UNIVERSIDADE D
COIMBRA

Internet das Coisas

Grupo 1

SIP: Smart Indoor Parking

Autores:

Bruno Miguel Fonseca Ferreira
2014201123

Gustavo Miguel Santos Assunção
2014197707

23 de Junho de 2019

1 Introdução

Este projecto tem como objectivo criar um sistema correspondente a uma garagem inteligente na forma de uma interface autónoma que, de um modo geral, permite a utilizadores verificarem online que lugares de estacionamento estão disponíveis. O sistema reúne também dados estatísticos e processa-os temporalmente, mostrando os diferentes níveis de ocupação a nível diário, semanal e mensal, a diferentes horas. Isto foi planeado com o objectivo em mente de fornecer informação útil à administração de tal garagem e que lhes permitisse uma melhor gestão da mesma em termos de espaço e preçário. Adicionalmente, o sistema deverá também reunir dados de temperatura e humidade com vista a auxiliar a administração a regular a ventilação interna. Como primeira etapa deste projecto, foram definidos os requerimentos a nível tecnológico do trabalho, bem como definida a arquitectura subjacente do projecto que nos permitiu seguir em frente com o seu desenvolvimento.

Na segunda etapa do projecto, o sistema foi de facto implementado. Para tal, e tendo em conta a natureza IoT do projecto, foram utilizadas algumas placas RE-Mote da Zolertia, juntamente com alguns sensores aplicados para a extracção de dados. Considerou-se também o Border Router, que juntamente com o broker a correr na máquina local permitiu enviar dados para a cloud AWS que foi criada exclusivamente para este projecto. Finalmente foi criado um website (disponível em `https://orion.isr.uc.pt/~gustavo.assuncao/iot/index.html`) como aplicação frontend para os utilizadores, onde estes podem consultar as informação que foram já aqui definidas. O website foi optimizado para acessos através de *Google Chrome*.

2 Plano de Trabalhos

De modo a estruturar o desenvolvimento do nosso trabalho, foram seguidos os seguintes passos:

1. Desenvolvimento do sistema de recolha de dados com sensores e módulos Zolertia
2. Envio de dados para e processamento na Cloud
3. Criação do Website front-end, e estruturação da apresentação de dados
4. Testes Finais e Aperfeiçoamento

3 Hardware

De modo a ser possível desenvolver o sistema descrito foram necessários os seguintes elementos de hardware.

- 3 × Placa RE-Mote Zolertia com módulo Zoul
- 4 × Sensor de Proximidade 80cm Infravermelhos Grove
- 1 × Sensor de Temperatura e Humidade Grove
- 1 × Border Router

3.1 Placa RE-Mote Zolertia

Esta componente de hardware funciona como uma plataforma de desenvolvimento na área de Internet das Coisas. A placa em si, para além de todas as ligações que são possíveis de criar com os seus diversos pinos, inclui também exteriormente dois conectores analógicos e um digital de modo a facilitar as ligações a sensores. Juntamente com estas, há também dois botões (*user* e *reset*) úteis na fase de desenvolvimento, e um switch onde pode ser conectada uma antena RF para melhor comunicação com o Border Router do sistema integrante.

Aliado ao facto de que a placa é ultra low power, necessitando apenas de alimentação entre $1uA$ e $150nA$, esta corre também o sistema operativo *Contiki OS*. Este sistema operativo é ideal para o efeito uma vez que foi desenhado para

questões de Internet das Coisas e preparado para lidar exactamente com hardware restringido em termos de memória, energia, poder de computação e largura de bando, como é o caso da placa RE-Mote. Tendo isto em consideração, o Contiki faz uso de um modelo de programação diferente do normal.

3.1.1 Protothreads

Com vista a poder correr na placa RE-Mote, o sistema Contiki utiliza protothreads, abstracções de programação bastante eficientes em termos de memória. Neste modelo, o kernel invoca o protothread de um processo em resposta a algum evento externo ou interno, como por exemplo timers dispararem ou mensagens serem publicadas no caso interno, e também sensores captarem e enviarem alguma informação no caso externo.

Tendo em conta que protothreads são escalonados cooperativamente, um processo Contiki tem periodicamente de reverter controlo de volta ao kernel explicitamente. Os processos podem ainda assim utilizar certas construções de protothreads para se bloquearem à espera de eventos, ao mesmo tempo que revertem controlo de volta ao kernel entre cada invocação sua.

3.2 Sensor de Proximidade

Este componente, o SharpGP2Y0A21YK 80cm Proximity Sensor da Grove, é um sensor analógico multiusos que continuamente efectua leituras de distância e devolve à interface um valor de voltagem correspondente ao valor medido de distância. Uma vez que tem um alcance de entre $10cm$ e $80cm$, e um tempo de resposta de aproximadamente $39ms$ tornou-se um candidato ideal para integração de uma maquete demonstrativa do nosso trabalho.

O sensor é capaz de medir distâncias entre o intervalo indicado através de triangulação. Desta maneira, sendo o sensor composto por um emissor e por um receptor, inicialmente o emissor lança um pulso de luz infravermelha o qual tanto pode embater e ressaltar num objecto próximo, como não encontrar nenhum objecto no seu caminho. Neste último caso, o pulso nunca é reflectido e então a leitura mostra que não há nenhum objecto próximo. Se pelo contrário houver um objecto próximo, o pulso de luz embate neste e é reflectido de volta para o receptor do sensor, criando-se então um triângulo entre os três pontos intervenientes como se pode ver à esquerda na Figura 1.

Dependendo da distância a que um objecto se encontre do sensor, a reflexão provocada ao pulso de luz infravermelha terá um ângulo, o qual varia juntamente com essa distância. Este ângulo é então medido por uma array CCD (charge-coupled device), podendo a voltagem correspodente ser gerada. A relação entre a distância medida e a voltagem gerada é então mostrada à direita na Figura 1.

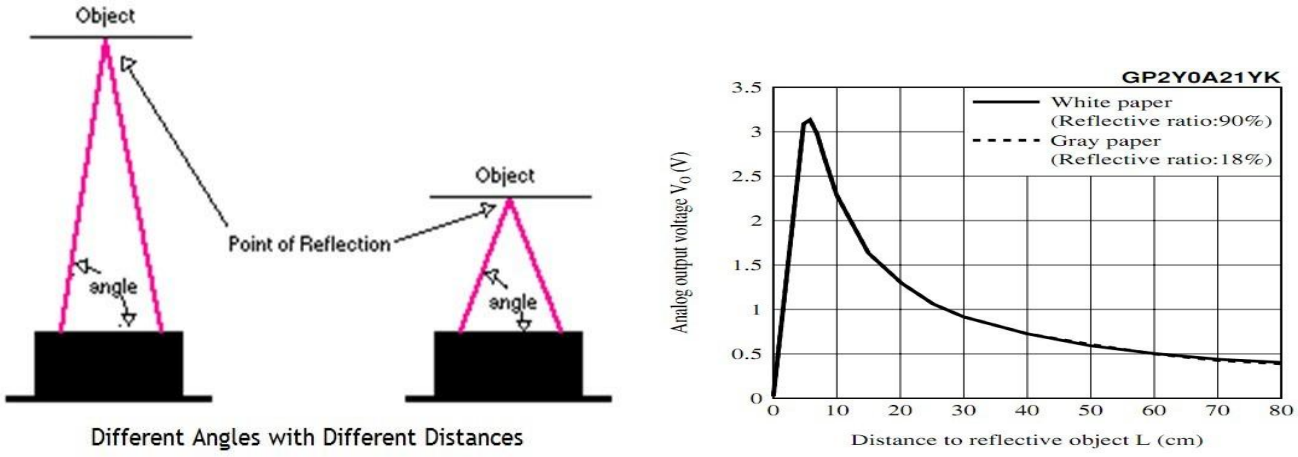

Figura 1: Esquerda: Triangulação efectuado com o sensor. Direita: Relação entre distância do objecto reflector ao sensor e o valor de voltagem gerado. Ambas: Retiradas do website da Grove.

3.3 Sensor de Temperatura e Humidade

Este outro componente do sistema, o AM2302 Temperature and Humidity Sensor Pro da Grove, serve tal como o nome indica para realizar medições de temperatura e humidade continuamente. No caso da medição da humidade, um elemento capacitivo único do sensor efectua leituras de humidade relativa enquanto que, no caso da medição da temperatura, as leituras são efectuadas utilizando um termístor NTC (negative temperature coefficient). Assim sendo, o sensor é capaz de lidar com valores de humidade entre 5% RH e 99% RH, e de temperatura entre $-40°C$ e $80°C$, com incertezas de 2% RH e $0.5°C$ respectivamente.

O sensor permite efectuar leituras quase ininterruptamente graças a um tempo de respostas bastante baixo. No diagrama de fluxo da Figura 2 pode ser visto o esquema de como as leituras de temperatura e humidade são feitas e atualizadas pelo sensor.

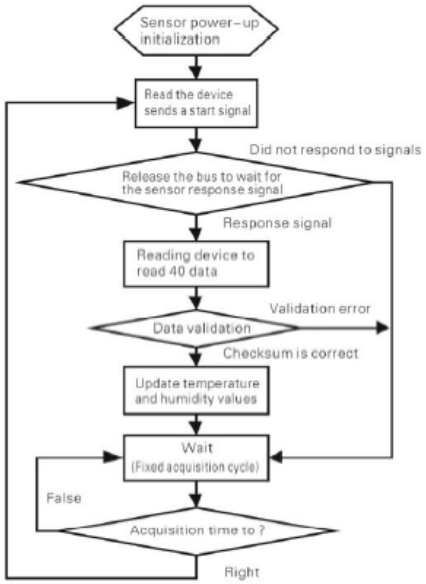

Figura 2: Diagrama de fluxo do sistema implementado no sensor para leitura de dados. Retirado da datasheet do sensor.

3.4 Border Router

Este componente intermediário do sistema é para todos os efeitos um *router* que tem como função criar a ligação entre elementos de baixo consumo, como é o caso de dispositivos IoT, e o resto da Internet por interface IPv6. Tendo em conta a natureza dos dispositivos e rede utilizados, foi considerado um border router 6LoWPAN, o qual implementa IPv6 sobre redes wireless e *low-power* de área pessoal. Desta forma, este componente foi simulado utilizando a ferramenta Tunslip.

4 MQTT Broker

O *broker* em contexto de IoT refere-se ao componente de software do sistema encarregue de distribuir mensagens JSON dos publicadores para os subscritores interessados. No caso específico deste projecto, e dado a que o protocolo mais adequado para a troca de mensagens entre publicadores/subscritores era o MQTT, foi aplicado o software *Mosquitto* indicado para a implementação de clientes MQTT. Este software tem várias vantagens para IoT uma vez que é bastante leve, tornando-se ideal para implementação em todo o tipo de dispositivos desde sensores *low power* até servidores completos.

4.1 Protocolo MQTT

Este protocolo IoT da camada de aplicação implementa o paradigma publicação/subscrição em modo *one-to-many* de forma eficaz e eficiente. Deste modo, existe um broker (como descrito acima) em todos os sistemas e ao qual os subscritores se autenticam enquanto que os publicadores lhe enviam mensagens.

Em termos técnicos, o protocolo MQTT é implementado sobre o protocolo TCP usando o porto 1883, o qual garante transporte confiável e ordeiro de uma stream de bytes sem perdas entre um cliente MQTT e um servidor MQTT. Opcionalmente, o protocolo aceita também a possibilidade de ser protegido através da aplicação do protocolo de encriptação TLS no porto 8883, como foi feito neste trabalho tendo a vista a ligação à cloud AWS.

Cada pacote de controlo do protocolo consiste em apenas um header fixo de tamanho 2 bytes, tendo este campos opcionais e sendo a payload também opcional. Apesar da payload poder ocupar até 256 MB, o protocolo continua a considerar-se leve.

5 AWS Cloud

Este serviço público de *cloud computing*, oferecido pela Amazon, baseia-se numa plataforma que integra diversos componentes de hardware, software e estrutura internet juntos com vista a fornecer serviços de computação a clientes que os requisitem, tal como a maioria dos serviços de *Cloud*. Este fornecimento é feito de modo *on-demand* e *pay-as-you-go*, o que significa que os serviços estão disponíveis para os clientes a qualquer altura em que sejam necessários e os respectivos custos serão apenas cobrados aquando do seu uso. Tendo em conta que a AWS oferece também os primeiros 12 meses grátis de acesso básico para novas contas e uma vez que contém todas as ferramentas necessárias a nível de *cloud* para este projecto, então apresentou-se como uma opção ideal para integração no sistema aqui proposto.

Apesar da *cloud* AWS fornecer diversos serviços aos clientes, aqueles que foram aplicados neste projecto foram os seguintes:

- IoT Core: Esta plataforma permite essencialmente ao utilizador conectar dispositivos IoT aos serviços da AWS. Deste modo, assegura proteção aos dados e às interações, processa e age consoante os dados recebidos pelo dispositivo, e ainda permitir a aplicações interagirem com os dispositivos, mesmo estes estando offline.
- DynamoDB: É um serviço que disponibiliza uma gestão total a uma base de dados NoSQL. Esta base de dados caracteriza-se por fornecer uma performance incrivelmente rápida e previsível, permitindo ao utilizador criar tabelas que guardam e devolvem dados na quantidade desejada.
- Lambda: É uma plataforma de computação "serverless"e orientada a eventos. Ou seja, é responsável por correr código na resposta a eventos, gerindo automaticamente os recursos computacionais necessários. Deste modo, permite criar aplicações simples e "on-demand"que são responsivas a eventos e a novas informações.
- IAM (Identity and Access Management): É um serviço web que permite aos clientes controlar os acessos aos recursos da AWS, isto é, saber quem está autenticado e autorizado a utilizar esses mesmos recursos.
- EC2 (Elastic Compute Cloud): Este serviço constitui uma parte central da plataforma de cloud-computing da Amazon, uma vez que permite aos utilizadores alugarem computadores virtuais que correm as suas aplicações, fornecendo uma capacidade de computação escalável na cloud AWS. Para além disso, permite que não seja necessário um investimento inicial de hardware a empresas/clientes que procuram desenvolver e lançar as suas aplicações de uma forma mais rápida.

6 Overview do Sistema

O sistema desenvolvido segue essencialmente o diagrama de arquitectura mostrado abaixo na Figura 3. Neste estão expostos todos os principais componentes integrantes do projecto.

A informação relativa ao parque de estacionamento (estado dos lugares a estacionar, temperatura e humidade) é recolhida através dos sensores previamente descritos, a uma taxa de amostragem de $2s$. Esta informação é devidamente acondicionada em formato JSON para ser enviada para a cloud através do protocolo MQTT, utilizando o Tunslip como simulação de um Border Router. Como referido anteriormente, o MQTT é implementado sobre o protocolo TCP, assegurando assim que não existe perdas durante a transmissão dos dados. Por outro lado, a transmissão é segura devido à encriptação TLS/SSL. A chegada dos dados à cloud é feita à mesma frequência que a de aquisição, o que provoca um evento, sendo gerido por uma função lambda que assegura o processamento correto dos dados: o estado dos lugares de estacionamento é atualizado e novas estatísticas são criadas.

Por fim, como todas as informações estão devidamente armazenadas na nuvem, estas podem ser facilmente mostradas numa aplicação desejada, neste caso, um website. Considera-se que o website constitui uma valiosa interface com o utilizador, uma vez que é cross-platform e permite uma experiência plug-and-play.

Um exemplo do funcionamento do sistema que mostra a execução dos passos anteriormente descritos pode ser visualizado em: https://www.youtube.com/watch?v=Mq7XqlAldc8.

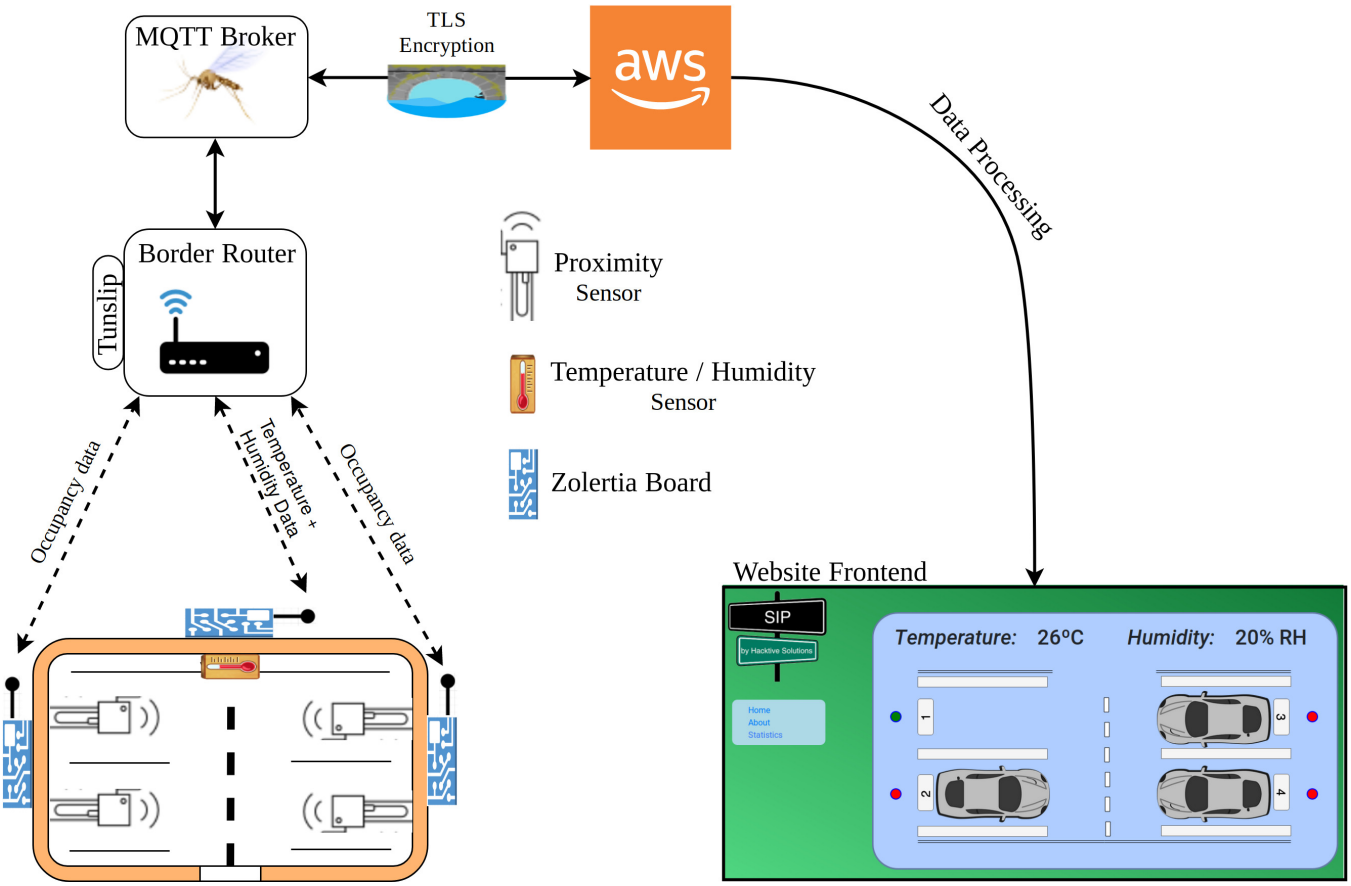

Figura 3: Diagrama de arquitectura do sistema desenvolvido.

7 Frontend Website

Por um lado, o website desenvolvido permite aos utilizadores do parque visualizar o estado atual dos lugares de estacionamento, possibilitando uma decisão rápida, e por sua vez poupar o seu tempo. Através das estatísticas presentes neste website, poderão também observar qual o dia sugerido para estacionar. Por outro lado, os proprietários do estacionamento possuem nas suas mãos uma excelente ferramenta para ajustar o preçário do mesmo, bem como monitorizar a qualidade do ar, acionando se necessário actuadores externos ao sistema.

O website está disponível em https://orion.isr.uc.pt/~gustavo.assuncao/iot/index.html.

8 Outras Considerações

Considera-se que o sistema desenvolvido constitui uma ferramenta útil para o dia-a-dia dos utilizadores que recorrem a parques de estacionamento com uma elevada taxa de lotação.

O sistema desenvolvido conta com diferentes linguagens de programação: PHP/Javascript/HTML/CSS (website), C (placas RE-Mote) e Python (Função Lambda e Bridge de encriptação). Para aceder à DynamoDB a partir do website, foi aplicado o AWS SDK para PHP fornecido pela AWS.

Dado a que o core foi inteiramente desenvolvido e testado, este sistema poderá ser utilizado em parques de estacionamento de maior escala, sendo apenas necessário substituir a planta do parque apresentada no website, e replicar os pares de sensores-módulos utilizados. No entanto para uma melhor gestão de recursos, recomenda-se a alteração dos sensores utilizados para sensores de natureza digital.