

**INSTITUTO SUPERIOR TÉCNICO**

**REDES MÓVEIS E SEM FIOS**

**Projeto**

Development of Internet of Things sensor monitoring based on IEEE 802.15.4, MICAz and Android

Grupo 9

75847 – José Dias

76090 – Diogo Ferreira

Prof. António Grilo

22 de Maio de 2016

# Motivação

A *Internet of Things* consiste na interligação de vários sensores à rede de *Internet*. Os dados recebidos de sistemas de baixa potência são enviados e armazenados em servidores permanentemente ligados à rede *(Cloud*) que permitem a mobilidade no acesso por qualquer dispositivo portátil com capacidade de ligação, em particular, dos *smartphones*.

Pretende-se desenvolver um sistema de segurança composto por sensores de movimento que alerte o seu utilizador em caso deteção. No processo, serão aprofundados conhecimentos em redes de computadores, comunicação entre máquinas e funcionamento em rede, aperfeiçoamento e aprendizagem de novas linguagens de programação. Prevê-se um benefício pedagógico pela junção de temas aprendidos na área de Telecomunicações e Redes de Comunicação e em especial pela verificação do objetivo de cada camada do modelo OSI no funcionamento de um sistema de comunicação.

# Arquitetura

O sistema a implementar consiste na ligação de um conjunto de sensores de movimento à rede de *Internet*. O sistema a implementar pode-se dividir em 3 etapas: aquisição, armazenamento e acesso.

Na aquisição, os dados serão recolhidos e enviados para o servidor via Internet. Será usado o sistema MICAz e MIB520. Este sistema consiste em várias placas com comunicação via rádio (norma IEEE 802.15.4) que correm o sistema operativo orientado a eventos *TinyOS*. Define-se uma para receber os dados e as restantes serão ligadas uma a cada sensor. Os dados recebidos pela estação de base são passados à placa MIB520 através de um conector de expansão de 51 pinos e para o computador via USB. O computador corre o sistema operativo Linux, versão *Ubuntu* ou *Mint*. As tramas são processadas de acordo com a sua origem através de um identificador do nó, e é colocado um carimbo temporal através do computador. Só serão transmitidos dados entre as placas quando à atividade num dos sensores.

Os dados são armazenados num servidor *Web*, implementado em PHP. Este comunicará com o computador do qual receberá a informação relativa aos sensores de movimento.

O acesso aos dados será implementado a partir de uma aplicação que vai ser instalada num *smartphone* ou *tablet* que corra o sistema operativo *Android*, onde será possível verificar o histórico dos alarmes anteriores ou ouvir com pouco atraso um som caso um dos sensores seja ativado nesse momento. Este som poderá ser modificado consoante a preferência do utilizador.

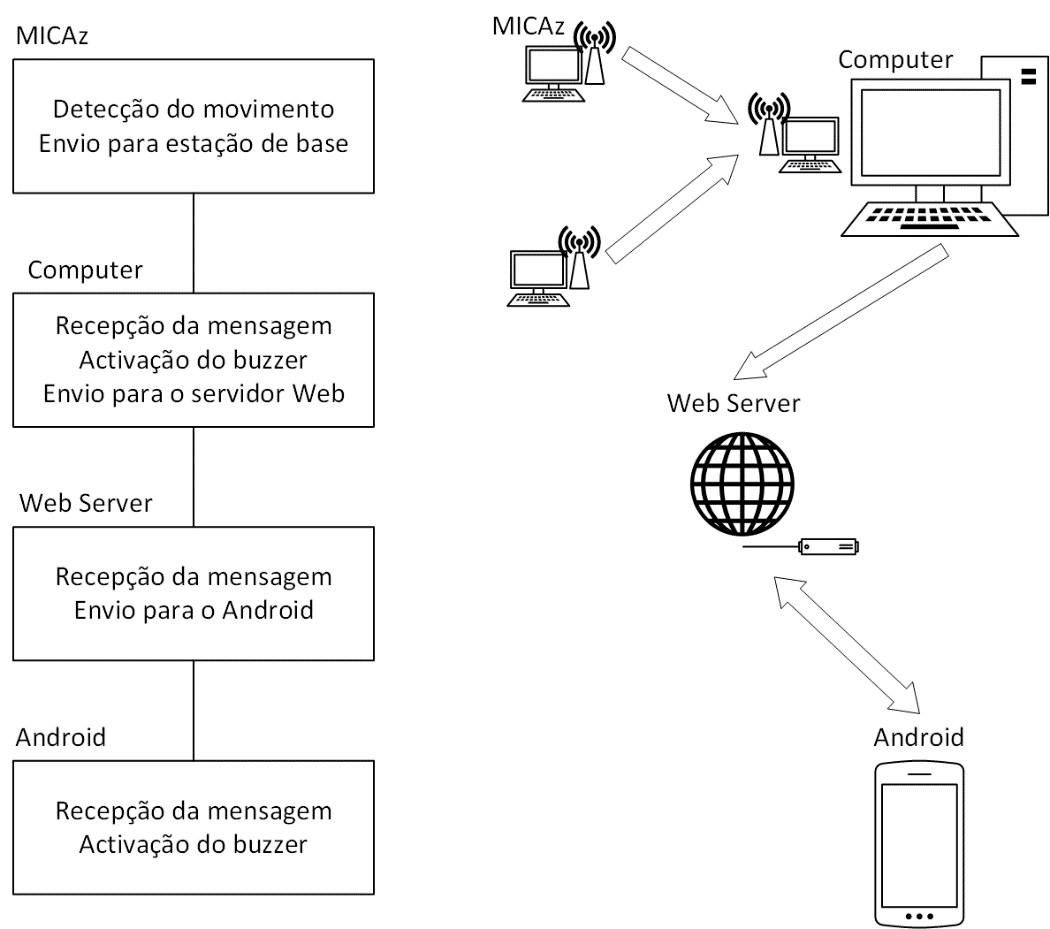


Figura 1 - Arquitetura high level do sistema e respetivo fluxograma.

# Objetivos iniciais

* Familiarização com a linguagem de programação *nesC* orientada a eventos e com o sistema operativo *TinyOS* que corre nas placas MICAz e que fazem a ligação entre o sensor e a estação de base.
* Interpretação das tramas enviadas pela estação de base ao computador (via USB) e implementação de *software* para tratamento automático da informação.
* Implementação de um servidor Web, em PHP, com ligação ao computador associado à estação de base e acessível através de um *browser* e da aplicação, onde serão guardados todos os dados.
* Desenvolvimento da aplicação *Android* para consulta em tempo real dos alarmes.

# Trabalho Desenvolvido

## MICAz, MTS310, MIB520

No sistema implementado consideram-se 4 tipos de nós com funções distintas. A estação de base, ligada à placa de comunicação (MIB520) com o computador, os sensores de movimento, as sirene (MTS300) e as sirenes com detetor de som (MTS310). Cada nó é programado de acordo com a sua função e todos os nós têm conhecimento da estrutura das mensagens que trocam via rádio através de um ficheiro \*.h e que será descrito na secção dos procotolos.

Cada sensor tem um parâmetro único que o identifica dentro da sua rede – Node ID. Ao programar cada sensor, este parâmetro foi alterado de sensor para sensor ao começar em 1 e incrementar o seu valor.

Utilizou-se uma interface já desenvolvida pelo INESC, *BinarySensor*, para detetar um evento de existência de movimento. Detetado o movimento, usaram-se as interface de comunicação via rádio disponíveis nos exemplos mais simples do *TinyOS* para comunicar a mensagem aos restantes nós. A mensagem segue com identificação do nó que lhe deu origem e deverá ser recebida por sirenes dentro do raio de comunicação do nó e pela estação de base. A mensagem é ignorada pelos restantes sensores de movimento pois só receberão mensagens da estação de base. Para sirenes escondidas do detetor de movimento, a estação de base difunde uma mensagem de alerta de deteção de movimento e que as fará tocar. Para os nós equipados com a placa MTS310, o detetor de som na faixa dos 4 kHz é usado para detetar outras sirenes a tocar.

Idealmente todas as sirenes teriam também detetor de som, ou seja, todas as placas seriam MTS310. Contudo, devido a um problema de interferência entre o módulo de rádio e o detetor de som não foi possível implementar a opção de configuração do efeito de propagação estando este sempre ativo.

O conjunto das sirenes tem um identificador bem definido (100) e igual entre si para que a opção de ligar ou desligar a sirene seja difundida com este identificador e interpretada por cada uma.

## Estação de Base

A estação de base envia e recebe mensagens através do módulo de rádio e através da porta USB. A sua execução está associada a um programa Java que é responsável por fazer a gestão das mensagens que chegam do servidor PHP central e das mensagens que deverão ser enviadas via rádio pela estação de base.

Assim que é iniciado, o programa requer informação ao servidor PHP acerca do estado da rede e das suas configurações internas. Assim que o pedido é respondido, a informação é extraída da *String* é colocada, byte a byte, no vetor que contém o pacote a ser enviado por USB para a estação de base. A estação de base limita-se a difundir este pacote e a alertar cada nó das suas configurações. A cada identificador de nó diferente é dedicada uma mensagem de configurações.

A gestão de pedidos é realizada através de duas threads: uma para enviar pedidos periódicos ao servidor de 3 em 3 segundos e receber as configurações dos nós, e outra para enviar alarmes assíncronos dos sensores.

## Servidor PHP

O servidor PHP é implementado de maneira a receber pedidos e a lidar com vários clientes em simultâneo: Utilizadores (*Android*) ou Estações de Base. Toda a comunicação via Internet é realizada através de *Sockets*, em que as mensagens trocadas são *Strings.*

Quando se executa o servidor este cria uma interface de comunicação bidireccional entre processos (Socket), seguido de um Bind do endereço IP no qual se encontra e do porto desejado e o Listen de modo a receber os clientes. Estes clientes são aceites através do Accept, a cada cliente é realizada uma chamada de sistema (fork) que cria um novo processo e este novo processo é que realiza a manipulação do cliente que acabou de aceitar. A base do servidor é um conjunto de verificações nas mensagens de protocolo, primeiro para decidir se se trata de um cliente Android ou de uma estação de base e posteriormente para decidir que tipo de operação ou query executar na base de dados.

## Base de Dados

A base de dados tem 5 tabelas e a sua construção teve como a escalabilidade de todo o sistema sem que fossem necessárias quaisquer alterações à base de dados.

Na tabela Pessoa (*Person*) é possível guardar os dados de cada utilizador, como o seu E-mail (E-mail), nome (Name), password (Password).

Na tabela PAN é possível guardar o ID de cada PAN (idPan) e correspondente Serial Key (Serial\_key), guardar se esta está ativa (Enable) ou se tem a propagação(Propagation) ou sirene (Buzzer) ativos.

A partir das tabelas anteriores é derivada uma tabela de relação entre elas, que vai associar cada pessoa através do seu e-mail a uma certa PAN definida pelo seu ID, ainda nesta tabela é dada a indicação de Notificação (Notify). Esta indicação é necessária pois vários membros podem aceder à mesma PAN e ser necessário saber quais destes é que já foram notificados pelo seu alarme.

Na tabela Node guarda-se o valor do ID correspondente a cada nó (idNode), se este se encontra ativado (Activated). Devido a existir uma relação entre os nós e a sua PAN e a esta relação ser de muitos para um (Many-to-One) não é criada uma tabela nova mas sim acrescentado um parâmetro à tabela Node com a chave primária correspondente à tabela PAN de modo a identificar a que PAN este nó pertence.

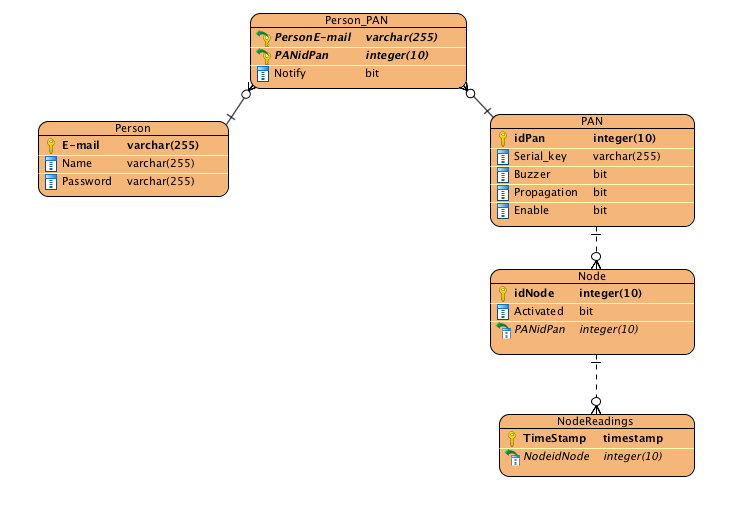
Na tabela NodeReadings guarda-se um Timestamp. Pela mesma explicação apresentada para a tabela Node deve ser também acrescentado um parâmetro à tabela NodeReadings acerca do nó a que está associado (NodeidNode).

Figura 2 Modelo E-R da base de dados.

## Android

No *Android* desenvolveu-se uma página inicial, sobre o qual se deve realizar o login (*Sign In)* ou realizar o registo (*Register*), este é apresentado na figura 3. Este é implementado de modo a só ser possível o acesso remoto aos dispositivos que pertencem ao utilizador registado que os detém. Independentemente da opção escolhida (*Sign* In ou *Register*) deverá ser dado um endereço IP válido para o servidor Note-se que este campo foi apenas inserido devido aos autores não terem acesso a um servidor com endereço IP constante e público. Em caso de implementação no mercado, seria comprado endereço IP externo e constante e a opção não deveria ser acessível ao utilizador.

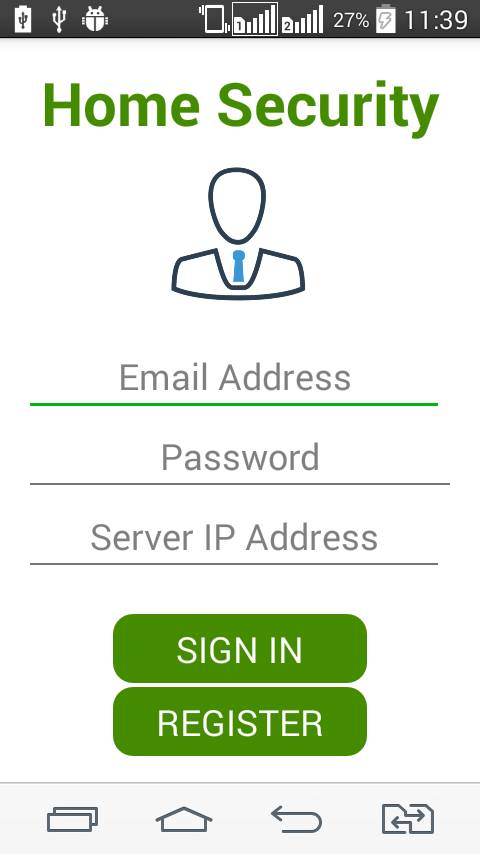
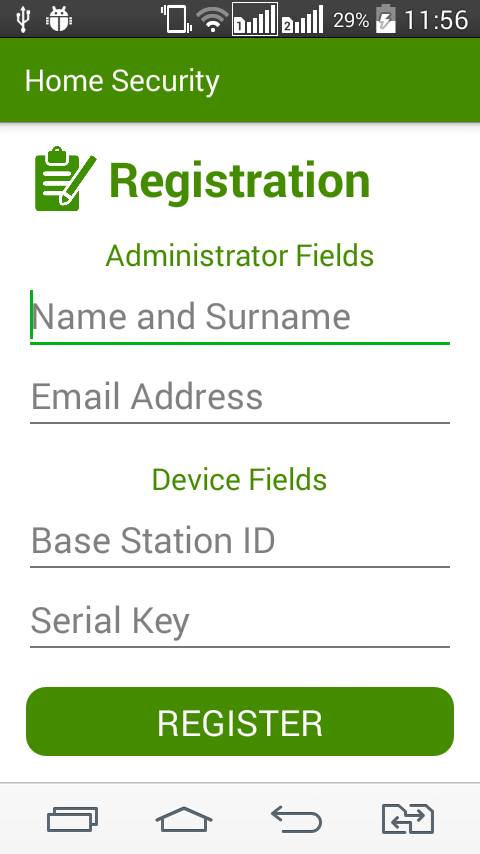


Figura 3 - Página Inicial. Figura 4 - Página de Registo

Caso a opção escolhida seja de registo, a página é alterada - figura 4. O ID da estação de base e a sua *serial key* são conhecidos pelo utilizador, imaginando a analogia com os routers quando são comprados e contêm algures na sua estrutura um selo com estes parâmetros.

Assim que o utilizador prime o botão de registo é verificado se o ID inserido existe, em caso afirmativo, verifica se corresponde à *Serial Key* indicada. Caso um dos parâmetros esteja incorreto é mostrado um aviso com a indicação de qual dos parâmetros está incorreto. Em caso de sucesso o servidor PHP enviará o *mail* com a *password* de acesso à aplicação.

Caso a opção seja de login, envia-se o login para o servidor. Se este for aceite pelo servidor, a aplicação muda para a página *Home* onde se pode iniciar ou terminar pedidos periódicos de 5 segundos ao servidor para receber notificações de novos alarmese aceder à página das configurações (*Settings*) ou à página dos registos (*Records*). Esta página está representada na figura 5. Quando o utilizador recebe uma notificação da aplicação, *o smartphone* vibra e toca um som a avisando o utilizador. Se o utilizador carregar na notificação é redirecionado para a página de *Login* da aplicação para que possa, por exemplo, desligar a sirene.

Caso o *login* esteja incorreto surge um *pop-up* com a informação relativa ao campo que se encontra errado.

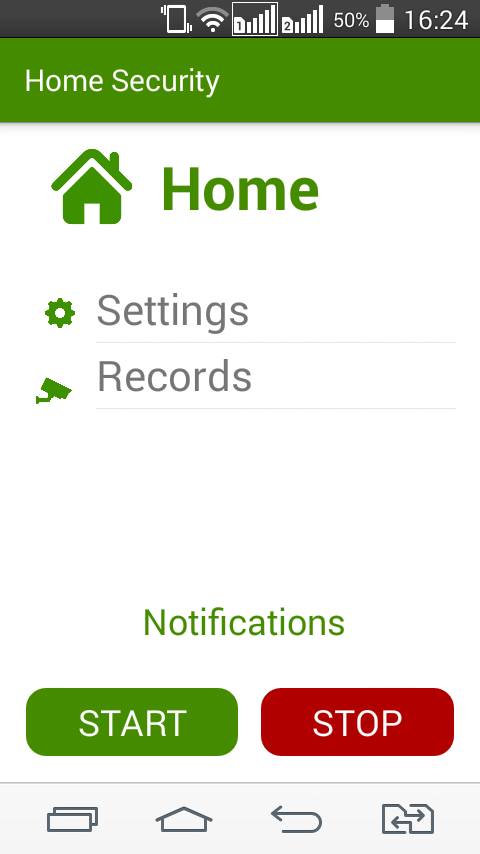


Figura 5 - Página da Casa

Ao entrar nas configurações (*Settings*), é enviado um pedido ao servidor a requerer as configurações dos nós para correcta apresentação da página. Apresentam-se três botões de alternância (*toggle button*) e uma lista dos sensores de movimento associados a uma única PAN. Através dos botões é possível ativar ou desativar o sistema todo (*Enabled*), apenas a sirene (Buzzer), ou um sensor de movimento em específico ao carregar por cima do elemento da lista que se pretende ativar ou desativar e que irá alterar a sua cor de fundo caso esteja activo (Verde) ou descativo (branco).

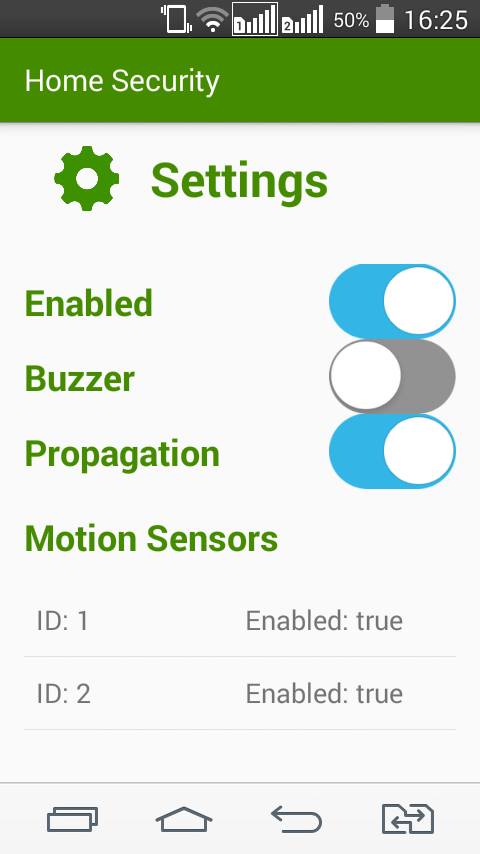
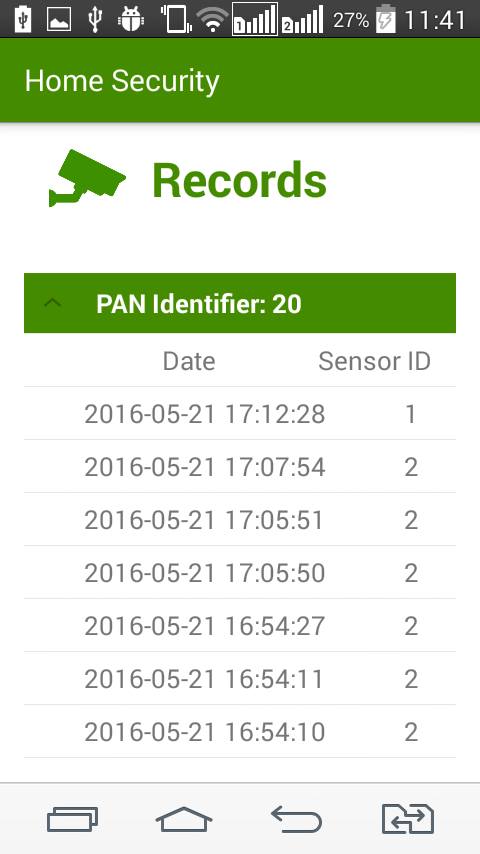


Figura 6 - Página das Configurações. Figura 7 - Página do histórico dos alarmes

Na página de registos (*Records*) é apresentada uma lista dos alarmes que ocorreram no passado, ver exemplo na figura 7 e é indicado num *pop-up* o número total de registos transferidos.

Todo o código da aplicação foi desenvolvido no ambiente *Android Studio*. Utilizou-se uma atividade para cada uma das páginas e um serviço com uma *thread* a correr em segundo plano para realizar os pedidos periódicos. Todas as ligações à rede requerem uma *thread* diferente da *thread* de interface com o utilizador.

**V. Protocolos Utilizados**

**V.I Comunicação da Estação para Base e Servidor**

Na comunicação da Estação de Base para Servidor envia-se BASE no início do pedido de maneira a que o servidor consiga perceber o remetente do pedido, este deve ser precedido de NOTIFICATION ou REQSETS consoante o pedido.

Após um NOTIFICATION surge dois campos com o ID do nó e o ID da PAN.

Depois de um REQSETS deve aparecer o ID da PAN.

**V.II Comunicação do Android para Servidor**

Na comunicação do Android para Servidor envia-se ANDROID no início do pedido de maneira a que o servidor consiga perceber o remetente do pedido, este deve ser precedido de LOGIN, REGISTER, MODIFY, RETRIEVE ou NOTIFICATION consoante o pedido que o Android desejar.

A seguir a um LOGIN deve ser apresentado dois campos com o e-mail e a password do utilizador.

Após um REGISTER deve surgir quatro campos correspondentes ao nome e e-mail do utilizador, ID e serial key da PAN.

Depois do MODIFY aparece quatro campos com o ID da PAN (idPAN), o parâmetro que se pretende alterar (Parameter), o valor que se pretende colocar (Value) e o nó (idNode).

Posteriormente a um RETRIEVE surge o ID da PAN.

Em seguida a um NOTIFICATION aparece um campo com o e-mail do utilizador.

**V.III Comunicação do Servidor para Estação de Base**

**V.IV Comunicação do Servidor para Android**

Na comunicação do servidor para o Android envia-se um OK ou um NOK.

**V.V Comunicação entre MICAz**

**VI. Configuração para envio do e-mail**

**VI. Considerações finais**