjunto de atuadores e sensores que permitem a interação do utilizador com o sistema:

5.1. Sensores integrados[1]:

- 2 sensores ultrassónicos – Sensores de distancia para detetar a presença de uma pessoa através da variação da distancia emitida por uma componente do sensor e recebida pela outra componente do mesmo. O uso de dois sensores serve para distinguir a entrada da saída da habitação.
- 1 sensor de *RFID* – Sensor para autenticação adicional do utilizador, através de componentes extra que acompanham o indivíduo a autenticar quando este atravessa o sistema – através de uma componente associada ao porta-chaves ou de um cartão de *RFID*.

5.2. Atuadores integrados:

- *Pushbutton* que sendo pressionado, emite alerta para os restantes membos da habitação e ativa os atuadores (*LED*s e *Buzzer*), para evitar falsos positivos, é necessário passar uma tag que o sistema reconheça para ativar o alarme.
- *2 LEDs*, um que emite luz verde e outro para emitir luz vermelha mediante uma entrada autenticada ou não, respetivamente., também servem para efeitos de debug visual para distinguir uma entrada de uma saida, no momento em que é feita uma passagem pelos sensores ultrassom.
- *Buzzer* que emite som de alerta na eventualidade de alguem efectuar uma passagem pelos sensores, foi implementado um modo silencioso que inibe o buzzer de emitir som.

[1] As emissões de sinais de ambos os sensores convergem para a conclusão de que a pessoa não é meramente um estranho.

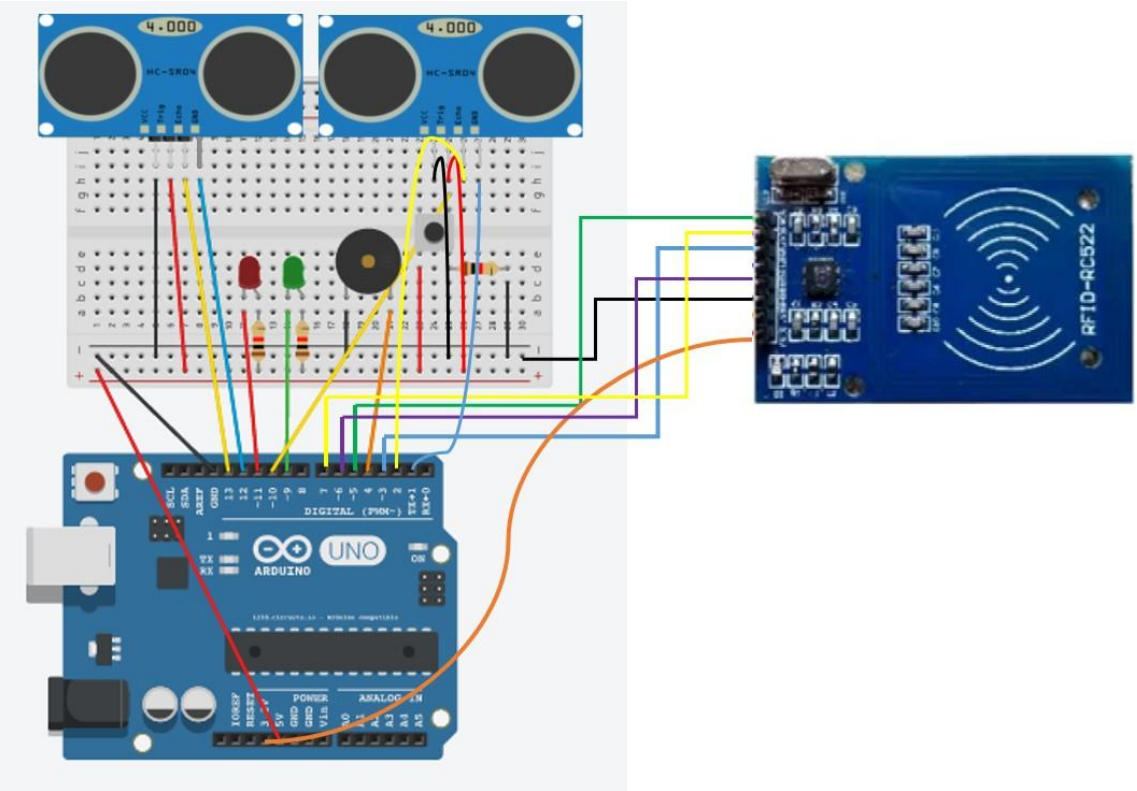

Figure 12 Sistema ubíquo com os devidos sensores e atuadores integrados com a placa de Arduíno

6. Experiencias efetuadas

Para teste da aplicação foram verificados os resultados registados pelos sensores e atuadores do sistema. Foram testadas passagens como entradas ou como saídas, seguidas ou não de passagem de cartão *RFID*. Nos casos em que o cartão não foi usado, o sistema dispara o alarme e os restantes atuadores.

Verificou-se também que o servidor comunicava na perfeição com o Arduíno, sendo trocados dados entre ambos. Quando autenticação por cartão, verificou-se que o servidor guarda corretamente os registos em cada ficheiro associado.

A *app* foi testada e confirmada a adição e edição de utilizadores, de *Tags* personalizada e todas as funcionalidades acima descritas.

7. Conclusões

As componentes do sistema, mais que permitirem o funcionamento do mesmo, abrangem diversos problemas que possam surgir, uma vez que conferem diferentes abordagens ao mesmo. Isto porque, aplicado ao mundo real, seria necessário englobar casos em que um dos membros da família não tivesse a aplicação para autenticação. Desta forma, durante a elaboração da solução do projeto considerou-se que a efetividade da solução passaria por abordar todos os casos possíveis e que, por isso, seria necessário encontrar funcionalidades que não sejam exclusivas à app nem ao Arduíno e, ao mesmo tempo, práticas e funcionais.

Embora a deteção de passagem de uma pessoa pela entrada seja fundamental e indispensável, o uso de um leitor *RFID* não o é, uma vez que através da app é possível autenticar a pessoa. Todavia, o seu uso, além de prático, veio a mostrar-se uma vantagem com a possibilidade de escrita de *Tags* personalizadas e a associação direta destas a cada utilizador. Com isto, o sistema tornou-se bastante mais eficiente e funcional.

O mesmo acontece com a possibilidade de disparar alarme. A primeira estratégia passava apenas por disparar o mesmo quando não autenticado, mas para um sistema mais prático foi adicionada a mesma funcionalidade à aplicação e a um *push button* com esse efeito.

De destacar ainda que a possibilidade de trabalhar com uma panóplia de sensores, um Arduíno e os restantes dispositivos de hardware permitiu explorar potencialidades dos dispositivos a nível *hardware*, mais que o desenvolvimento de *software* habitual, que reunidos, convergem num projeto que faz destacar as potencialidades do mundo de *IoT*.