INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Sistema de apoio à gestão energética

JOSÉ CARLOS NOGUEIRA TEIXEIRA
outubro de 2017

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Sistema de apoio à gestão energética

José Carlos Nogueira Teixeira

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Engenharia de Software

Orientador: Nuno Bettencourt

Júri:

Presidente:

Vogais:

Porto, Outubro de 2017

Resumo

Este documento foi desenvolvido na disciplina de Tese de Mestrado (TMDEI) do Curso de Mestrado em Engenharia de Software no Instituto Superior de Engenharia do Porto com o tema Sistema de Apoio à Gestão Energética.

Este projeto tem por objetivo a criação de um sistema capaz de gerir a energia utilizada num determinado espaço, de forma automática e eficiente. Esse sistema é composto por uma central, que será um computador capaz de comunicar com vários dispositivos[1], que fazem a leitura dos consumos energéticos. Esses dispositivos estão ligados aos vários equipamentos[2] que se pretende que sejam monitorizados com vista ao aumento da eficiência energética ou de vida útil dos mesmos. A central comunica com os vários dispositivos e sensores tanto para receber os dados para análise como para executar ações, como por exemplo, parar o fornecimento de energia a um determinado equipamento. Sensores como os de temperatura ou movimento podem ser utilizados para melhorar a forma como é feita essa gestão.

Palavras-chave: Poupança, Energia, Internet das Coisas

[1] Termo utilizado para referir o dispositivo de monitorização energética
[2] Termo utilizado para referir o equipamento que se pretende que seja monitorizado

Abstract

This document was developed into the subject of the Master Thesis of the Software Engineer Master Degree in the Instituto Superior de Engenharia do Porto with the following title: Support System to Energy Management.

The main goal of the project is the creation of a system that can manage the used energy in a specific space, in an automate and efficient way. The system is composed with a central, which will be a computer able to communicate with other devices[3] that will do the reading of the energy consumption. The devices are connected with other equipment[4] that are meant to be monitored with the main goal of the increase of the energy efficiency or their life spam. The central will communicate with the devices and sensors either to receive the data for analysis or to execute actions, like stopping the energy supply of a specific equipment. Temperature sensors or movement sensors can be used to improve the way these management is done.

Keywords: Saving, Energy, Internet of Things

[3] Term used to define the device used in energetic monitorization
[4] Term used to define the equipment that will be monitored

Agradecimentos

Agradeço a todas as pessoas que me ajudaram na realização desta tese.

Ao ISEP e ao DEI por todo o conhecimento que obtive durante a Licenciatura em Engenharia Informática e o Mestrado em Engenharia de Software.

Ao meu orientador Nuno Bettencourt do ISEP pela disponibilidade durante todo o projeto para qualquer problema que surgisse.

À Susana Marques por todo o apoio incondicional e ajuda na elaboração do documento assim como ao longo do desenvolvimento.

Ao Pedro Salazar, Bruno Fernandes, Rafael Soares e André Francisco pelas sugestões e dicas na criação do sistema.

Ao Carlos Teixeira por todo o apoio no início do meu percurso académico no ISEP.

Aos meus pais e à minha irmã pelo apoio e conselhos durante todo o meu percurso enquanto estudante.

A toda a minha família e amigos que me apoiaram durante este percurso.

Índice

Lista de Figuras

Lista de Tabelas

Lista de listagens de código

Acrónimos e Símbolos

Lista de Acrónimos

API	*Application Programming Interface*
ATT	*Attribute Protocol*
BLC	*Bluetooth Classic*
BLE	*Bluetooth Low Energy*
BR	*Basic Rate*
EDR	*Enhanced Data Rate*
IDE	*Integrated Development Environment*
IOT	*Internet of Things*
ISP	*Internet Service Provider*
FFE	*Fuzzy Front End*
L2CAP	*Logical Link Control and Adaptation Protocol*
MFi	*Made for iPhone/iPod/iPad*
MVC	*Model View Controller*
P2P	*Peer to Peer*
REST	*Representational State Tranfer*
SDK	*Software Development Kit*
SOAP	*Simple Object Access Protocol*
SSD	*System Sequence Diagram*
UC	*Use Case*

1 Introdução

Cada vez mais, na atualidade, se tem discutido sobre as questões relativas á proveniência da energia, essencial á manutenção da sociedade tal como a conhecemos. Com a escassez de petróleo sentida internacionalmente e alterações climáticas provenientes da combustão de combustíveis fósseis torna-se urgente a procura de alternativas e formas de economizar os recursos existentes[1].

O estado do meio ambiente é preocupante e tem-se agravado significativamente nos últimos anos [2]. O consumo excessivo de energia potencia o aquecimento global do planeta devido a libertação de carbono que provém dos combustíveis fósseis, que são por sua vez, a sua fonte primária [3], [4]. Estes são problemas que afetam as gerações atuais e futuras.

Segundo o U.S. Energy Information, nos Estados Unidos da América, somente cerca de 15% da eletricidade provém de recursos renováveis, 20% de energia nuclear, sendo os restantes 65%, gerada com recurso a combustíveis fósseis, na maioria com recurso a gás natural com 33.8% de utilização [5]. Essa eletricidade é indispensável para a maioria das pessoas, sendo utilizada de forma contínua durante o dia, seja para iluminação, conservação de alimentos, cozinhar entre outros [6].

Na maioria dos países, cerca de 40% da energia é consumida pelos edifícios sendo possível estimar que uma grande parte dessa energia pode ser poupada [7], o que pode significar numa redução anual de 400 milhões de tonelada de CO2[8].

Algumas das soluções já implementadas são, por exemplo os painéis fotovoltaicos utilizados em algumas habitações, que têm por objetivo minimizar o uso da energia elétrica proveniente do exterior. Nos E.U.A., na última década o aumento da aposta em energia solar tem sido de cerca de 60% por ano[9]. Apesar da adoção estar a ser relativamente lenta, mostra uma preocupação crescente por parte da população em relação à utilização de recursos não renováveis. Na Califórnia, o objetivo passa por em 2020 as construções de novas casas residenciais não terem fornecimento de energia líquida e para isso foi criado o plano Zero Net Energy[10]. Esse plano quer tornar as casas mais eficientes e autossustentáveis fazendo uso de recursos renováveis para gerar energia [11].

Quantos aos equipamentos, alguns fabricantes já podem ser impedidos de vender os seus produtos se não tiverem determinadas preocupações ambientais, tais como os custos em termos energéticos. É o caso da União Europeia que desde 2013 um equipamento não deve consumir mais de 0,5W em *standby*[12].

Atualmente já existem dispositivos, tais como, as tomadas com temporizador, que ligadas a determinados equipamentos eletrónicos ligam ou desligam o equipamento na hora que o utilizador determinar. Por contraponto, a maioria da gestão é manual, isto é, se por lapso, algum equipamento ficar ligado à rede elétrica mais tempo do que deveria, ou sem estar em utilização, o mesmo só será desligado quando o utilizador fizer essa verificação ou o equipamento se desligar automaticamente. Isto conduz a consumos energéticos que têm custos para o utilizador e meio-ambiente, assim como um desgaste superior do equipamento.

É cada vez mais importante, a existência de um controlo energético dos equipamentos, seja para a obtenção de uma redução de custos ou para aumentar o seu tempo de vida útil.

Neste capítulo apresenta-se o problema que se pretende resolver e respetiva descrição do mesmo. Com base no problema serão formulados os objetivos do projeto assim como as perguntas que se necessita responder.

1.1 Objetivos

Este trabalho tem vários objetivos, com vista a demonstrar que um sistema pode ser capaz de gerir corretamente um equipamento ligado à corrente elétrica de modo a aumentar a sua eficiência energética.

Para efetuar essa gestão de forma automática, torna-se basilar a criação de um sistema que se encontre ligado a uma rede de sensores. Alguns desses sensores estão ligados a pontos de acesso de corrente elétrica, de forma a medir e controlar os consumos, e outros serão capazes de obter vários valores como a temperatura ou se existe movimento no espaço. Assim o sistema, com o auxílio dos sensores, permite ao utilizador verificar os dados recolhidos dos mesmos, e tem a capacidade de ajudar na tomada de decisões com vista a um melhor desempenho ao nível da eficiência energética. Isto pode permitir que equipamentos que utilizem baterias tenham um aumento da vida útil da mesma.

O utilizador será capaz de monitorizar os equipamentos e o espaço em tempo real, podendo definir como pretende que seja feita a gestão energética. Dado a gestão ser automática, pode ser pedido ao utilizador para escolher um plano com maior ou menor disponibilidade, isto é, que o sistema seja mais conservador e dentro dos parâmetros aceitáveis de utilização ligue equipamentos, como por exemplo, o frigorífico antes deste atingir uma temperatura alta, ou que seja mais ativo e tenha maior liberdade para não o fazer. Pode, portanto, escolher um plano para um menor consumo energético ou maior disponibilidade em termos de utilização. Com a apresentação dos valores obtidos dos sensores, é possível detetar problemas

relacionados com consumos exagerados o que poderá indicar nalguns casos defeito ou avaria de equipamentos.

Com a utilização deste produto, será possível analisar em maior detalhe e com rigor dados de consumo energético efetuado no edifício ou habitação em que a tecnologia seja aplicada. Com isto o cliente continua a ter o mesmo conforto e a mesma disponibilidade de todos os equipamentos, mas com uma redução da pegada ecológica.

1.2 Perguntas de investigação

Pretende-se com este trabalho dar resposta às seguintes questões de investigação:

- O sistema permite reduzir o consumo energético?
- Todos os equipamentos controlados pelo sistema reduzem o seu consumo energético?
- O sistema interfere com a habitual utilização dos equipamentos?
- Quando aplicado a dispositivos com baterias, existe uma melhoria na vida útil da mesma?

Com base nos objetivos listados anteriormente torna-se relevante conseguir responder a estas questões para comprovar os efeitos positivos ou negativos da utilização de um sistema deste tipo.

1.3 Sumário

Neste capítulo foi apresentado uma introdução ao problema, assim como os objetivos que se pretende atingir. É necessário proceder ao levantamento de um conjunto de tecnologias que possam ser úteis no desenvolvimento de um sistema que seja uma solução para esses problemas.

2 Contexto e estado da arte

Este capitulo é relevante para contextualização dos problemas e dos objetivos que se pretende atingir. Para solucionar os problemas, foi feito um levantamento de todas as funcionalidades que se pretendem implementadas e um levantamento das tecnologias capazes de ajudar nessa resolução.

2.1 Internet of Things

A *Internet of Things* (IOT) tem por objetivo ligar entre si ou à internet todo o tipo de equipamentos eletrónicos que utilizamos diariamente, por exemplo, *smartphones*, *smartwatches*, sensores entre outros [13].

2.2 Comunicação

Existem várias tecnologias que permitem fazer o envio de dados entre os dispositivos. Essa comunicação pode ser feita com ou sem fios, mas visto que a facilidade de instalação dos dispositivos assim como a mobilidade dos mesmos poderá ser uma vantagem, o objetivo é focar nas tecnologias sem fios. Existe, portanto, a necessidade de conectar vários sensores ou dispositivos que se encontram espalhados por uma área indeterminada. Outro dos requisitos é estas tecnologias estarem disponíveis para a maioria dos sistemas pois será relevante a sua comunicação no futuro com esses dispositivos.

2.2.1 Bluetooth

O Bluetooth é uma tecnologia de comunicação sem fios, desenvolvida pela empresa Bluetooth SIG (Special Interest Group) que lançou recentemente a versão 5 [14].

Os dispositivos Bluetooth utilizam as tecnologias designadas por Bluetooth *Low Energy* ou Bluetooth *Classic,* também conhecido como Bluetooth *Basic Rate/Enhanced Data Rate* (BR/EDR) para comunicar. Existem 3 tipos de dispositivos Bluetooth:

- Bluetooth – Os dispositivos que suportam Bluetooth *Classic*.
- Bluetooth *Smart* – Os dispositivos que suportam Bluetooth *Low Energy*.
- Bluetooth *Smart Ready* – Os dispositivos que suportam as duas versões da tecnologia: Bluetooth *Low Energy* e Bluetooth *Classic*. É possível efetuar conexões e envio de dados entre as ultimas duas versões da tecnologia referidas.

Na Figura 1 é possível ver como os diferentes dispositivos Bluetooth podem comunicar.

Figura 1 - Versões do Bluetooth [15]

Em 2014 existiam 3 mil milhões de dispositivos com a tecnologia Bluetooth e em 2018 são esperados existirem perto de 5 mil milhões de dispositivos. Este aumento é esperado graças aos dispositivos *Internet of Things* (IOT) porque o Bluetooth *Smart* é uma norma [16].

Para troca de dados depois de estabelecida a ligação, a tecnologia Bluetooth utiliza o protocolo designado por Atributte Protocol (ATT) [17].

Um dos tipos de Bluetooth é o *Classic*, tem velocidades de cerca de 2 a 3 Mbps e segundo a especificação tem um alcance de cerca de 100 metros [18]. A velocidade e a distância máxima diminui conforme as interferências e o meio em que se pretenda fazer a comunicação. Graças às velocidades que esta tecnologia oferece já permite, por exemplo, *streaming* de áudio o que permite ser utilizada em *kits* de mãos livres ou *Headphones*.

Como já referido anteriormente, outro dos tipos é o Bluetooth *Low Energy* é a versão mais recente e foi pensada para o mercado de IOT. A velocidade máxima é cerca de 236 kbit/s [19] e tem uma alcance esperado de cerca de 100 metros [20]. Tem um consumo energético bastante inferior ao Bluetooth *Classic* e por isso é utilizado em *beacons* e outros dispositivos IOT onde o baixo consumo energético é essencial. O consumo é entre 2 a 100 vezes inferior ao Bluetooth *Classic* [21]. A velocidade de envio é baixa e por isso é mais utilizada para o envio de pouca informação.

Existem dois modos de funcionamento para o Bluetooth *Low Energy*:

- Periférico
- Central

O modo Periférico permite enviar dados para a rede, isto é, o dispositivo é capaz de funcionar semelhante a um servidor, notificando a central quando tiver novos dados.

O modo Central permite ler os dados existentes nos Periféricos e subscrever características dos mesmos para quando existir uma alteração nos dados disponíveis, ser notificado. Na Figura 2 é possível ver um exemplo do modo Central e Periférico.

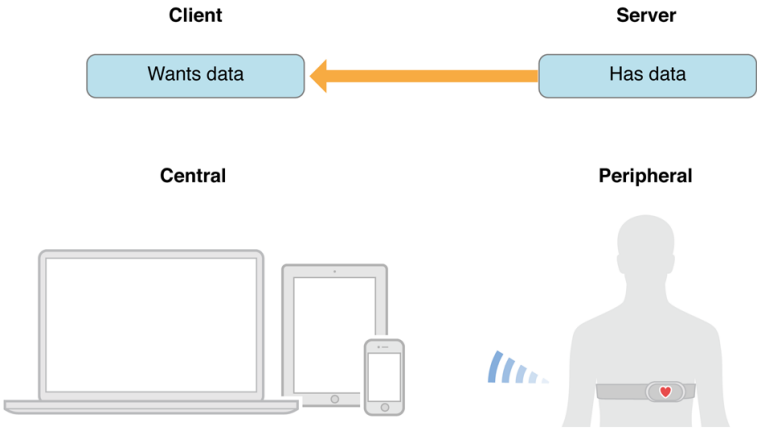

Figura 2 - Modo Central e Periférico [22]

Na Figura 3 é possível verificar o modo como a central faz a descoberta do periférico. O periférico emite pacotes de *advertising* e a central é capaz de os ler.

Figura 3 - Descoberta e Anúncio [22]

2.2.2 Wi-Fi

O Wi-Fi é uma tecnologia de comunicação sem-fios que funciona atualmente na banda 2.4Ghz ou 5.0Ghz [23]. Esta tecnologia permite-nos dois tipos de comunicação que podem ser interessantes para este projeto:

- Wi-Fi Infraestrutural
- Wi-Fi Direct

O Wi-Fi infraestrutural é a ligação de dois ou mais dispositivos através de uma infraestrutura. O alcance e velocidade de ligação vai depender da qualidade da infraestrutura, das normas que a infraestrutura utiliza e da qualidade dos adaptadores Wi-Fi dos dispositivos. Com a

norma 802.11ac já é possível atingir velocidade de 1300 Mbps [24]. Na Figura 4 é visível um exemplo de uma rede que utiliza Wi-Fi infraestrutural.

Esta infraestrutura pode permitir o acesso à internet e as velocidades vão variar de acordo com as velocidades contratadas ao *Internet Service Provider* (ISP).

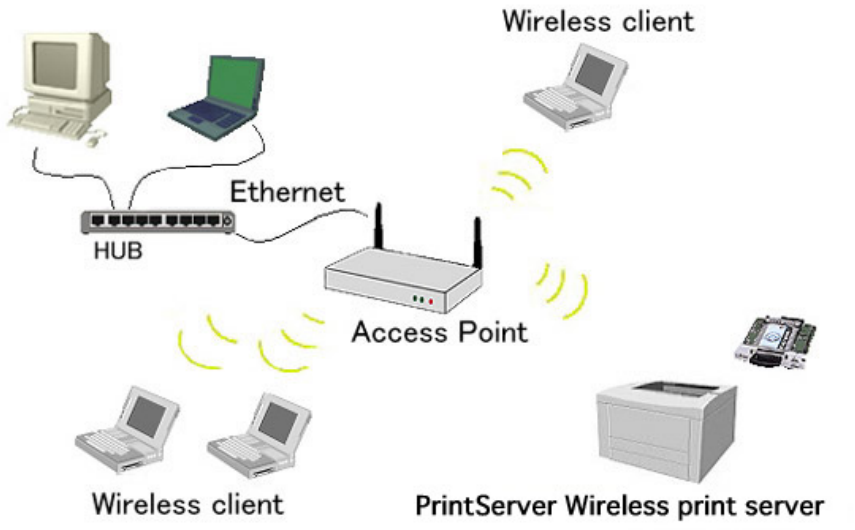

Figura 4 - Rede Infraestrutural [25]

Pode ser relevante a utilização desta tecnologia se o acesso à internet for relevante.

Quanto ao Wi-Fi Direct, é uma tecnologia sem fios presente em alguns dispositivos. Permite criar uma rede ponto a ponto e funciona de maneira semelhante a uma rede ad-hoc, ou seja, um dispositivo cria a sua rede e os outros dispositivos ligam-se, podendo ser enviadas mensagens de um para muitos (*broadcast)* ou de ponto a ponto (*unicast)*. Na Figura 5 é possível ver um exemplo dessa rede.

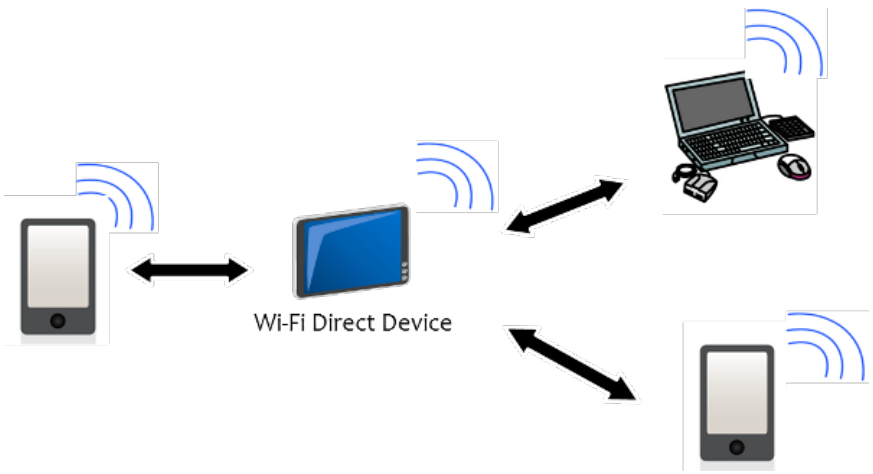

Figura 5 - Rede Wi-Fi Direct [26]

A grande vantagem em relação a uma rede infraestrutural é não precisar de um equipamento que faça o encaminhamento das mensagens para os dispositivos. A velocidade é dependente da tecnologia Wi-Fi utilizada, podendo atingir 250 Mbps [27] e tem um alcance máximo de cerca de 200 metros [28].

2.2.3 Consumo energético

Os consumos energéticos de cada tecnologia vão sempre variar de acordo com o equipamento que é utilizado e com os dados que vão ser enviados e recebidos. Para Wi-Fi infraestrutural ou Ad-hoc, segundo alguns estudos, os custos energéticos são muito semelhantes. Para um envio de dados, a uma média de 1186KB/s por TCP tem um custo médio de 1568mW. Enquanto que Bluetooth *Classic* fica-se pelos 520mW mas com uma média de 137KB/s [29].

Para outro estudo Bluetooth *Low* Energy, a anunciar pacotes por *broadcast* a cada 500ms tem um consumo de 0.147mW enquanto que o Wi-Fi, usando UDP a transmitir a 40Mbps, gasta um total de 210mW [30].

2.2.4 Comparativo

Na Tabela 1 é possível visualizar um comparativo das quatro tecnologias baseado nas características referidas anteriormente.

TECNOLOGIA	VELOCIDADE MÁXIMA	ALCANCE MÁXIMO	CONSUMO
WI-FI DIRECT	250 Mbps	200 metros	Elevado
WI-FI INFRAESTRUTURAL	1300 Mbps[5]	-	Elevado
BLUETOOTH LOW ENERGY	0,236 Mbps	100 metros	Baixo
BLUETOOTH CLASSIC	2 a 3 Mbps	100 metros	Médio

Tabela 1 - Comparativo de tecnologias

Para o Wi-Fi infraestrutural não foi atribuído um valor para o alcance máximo porque está bastante relacionado com a qualidade da infraestrutura e da norma utilizada.

2.3 Redes Mesh

As redes Mesh são uma topologia de rede que permite aumentar a distância de alcance de uma rede. Quantos mais nós existirem numa rede maior é a probabilidade de o seu alcance

[5] Velocidade máxima para a norma 802.11ac

ser também superior, como se pode ver na Figura 6, em que os círculos representam nós na rede e as linhas ligações entre os nós.

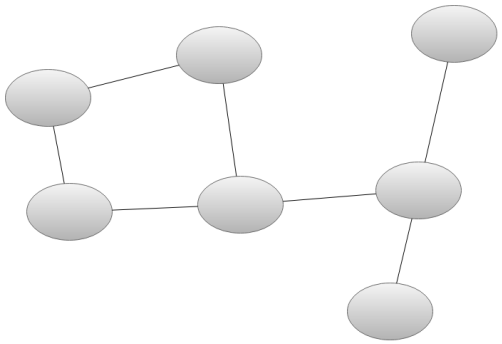

Figura 6 - Exemplo de rede mesh

Normalmente, são mais tolerantes a falhas porque podem ter vários caminhos para o mesmo destino. Estes sistemas costumam depender de uma tabela de rotas, que permite obter informações acerca do melhor caminho para entregar uma mensagem [31].

2.4 Hardware

Foi preciso efetuar um levantamento de todo o hardware que pode ser útil para a construção do sistema e que tenham disponível algum tipo de comunicação referida na secção 2.2. Os resultados dessa pesquisa são:

- O Raspberry PI 3 é capaz de comunicar através de Wi-Fi e de Bluetooth, tem 1 GB de memória Ram e um processador Quad-Core a 1.2GHz e tem ainda portas onde é possível ligar sensores[32].
- O Arduino é uma placa programável de baixo custo, que permitiu a muitos engenheiros desenvolverem soluções ao nível da eletrónica. Com a placa eletrónica Arduino é possível ligar luzes, sensores e vários outros dispositivos. A placa é programável na linguagem C ou C++ [33]. Tem disponíveis as interfaces I2C e SPI que permite acoplar sensores[34].
- O ESP8266 um microcontrolador Wi-Fi que pode ser acoplado de forma fácil a um dispositivo, por exemplo ao Arduino de forma a permitir a comunicação ou o acesso à internet com um baixo custo energético[35].
- A Wemos é uma placa programável, muito semelhante ao Arduino, e em que a grande diferença é já ter integrado o microcontrolador ESP8266EX o que permite comunicações por Wi-Fi sem ser necessário ligar um módulo especifico para o efeito[36].
- A Wavershare BLE400 tal como o Arduino ou o Wemos é uma placa programável mas que tem incorporado um Chip Nordic NRF51822 que permite comunicar através da tecnologia Bluetooth *Low Energy* e baixos consumos energéticos [37]. Tem disponíveis as interfaces I2C, SPI e UART para permitir ligar outros módulos ou sensores.

2.4.1 Sensor de Movimento

O sensor de movimento pode ser usado para identificar se alguém se encontra dentro do espaço e em que lugar se encontra, para por exemplo, as luzes serem ligadas só quando é identificado movimento nesse local. Existem vários sensores deste tipo, disponíveis tanto para a plataforma Arduino [38] como para o Raspberry PI [39]. Dos quais se destaca:

- **Infravermelho -** O infravermelho passivo deteta o calor corporal e são normalmente usados em segurança doméstica [40].
- **O micro-ondas -** emite várias micro-ondas e mede a reflexão em um objeto em movimento. Costumam cobrir uma área maior que os sensores infravermelhos, mas são mais sensíveis a interferências elétricas [40].
- **Tecnologia dupla -** São sensores que tem disponíveis duas tecnologias para analisar o meio de modo a reduzir os falsos alarmes, por exemplo, um sensor com infravermelho com micro-ondas [40].
- **Refletivo -** Emite vários raios infravermelhos e através da reflexão desses raios é capaz de identificar se existe dentro de determinada área um corpo em movimento [40].
- **Ultrassónico -** Envia várias ondas ultrassónicas e tal como o sensor por micro-ondas verifica a reflexão em num objeto em movimento [40].
- **Vibração -** É um sensor capaz de detetar vibração e assim identificar movimento num dado local [40].

Em caso de utilização de um sensor de movimento a escolha do tipo, vai estar sempre dependente do tipo de espaço que se pretende monitorizar.

2.4.2 Sensor de Temperatura

Os sensores de temperatura permitem obter a temperatura que se encontra num determinado local onde são colocados. Tal como os sensores de movimentos existem vários módulos que podem ser adicionados ao Raspeberry Pi, Arduino, entre outros, que fazem a leitura da temperatura nesse instante. A SparkFun, tem um sensor de temperatura, modelo TMP102 que permite ler temperaturas com uma resolução até 0,0625ºC e uma precisão até 0,5ºC[41].

2.4.3 Medidor de consumo energético

É necessário um dispositivo que permita fazer a leitura dos consumos energéticos. Esse dispositivo vai encontrar-se ligado entre a tomada e o equipamento e permitirá medir em tempo real os consumos naquele instante.

Já existem no mercado soluções que permitem monitorizar o consumo energético. O TED Pro Home é uma dessas soluções, sendo um sistema que inclui vários dispositivos que permite monitorizar o consumo energético num local [42]. Outra solução é o Neurio que através da

instalação de um equipamento de monitorização energética permite ao utilizador obter informação relativa aos consumos energéticos. A EDP, com o serviço RE:DY, permite a monitorização de uma habitação com o auxílio de tomadas inteligentes[43].

Existem algumas tomadas disponíveis no mercado que permitam ao utilizador através do *smartphone*, ligar ou desligar um equipamento e obter os consumos energéticos. De destacar a tomada HS110 da TP-Link [44] e Wemo da Belkin [45]. A grande desvantagem é o fato de nenhuma dessas tomadas ter disponível um API pública para a sua utilização por outros programadores.

O UNI-T UT230B-EU é outra das soluções que podem ser utilizadas, mas que obrigam o utilizador a verificar junto da própria tomada o consumo que o equipamento está a realizar. Este equipamento permite medir a intensidade da corrente, a tensão, a potência, o fator de potência e a frequência. O erro na leitura dos valores é de aproximadamente 1%, exceção feita à frequência que não é anunciado qualquer margem de erro e no caso do fator de potência o erro sobe para 2%[46].

2.5 Software

Depois de referidos vários dispositivos, é importante efetuar o levantamento de software que em conjunto com esses dispositivos possa ajudar a criar uma solução que resolva o problema apresentado.

Computadores com características semelhantes a um Raspberry Pi suportam sistemas operativos como Windows IOT e algumas distribuições Linux que são apresentadas em seguida:

- **Windows IOT -** É uma versão otimizada do Windows 10 para dispositivos mais pequenos como é o caso do Raspberry PI. Tem a vantagem de utilizar a *Universal Windows Plataform* (UWP), API que é usada atualmente nos restantes dispositivos Windows 10 [47].
- **Ubuntu Core -** é mais leve que o Ubuntu e desenvolvido para ser utilizado em sistemas embebidos ou IOT [48]. Suporta vários *chipsets* e *boards* em que se inclui o Raspberry Pi 2 e 3 e também outros equipamentos como Intel NUC ou o Samsung Artik [49].
- **Raspbian -** é um sistema operativo gratuito otimizado para o Raspberry Pi. Está em constante desenvolvimento e a receber atualizações e vem com mais de 35.000 *packages* [50].

Para programar nestes sistemas operativos ou para alguma das placas programáveis referidas anteriormente podem ser utilizadas linguagens como:

- **C -** A linguagem foi escrita por Dennis Ritchie em 1972 [51], e no início dos anos 80 foi criada uma extensão da linguagem C, o C++ por Bjarne Stroustrup nos laboratórios

Bell. O C++ fornece vários recursos de C e permite programação orientada a objetos [52].

- **Java -** É uma linguagem da Oracle e foi incorporado pela primeira vez em 1995 no *Browser* Netscape Navigator Internet. Atualmente é utilizado em várias plataformas, telemóveis, jogos e até sistemas de navegação [53].
- **Python –** É uma linguagem criada nos inícios dos anos 90 por Guido van Rossum. Foi apresentado como sucessor da linguagem ABC [54].
- **C# -** É uma linguagem orientada a objetos com algumas semelhanças a Java e que neste momento se encontra na versão 7.0 [55].

Das linguagens apresentadas anteriormente C, C++ e Python podem ser utilizadas em todos os sistemas operativos, Java para distribuições Linux e C# para Windows IOT [56]. Algumas destas linguagens podem ser usadas para programar as placas, tais como C e C++. Apesar disso, pode também ser útil o uso de bibliotecas do sistema ou externas que permitam acesso a API's de Bluetooth, Wi-Fi ou que auxiliem no desenvolvimento do software.

Para o acesso às funcionalidades de Bluetooth através de distribuições Linux, pode ser utilizado a biblioteca BlueZ. Esta biblioteca encontra-se na versão 5 e fornece suporte no desenvolvimento de soluções que utilizem Bluetooth *Classic* ou Bluetooth *Low Energy* [57]. Apesar da falta de documentação disponível para esta biblioteca, o seu uso é simplificado se for utilizado o GDBus[6] que é um software que facilita a comunicação entre diferentes módulos de software [58].

Quanto às placas programáveis, se estas tiverem suporte para tal, a biblioteca Arduino BLEPeripheral simplifica e facilita a interação com o adaptador de Bluetooth [59] e para a monitorização dos sensores o Arduino disponibiliza a biblioteca Wire, que permite fazer a leitura dos valores detetados pelos sensores pela interface I2C[60].

Para o desenvolvimento de uma solução que faça parte do sistema e que inclua um servidor, existem algumas *frameworks* disponíveis tais como Django ou Node.JS. O Django tem por objetivo o desenvolvimento rápido de aplicações e um design limpo, utiliza a linguagem Python, é gratuito e o seu código é aberto[61]. Quanto ao Node.JS utiliza uma programação orientada a eventos, a linguagem de programação usada para desenvolver aplicações é JavaScript e tem disponível o maior ecossistema de bibliotecas de código aberto que é o npm[62]. Caso seja necessário comunicação da máquina para o servidor, ambas as soluções suportam a utilização do estilo arquitetural REST (*Representational State Tranfer*) que simplifica a criação de *web services*, e usa o protocolo HTTP para comunicar com o servidor ou do protocolo SOAP (*Simple Object Access Protocol*) que normalmente é mais utilizado em meios empresariais com finalidades semelhantes ao REST[63]. Se a direção da comunicação for a inversa, isto é, do servidor para a máquina podem ser utilizados websockets ou *long polling*. A utilização de *long polling* significa que, através de um pedido AJAX a máquina fica à espera de resposta do servidor por tempo indeterminado, isto permite ao servidor enviar dados para a máquina em forma de resposta a um pedido anterior que tenha sido feito [64].

[6] https://git.kernel.org/pub/scm/bluetooth/bluez.git/tree/doc

Websockets permitem comunicações bidirecionais entre o servidor e a máquina e são mais eficientes que a utilização do *long polling* [64].

2.6 Vida útil das baterias

O sistema se obtiver a informação do nível de bateria de um equipamento, pode aumentar a vida útil das mesmas, diminuído a velocidade com que existe uma degradação da bateria. Existem vários tipos de baterias como se pode ver na Tabela 2.

Tipo de bateria	Utilização
Níquel cadmio	Equipamento biomédico Rádios
Hidreto de níquel metálico	Telemóveis Computadores portáteis
Chumbo ácida	Equipamento de hospital Cadeiras de rodas
Ião de lítio	Telemóveis Computadores portáteis
Polímero de ião de lítio	Telemóveis

Tabela 2 - Tipos de bateria [65]

De todos esses tipos as mais importantes para o tipo de utilização que se pretende são as de ião de lítio, utilizadas em muitos computadores e telemóveis atuais.

Uma bateria de lítio dura em média perto de 500 ciclos de carga e descarga e começam a perder capacidade ao longo do tempo e quando submetidas a temperaturas elevadas. De modo a aumentar a vida útil das mesmas devem ser evitadas descargas e cargas completas da bateria e carregar a bateria mais frequentemente entre utilizações. Outro dos problemas que reduz a capacidade das baterias é quando ficam muito tempo a serem carregadas o que pode provocar um desgaste ainda maior que os ciclos de carga [66].

2.7 Consumo energético

Para o cálculo do consumo energético, existem três tipos de potências que podem ser calculadas, a potência aparente, ativa e reativa. Por potência aparente entende-se que é potência fornecida ao circuito [67]. Para o seu cálculo pode ser utilizada a seguinte formula:

$$P = I * V$$

Em que P é a potência, I a intensidade da corrente e V é a tensão da corrente também denominada por diferença de potencial (D.D.P).

Por potência ativa entende-se que é a potência útil, utilizada para fazer o trabalho [67]. Para o seu cálculo pode ser utilizado a seguinte fórmula:

$$P = I * V * \text{P. F.}$$

Em que em relação à formula anterior é adicionado a multiplicação pela variável, fator de Potência (F.P.).

Existe ainda a potência reativa que é a potência que representa a quantidade de trabalho não usada [67], que pode ser gerada por componentes reativos.

2.8 Mercado

O mercado neste momento já conta com um grande número de companhias a investir na área da eficiência energética dos edifícios, como por exemplo, a Discovergy[68], EMS3[69], eSight Energy[70] entre outras. Têm produtos que prometem ao utilizador uma redução nos custos energéticos através da monitorização em tempo real de vários equipamentos.

A empresa Discovergy permite ao utilizador receber notificações e dicas com base nos consumos energéticos de forma a conseguir reduzir custos. Tem também uma aplicação disponível para os *smartphones* [68].

A EMS3 tem disponível contadores inteligentes sem fios que fornecem todo o histórico do uso de energia e com isso ajuda a gerenciar melhor todas as fontes que gastam energia [69].

A eSight fornece uma plataforma que permite monitorizar e reduzir o consumo de energia. A plataforma é capaz de identificar ineficiências, notificar o utilizador e dispõe de um monitor de progresso que dispõe de todo o histórico de utilização [70].

A Digital Energy tem uma solução que permite reduzir os custos de energia e os consumos, analisando e interpretando o uso da energia através de relatórios e gráficos.

2.9 Sumário

Foram apresentadas várias tecnologias que podem ajudar a desenvolver um sistema que permita atingir os objetivos definidos, e a partir disso é possível gerar uma proposta de sistema mais concreta que será apresentada no próximo capítulo.

3 Proposta

O propósito desta tese é desenvolver um sistema que em conjunto com dispositivos de monitorização energética, seja capaz de gerir e analisar os consumos de modo a fornecer dados ao utilizador com o objetivo de uma redução do consumo energético.

As soluções semelhantes existentes no mercado acabam por exigir algum tipo de interação com o utilizador, ou seja, o produto identifica unicamente o motivo ou o local onde existe um gasto anormal de energia, notifica o utilizador e dá sugestões acerca da forma de minimizar esses mesmo impactos. Nenhum conta com um sistema de gestão automática, ou seja, o utilizador tem que verificar os dados ou as dicas indicadas por esses produtos e tomar medidas para reduzir o consumo. Acaba por ser um processo que apesar de trazer vantagens ao utilizador é manual e exige a sua interação.

Estes tipos de soluções acabam por ser pouco práticas, e se o utilizador não conseguir agir no momento tornam-se inúteis.

O que se pretende com este projeto é ir além das soluções existentes tornando o sistema mais autónomo e que consiga economizar mais energia. Por estes motivos optou-se por um sistema que usa sensores de vários tipos, e que se propõe obter melhor resultados no que diz respeito à economia energética.

3.1 Visão Global

Na Figura 7 é possível ver um diagrama de contexto, que tem por objetivo demonstrar a vista de contexto do sistema. O utilizador interage com um único sistema que é o sistema de gestão energética.

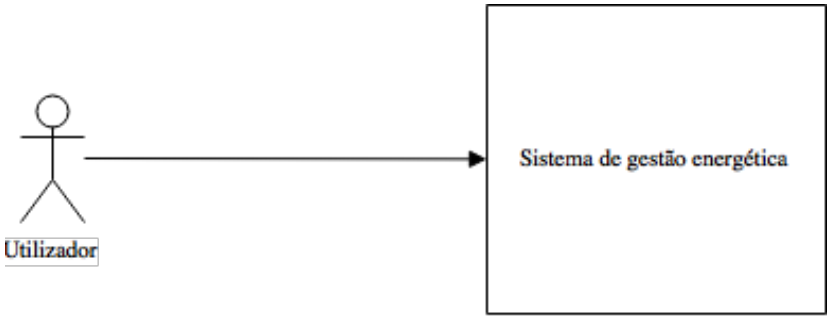

Figura 7 - Diagrama de contexto

O sistema de gestão energética engloba os dispositivos que tem acoplados sensores, o servidor e a central. O utilizador pode visualizar os dados de utilização imitidos pelos vários dispositivos. Na Figura 8 é possível ver um exemplo desse sistema, em que num determinado espaço estão distribuídos vários dispositivos que podem ter um ou vários sensores conectados, também é visível um equipamento central que recebe os valores enviados pelos dispositivos e os reencaminha para o servidor.

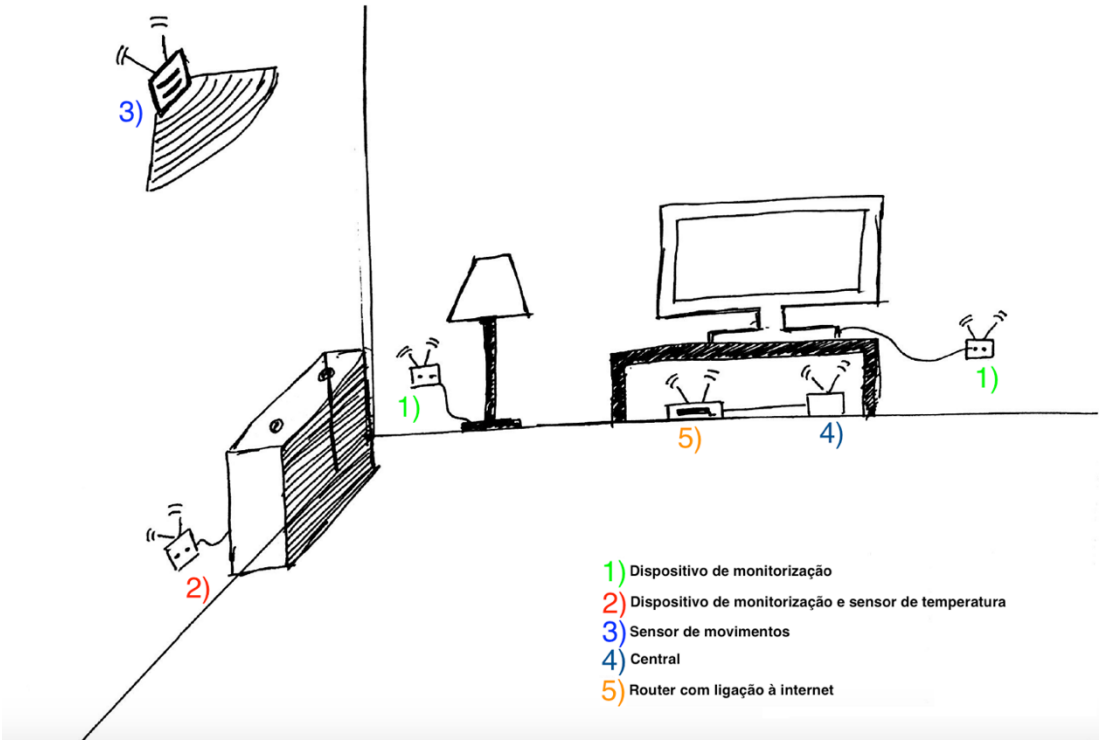

Figura 8 - Exemplo do sistema

Cada dispositivo, tem ligados vários sensores de diversos tipos que se encontram numa determinada área, ligados a equipamentos eletrónicos, para a monitorização do consumo energético num determinado instante.

Os sensores (por exemplo, de temperatura e movimento) fazem a leitura de vários parâmetros do espaço onde se encontram e notificam o dispositivo. Esse dispositivo através

de algum tipo de ligação que permita a comunicação, seja com ou sem fios, envia os diversos valores dos sensores para um equipamento central.

A central recebe os vários dados e deverá enviá-los para um servidor, que os analisa, e identifica uma ação que tenha de ser tomada em determinado momento. Essas ações têm como intuito aumentar a eficiência energética, desligando ou ligando os equipamentos.

Em suma, este sistema poderá ser dividido em três módulos principais:

- Sensores de monitorização, em que existe a gestão dos dispositivos que fazem a supervisão do espaço, isto é, os dispositivos devem ser capazes de recolher os dados dos sensores que estão conectados.
- Recolha dos dados, em que os dispositivos devem comunicar diretamente com um equipamento central os valores obtidos. Essa central deve fazer uma gestão dos vários dispositivos e enviar os valores para um servidor.
- Análise dos dados, onde será feita a gestão de energia através dos valores recebidos pelo servidor em que será possível tomar decisões de desligar ou ligar o equipamento com vista à poupança de energia.

Esses módulos representam as fases pelas quais será desenvolvido o projeto.

Na Figura 9 é apresentada uma representação do sistema com os diferentes intervenientes.

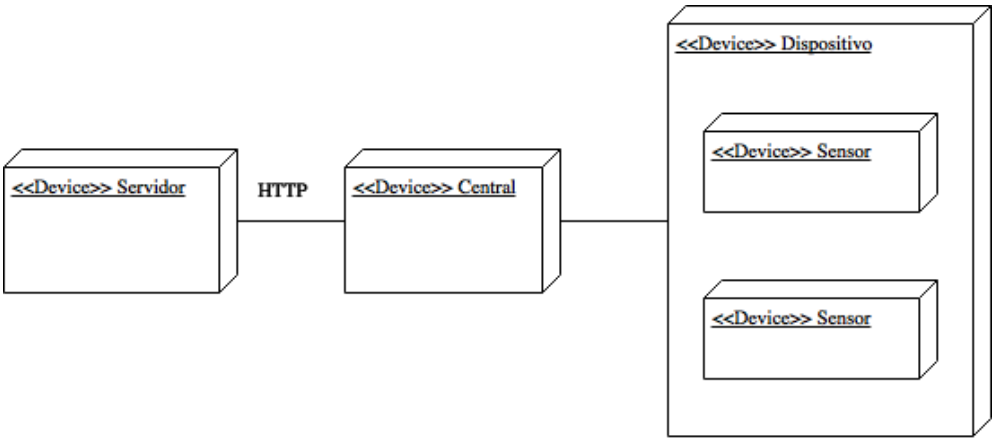

Figura 9 - Representação do sistema

Nessa representação é visível um servidor conectado a uma central que se encontra ligada a um dispositivo que tem 2 sensores acoplados.

3.2 Sensores

Os sensores a serem utilizados comunicam por uma interface com o dispositivo. Existem várias interfaces disponíveis, I2C, UART ou SPI, mas a escolha vai ser dependente da placa utilizada e se suporta esse tipo de ligação. Após serem ligados, os sensores enviam através de uma dessas interfaces os valores recolhidos.

3.3 Dispositivos

Os dispositivos serão placas programáveis onde podem ser conectados os sensores, através das interfaces referidas anteriormente. Estes dispositivos devem ser programados de forma a efetuarem a leitura dos sensores e enviar esses valores através de uma tecnologia de comunicação. Independentemente da tecnologia utilizada, o dispositivo também deve ser capaz de receber dados para, por exemplo, através do uso de uma relé, cortar o fornecimento de energia elétrica a um equipamento eletrónico que se encontre ligado a um sensor monitorização energética.

3.4 Central

A central tem como finalidade a recolha de informação imitida pelos vários dispositivos para posteriormente executar o envio para o servidor, da mesma forma que, efetua o envio dos dados do servidor para os dispositivos. Será, portanto, um equipamento com a responsabilidade de fazer a ponte entre o servidor e o dispositivo.

3.5 Servidor

O servidor recebe toda a informação recolhida pela central, e terá como objetivo guardar esses dados, analisá-los assim como aplicar técnicas que permitam identificar formas de poupar energia.

3.6 Sumário

Neste capítulo foi apresentada uma proposta com o intuito de atingir os objetivos descritos para o projeto. Assim como uma visão global foram também descritos em traços gerais o equipamento necessário para criar o sistema. Em seguida é importante fazer uma análise da proposta de forma a realizar o levantamento dos requisitos do sistema.

4 Análise

Neste capítulo de análise é identificada a análise de valor do sistema, os *stakeholders*, atores, requisitos funcionais e por fim, os requisitos não funcionais.

4.1 Análise de valor

Com vista à inovação dos produtos de uma organização devem ser aplicadas técnicas para uma melhor gestão dos mesmos com o objetivo de ajudar as empresas na adoção de práticas que permitam aumentar a diferença em relação a outros produtos concorrentes presentes no mercado. Esse processo é divido em três fases [71]:

1. *Fuzzy Front End.* (FFE);
2. *New Product Development.* (NPD);
3. Comercialização.

O New Concept Development (NCD) foi criado com o objetivo de ser um modelo de gestão eficaz para o FFE [72]. Este modelo que tem um motor (*engine*) que faz a gestão de cincos processos chave:

- Identificação da oportunidade;
- Análise da oportunidade;
- Geração e enriquecimento da ideia;
- Seleção da ideia;
- Definição de conceito.

Na identificação da oportunidade é onde a organização indica as oportunidades que pode desejar atingir e é direcionado pelos objetivos de negócio. Uma oportunidade pode ser uma melhoria a um produto já existente ou um produto completamente novo [73]. Hoje em dia existem produtos que permitem ao utilizador efetuar poupanças em alguns dos seus gastos

diários. Os custos podem variar, assim como o número de meses que temos que utilizar o produto até se obter lucro comparativamente ao que foi gasto. Com todos estes dados é possível verificar que existe uma oportunidade de mercado e ainda é um mercado que promete crescer.

Tem sido uma aposta de empresas como a Discovergy [68] criar sistemas capazes de guardar os dados de utilização de modo a fornecer ao utilizador uma perspetiva mais real da sua utilização.

Como análise de oportunidade para este produto, e através da recolha de informação de diversos tipos de forma a tornar a identificação de oportunidade numa oportunidade concreta de negócio [73], foram identificados os gastos com a energia elétrica como a oportunidade mais concreta. Estes gastos incluem tanto os custos para o utilizador como os recursos renováveis ou não renováveis que são utilizados.

Na geração e enriquecimento da ideia é onde se faz a maturação e desenvolvimento da ideia em que se inclui um processo de iteração de ideias onde são construídas e atualizadas de maneira a desenvolverem-se [73].

Depois de identificada e analisada a oportunidade de negócio, são geradas várias ideias possíveis para resolver o problema. Podem existir várias maneiras de abordar o problema, por exemplo, reduzir os consumos energéticos reduzindo a luminosidade da iluminação, notificar o utilizador sempre que existirem gastos excessivos ou anormais dentro de determinado espaço ou criar um sistema capaz de fazer a gestão automática de todos os aparelhos eletrónicos que se encontram num determinado espaço.

Na maioria das organizações existem tantas ideias para novos produtos ou para novos processos que é complicado encontrar a que é melhor em termos de negócio. São utilizados modelos de modo a considerar várias variáveis, desde tecnológicas a níveis de investimento necessário, de modo a escolher a ideia mais acertada [73]. Neste produto, foi escolhida a ideia de criar um sistema que faça a gestão automática de todos os equipamentos eletrónicos utilizando dispositivos que possam cortar o fornecimento de energia elétrica.

Devido ao conhecimento que é necessário para o seu desenvolvimento, e pela abrangência que este produto traz, não se encontra limitado a um só equipamento eletrónico. Foi possível através da junção de várias tecnologias pensar num produto com funcionalidades únicas, que não necessita de interação com o utilizador e fornece ganhos consideráveis no que diz respeito aos custos energéticos.

Na definição de conceito existe o desenvolvimento de um modelo de negócio onde são tidos em conta vários parâmetros como necessidades do cliente, investimento, desconhecimentos de tecnologia entre outros [73]. O objetivo passa por desenvolver um produto capaz de trazer poupanças significativas para o utilizador com um custo baixo. Este produto tem o intuito de realizar a gestão dos vários equipamentos eletrónicos dispersos por um espaço, de forma a diminuir o consumo energético geral de todo o espaço que é gerido.

O valor é um conceito caracterizado por ser um processo de criação de valor que acontece entre a empresa e o cliente [74]. Isto é, o valor que a empresa pode oferecer a um cliente. O valor percebido (*Perceived value*) apareceu em 1990 como sendo o principal problema de negócios [75].

O valor do produto aqui apresentado é definido pela facilidade de utilização e que proporciona uma redução dos custos diários do consumidor.

O valor percebido do produto pode ser consultado na Tabela 3 e tem por objetivo identificar em função do tempo os benefícios e sacrifícios que o produto tem.

	BENIFÍCIOS	**SACRÍFICIOS**
ANTES DA COMPRA	Poupança Otimização energética	Escolha da melhor solução
NO MOMENTO DA COMPRA	Poupança Otimização energética	Custo do software Custo dos dispositivos
APÓS A COMPRA	Qualidade do Produto	Familiarização com o método de utilização
APÓS UTILIZAÇÃO	Redução de custos com energia Diminuição dos recursos energéticos utilizados Aumento da vida útil dos equipamentos	-

Tabela 3 - Perspetiva Longitudinal

No momento antes da compra existe uma escolha por parte do utilizador da melhor solução para obter as poupanças desejadas e existe um valor percebido como é o caso da poupança e da otimização energética.

No momento da compra tem como contra os custos do *software* e dos dispositivos que são necessários para suportar o *software*.

Em seguida, após a compra existe uma familiarização com o método de utilização e com funcionamento do produto. Existe também um benefício com a qualidade que o produto tem.

Depois de concluído esse pequeno processo de aprendizagem para a utilização, e após a sua utilização foram identificados alguns benefícios como a redução de custos e de recursos energéticos.

O valor do produto para o consumidor é a poupança que pode originar com a sua utilização. Essa poupança tanto ao nível monetário como à duração de vida dos próprios equipamentos a que for ligado.

Na Tabela 4 é possível consultar o modelo Canvas e obter mais informação do modelo de negócio pretendido para este produto.

<table>
<tr>
<td rowspan="2">Parceiros chave
• EDP
• Iberdrola
• Galp Energia
• DGEG (Direção Geral de Energia e Geologia)
Estes parceiros podem ajudar na divulgação do produto.</td>
<td>Atividades Chave
• Software
Desenvolvimento contínuo do software para o tornar mais eficiente.
• Hardware
Todo o equipamento usado para a tomada de decisão deve ser o mais preciso e fiável possível.</td>
<td rowspan="2">Proposta de valor
• Sistema de apoio à gestão energética.
O produto é o conjunto de Hardware se Software que permite fazer a gestão dos equipamentos eletrónicos.
• Poupança de energia
Os consumos serão reduzidos desligando os equipamentos quando não são necessários.
• Gestão automática
Os equipamentos são ligados/desligados de forma automática.</td>
<td>Relação com os clientes
• Suporte
É dado suporte ao cliente através de email.
• Sugestões
São dadas sugestões ao utilizador com base nas estatísticas do consumo, por exemplo, para trocar determinado equipamento.</td>
<td rowspan="2">Segmentos de Clientes
• Consumidores com grandes custos energéticos
Esses consumidores irão reduzir os seus custos.
• Consumidores com preocupações ambientais
A utilização do produto reduz o uso de recursos energéticos.</td>
</tr>
<tr>
<td>Recursos chave
• Equipa de desenvolvimento de Software
Onde se atualiza o produto com novas funcionalidades.
• Parcerias com empresas de desenvolvimento de sensores
Estas empresas são importantes para uma melhoria continua de todos os dispositivos que ajudam na gestão energética.</td>
<td>Canais
• Website informativo
Onde se explica o produto.
• Redes sociais
Onde se promove e explica o produto.
• Lojas da especialidade (Worten, Fnac, entre outros)
Onde se vende e promove o produto.</td>
</tr>
<tr>
<td colspan="3">Estrutura de Custos
• Custos do servidor
Domínio e custos de alojamento.
• Equipa de suporte
Uma equipa preparada para oferecer suporte ao cliente.
• Equipa de desenvolvimento de software.
Uma equipa para continuamente desenvolver o software e novas funcionalidades.</td>
<td colspan="3">Fluxo de Receitas
• Licença anual de suporte
O utilizador obtém uma licença de um ano do software e durante esse período recebe todas as atualizações que surgirem, após esse ano deixa de receber suporte. Caso seja pretendido pelo cliente, esta licença pode ser vendida em conjunto com o hardware necessário.</td>
</tr>
</table>

Tabela 4 - Modelo Canvas

Empresas ligadas ao setor Energético foram vistas como parceiros chaves, assim como a Direção Geral de Energia e Geologia (DGEG), que é um órgão da Administração pública Portuguesa, que tem também como propósito a diminuição das necessidades de recursos não renováveis e de desperdícios energéticos. Todos podem ajudar na expansão e divulgação do produto. Empresas como, a EDP tem preocupações ambientais e de sustentabilidade [76], e mostram interesse em encontrar soluções que ajudem na diminuição do desperdício energético.

Como atividade chave foram identificados o *software* e *hardware* utilizados para conseguir construir o sistema. O *software* deve receber atualizações com vista a melhorias na gestão e o *hardware* deve ser o mais fiável e preciso possível.

Os recursos chave necessários para desenvolver as atividades chave são uma equipa de desenvolvimento de software que vai permitir melhorias contínuas ao produto e parcerias com empresas de desenvolvimento de sensores que seria importante para obter Hardware que se adaptasse ao Software para uma melhor gestão.

A proposta de valor do produto é a poupança de energia e a gestão automática. O produto faz a gestão automática de vários equipamentos, desligando-os quando não são necessários e assim reduz os recursos energéticos utilizados. Equipamentos que utilizem baterias poderão ter um aumento da vida útil, pois o sistema será competente para desligar os mesmos quando se encontrarem carregados, diminuindo assim a degradação da bateria.

A relação com o cliente será feita através de uma equipa de suporte, disponível para ser contactada por email e ajudar na resolução de problemas. Serão enviados ao longo do tempo emails para o utilizador com sugestões para uma maior poupança de recursos energéticos com base nos consumos que costuma efetuar.

Como canais foram identificados um website para informar o possível cliente, redes sociais para promover o produto e lojas da especialidade onde é possível também promover e vender.

No que diz respeito aos segmentos de clientes, os principais interessados serão os consumidores com grandes consumos energéticos, com preocupações ambientes e empresas que desejem reduzir os custos energéticos ou que usem o sistema em espaços públicos ou o forneçam a clientes de forma a diminuir os custos. Estes segmentos de clientes também podem ser considerados como *stakeholders*[77], ou seja, as partes interessadas no projeto.

Relativamente à estrutura de custos, vão existir custos com o servidor, com a equipa que dá suporte ao utilizador e com o desenvolvimento de software. Em relação às receitas o produto será vendido com ou sem hardware com uma licença de utilização por um ano. No fim desse ano o produto poderá continuar a ser usado, mas sem mais atualizações.

4.2 Requisitos funcionais

Os requisitos funcionais descrevem as funcionalidades do sistema, indicando como o sistema deve reagir ou se comportar em situações particulares. Para indicar as diversas interações com o sistema é necessário numa primeira fase identificar os atores. Estes atores podem ser sistemas ou pessoas que interagem com o sistema [78], e para este projeto são:

- Os dispositivos que têm sensores acoplados.
- A central que faz a gestão desses dispositivos.
- O servidor que recebe e armazena os dados enviados pela central.
- O utilizador do sistema.

Os dispositivos interagem com a central, assim como a central com o servidor, por fim o utilizador vê resultados através de uma página Web.

Através do diagrama de casos de uso apresentado na Figura 10, é possível observar as interações entre os diferentes atores.

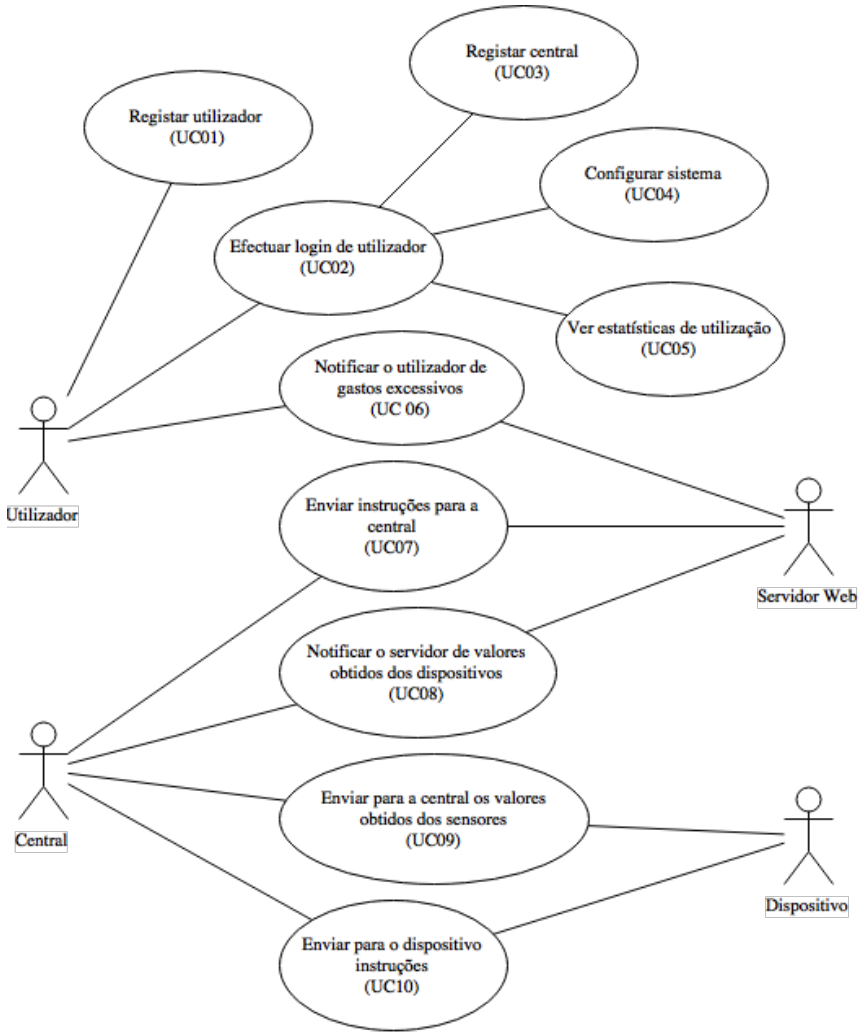

Figura 10 - Diagrama de casos de uso

O utilizador pode registar-se na plataforma e para isso deve introduzir alguns dados como o nome de utilizador, morada, localidade, email e password. Depois disso terá possibilidade de fazer login com o email e a password com que se registou. Na Figura 11 é possível ver um diagrama de sequência do sistema (SSD) que representa este caso de uso.

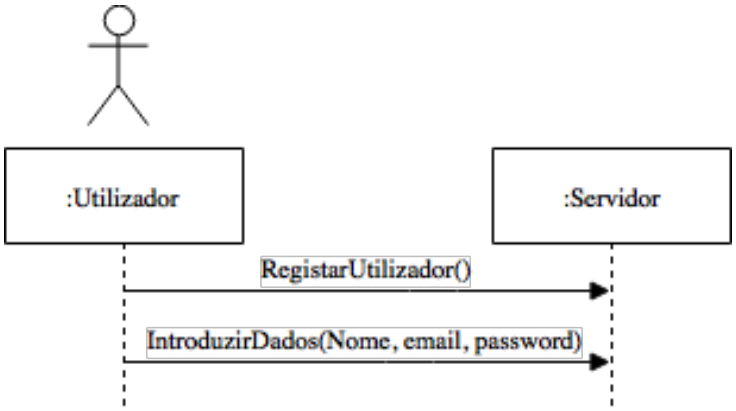

Figura 11 - SSD do caso de uso 01

Quanto ao login, o utilizador deve introduzir o email e a password com que se registou, como se pode ver na Figura 12. Depois disso será mostrado o menu principal onde poderá configurar o sistema, ver estatísticas de utilização e registar novas centrais.

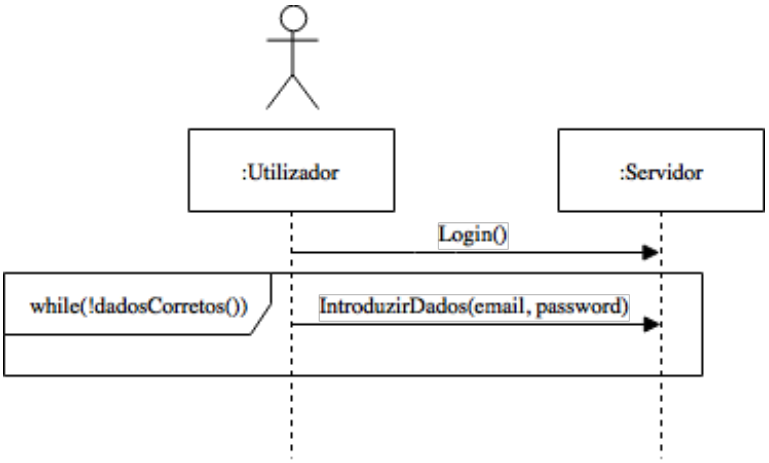

Figura 12 - SSD do caso de uso 02

Caso pretenda registar a central o utilizador deve introduzir um identificador que se encontra junto com a central, esse fluxo pode ser visto na Figura 13.

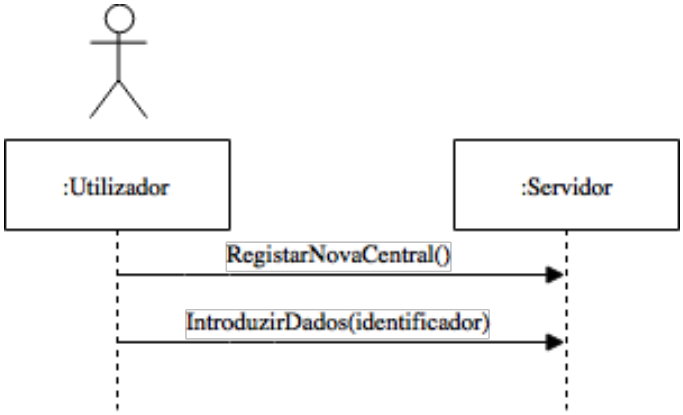

Figura 13 - SSD do caso de uso 03

O utilizador pode também configurar o sistema, isto é, adicionar dispositivos que se encontrem no espaço através da introdução do identificador que se encontra nos próprios. Caso o dispositivo tenha a capacidade de efetuar a monitorização energética pode indicar o tipo de equipamento que se encontra a ser monitorizado e identificar o tipo de disponibilidade do sistema que pretende, ou seja, permitir que esse equipamento possa ser desligado mais vezes tendo, portanto, uma menor disponibilidade, esse fluxo é visível na Figura 14.

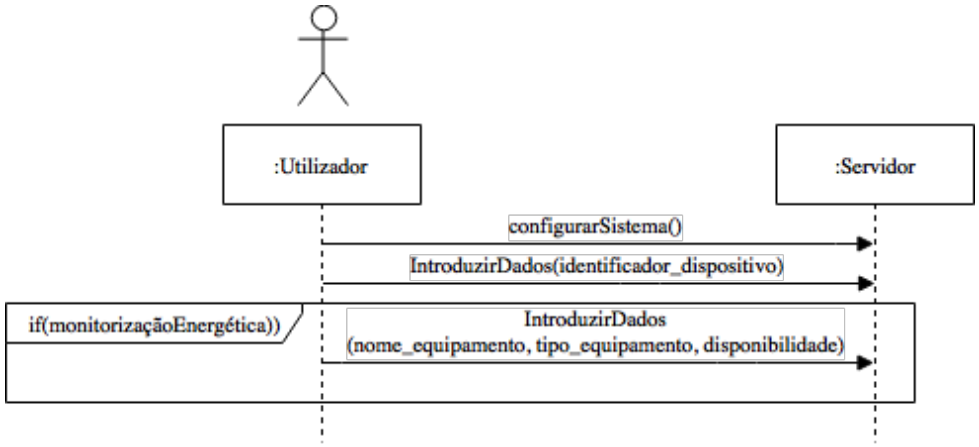

Figura 14 - SSD do caso de uso 04

O sistema depois de receber alguns valores dos dispositivos gera gráficos de utilização e apresenta os dados em tabelas de forma a uma mais fácil análise. Na Figura 15 é possível ver um SSD que representa esse caso de uso.

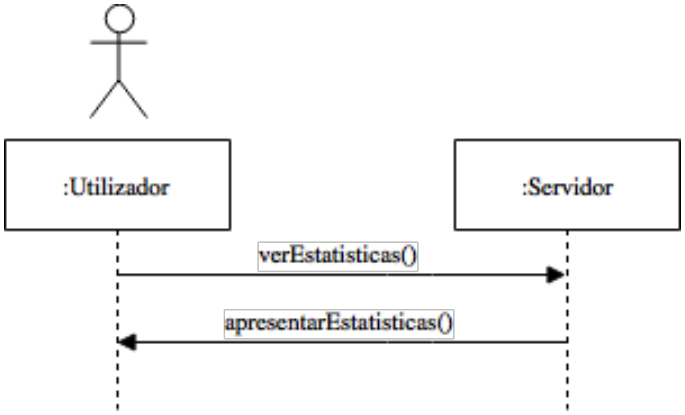

Figura 15 - SSD do caso de uso 05

Sempre que o servidor mediante um algoritmo que analisa continuamento os dados recebidos, detetar variações anómalos dos dados emitidos pelos sensores, são enviados emails para o utilizador, tal é representado na Figura 16.

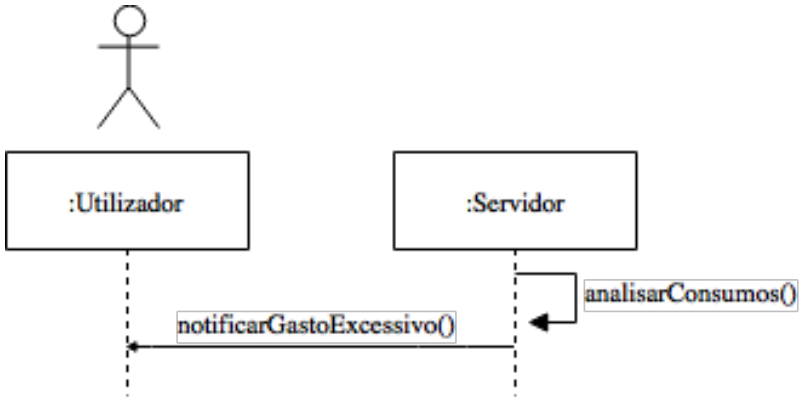

Figura 16 - SSD do caso de uso 06

Além disso o, servidor envia instruções para a central, por exemplo, para determinado dispositivo cortar a corrente energética, como é possível ver na Figura 17. Essas instruções sobre dados recebidos e processados pela central são reenviadas para o dispositivo correto.

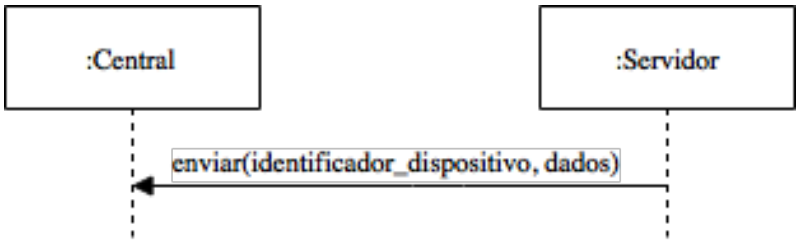

Figura 17 - SSD do caso de uso 07

A operação inversa também é feita, isto é, os dispositivos ao longo do tempo recolhem dados e enviam-nos para a central. Quando esta os recebe, envia cada um desses valores para o

servidor, através de pedidos HTTP, de forma a serem persistidos. A Figura 18 representa essa interação.

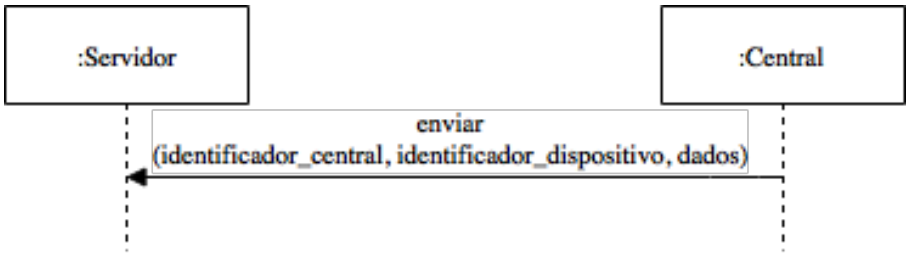

Figura 18 - SSD do caso de uso 08

A central pode também receber dados dos dispositivos relativos a leitura dos diversos sensores e que depois de recolhidos são enviados à central, como é visível na Figura 19.

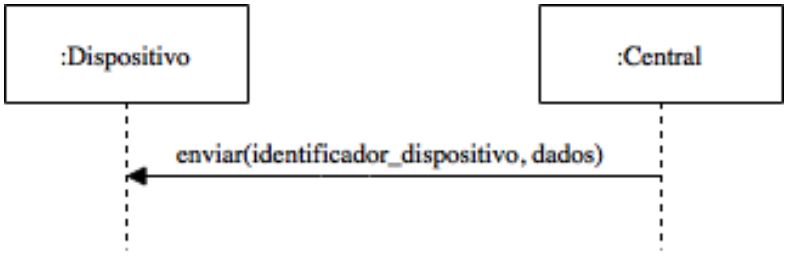

Figura 19 - SSD do caso de uso 09

Por fim, a central deve poder enviar para o dispositivo informação com instruções a serem executadas de forma a adaptar o dispositivo, por exemplo, cortar a corrente elétrica a um determinado equipamento que está acoplado a esse dispositivo. O SSD correspondente a esse caso de uso é mostrado na Figura 20.

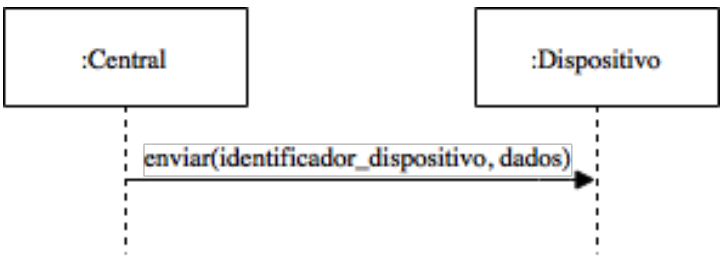

Figura 20 - SSD do caso de uso 10

4.3 Requisitos não funcionais

Os requisitos não funcionais são a descrição de atributos que o *software* deve ter de forma a cumprir com as tarefas exigidas [78]. Para o sistema ser capaz de cumprir todas as tarefas de forma precisa e correta deve ter em conta as seguintes exigências:

- **Transmissão de dados:** Este sistema obriga á troca de informação entre vários dispositivos, informação essa que pode ser enviada através de tecnologias com ou sem fios. Essa comunicação deve ser feita através do uso de tecnologias inovadoras e que tenham um gasto reduzido de energia.
- **Consumos energéticos:** É importante que todos os equipamentos e tecnologias envolvidas no sistema exijam o mínimo possível de energia para executarem as tarefas necessárias. Quanto maior o consumo dos equipamentos necessários para o sistema funcionar plenamente, mais difícil será reduzir os consumos do espaço. Para garantir uma redução de consumo energético, todo o hardware deve consumir valores de energia muito baixos que, por exemplo no caso dos dispositivos, a recolha dos dados dos sensores permite uma redução de consumo suficiente para compensar ter esse dispositivo ligado a monitorizar o espaço.
- **Usabilidade:** A interface a que o utilizador tem acesso deve ser prática e intuitiva de modo a um mais fácil acesso aos dados, e em que possa tirar real proveito dos dados estatísticos recolhidos.
- **Adaptabilidade:** O sistema deve ser capaz de se adaptar a diferentes espaços onde se encontre instalado, adaptando a gestão energética ao mesmo.
- **Sensores**: Os dispositivos podem ter acoplados sensores que permitam monitorizar o consumo energético em tempo real, assim como outros que possam ser usados para uma melhor gestão de energia.

Os requisitos devem ser tidos em conta no momento do desenho e da implementação, para que o sistema seja capaz de os suportar.

4.4 Sumário

Ao longo deste capítulo foi feita uma análise do sistema, a que inclui a apresentação dos requisitos funcionais e não funcionais. Esses elementos são essenciais para o desenho e posterior implementação do sistema.

5 Desenho

Este capítulo tem por propósito descrever e mostrar uma solução que permite sustentar de forma coerente todas as funcionalidades que foram propostas anteriormente.

5.1 Implantação da Proposta

O diagrama de *deployment* visível na Figura 21, descreve a configuração do ambiente durante a execução do sistema.

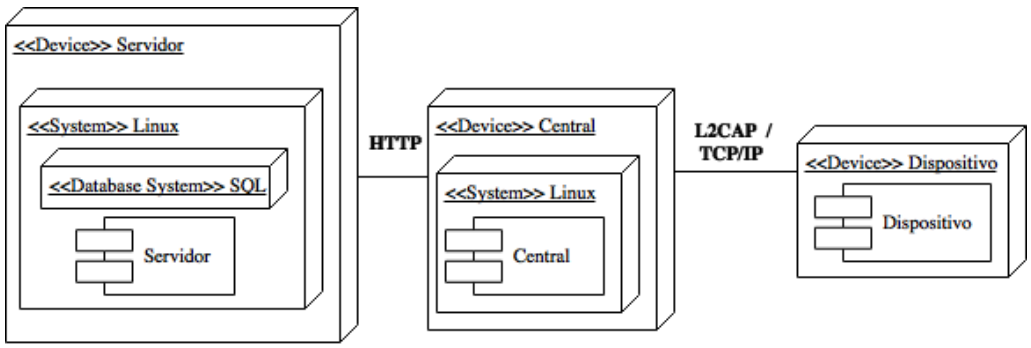

Figura 21 - Diagrama de *deployment*

Conforme se pode observar, o servidor vai executar uma distribuição de Linux em que vai ter incorporada uma base de dados SQL. A comunicação com a central será feita através do protocolo REST. A central vai executar uma distribuição de Linux e comunica, através do protocolo REST, com o servidor para enviar os vários dados obtidos dos dispositivos. O dispositivo vai comunicar com a central através de comunicações por Bluetooth ou Wi-Fi (L2CAP ou TCP/IP respetivamente) de forma a enviar os valores que vão sendo obtidos dos vários sensores que tem acoplados.

5.2 Componentes

O sistema tem vários componentes representados no diagrama de componentes da Figura 22.

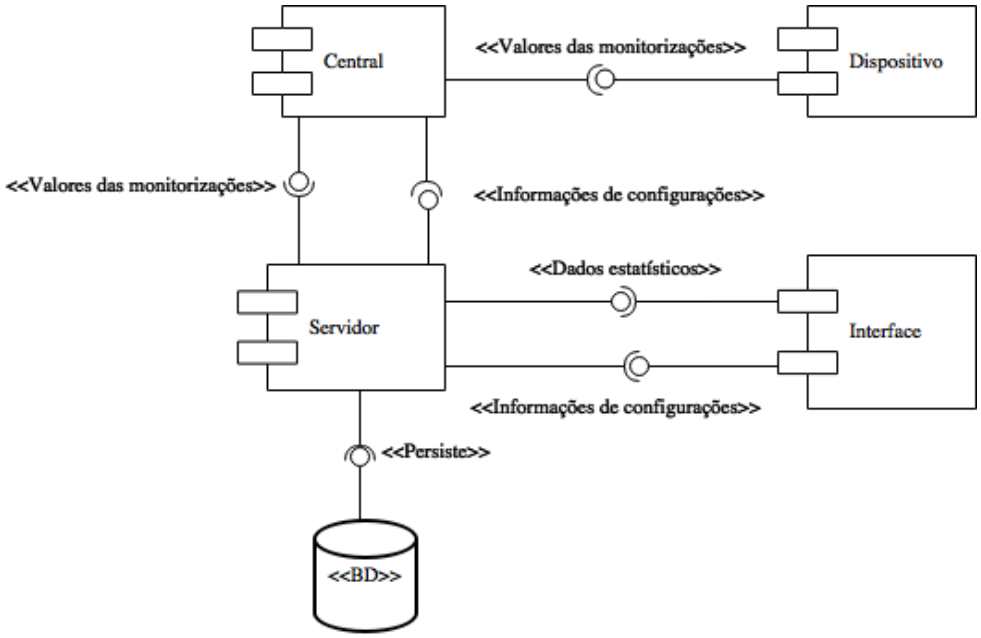

Figura 22 - Diagrama de componentes de 1º vista

O servidor tem uma interface e persiste os dados na base de dados, fornece dados estatísticos que forem pedidos pela interface, obtém dados das monitorizações da central e fornece informações de configurações que obteve da interface. A interface é responsável por mostrar os dados fornecidos pelo servidor. Quanto à central, obtém dados do servidor, por exemplo, configurações do utilizador e envia também para este servidor, dados relativos ao funcionamento energético dos equipamentos que se encontra a monitorizar. Tem uma interface disponível para os dispositivos, para monitorizar os valores que estes transmitam. Os dispositivos são responsáveis pela monitorização dos equipamentos, por isso comunicam com a central para o envio dos valores recolhidos. Na Figura 23 é visível um diagrama de atividades que visa mostrar o fluxo de uma atividade, neste caso, a leitura de dados de um sensor até à tomada de decisão e consequente ação.

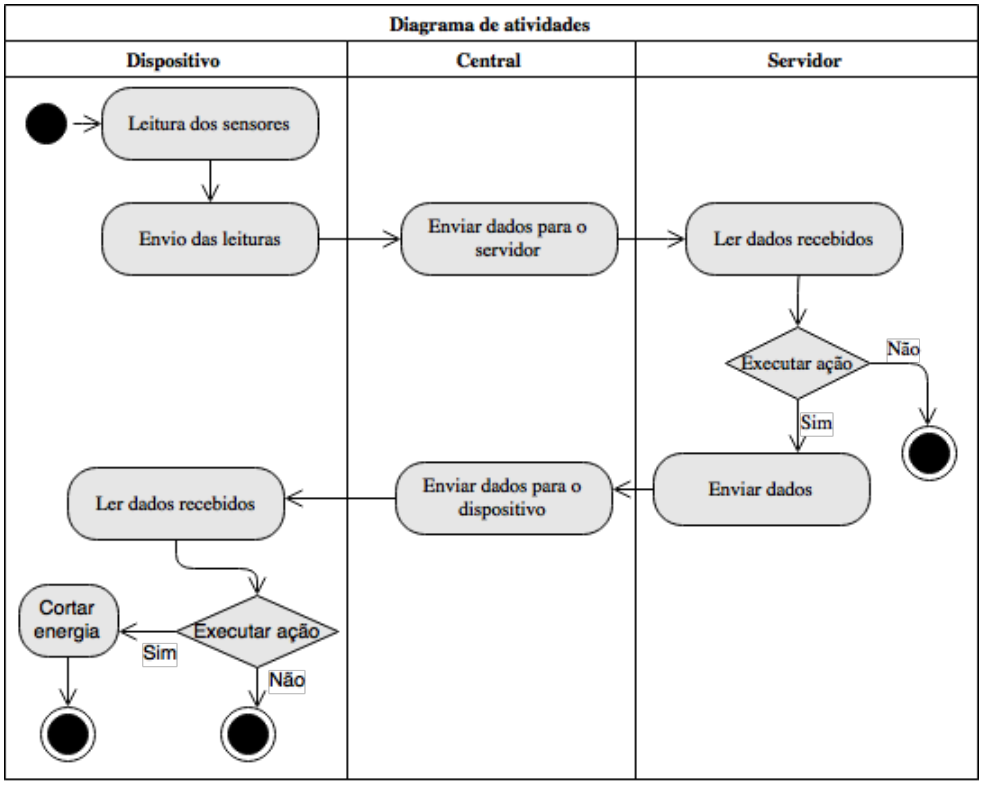

Figura 23 - Diagrama de atividades

O dispositivo lê um valor do sensor, envia a leitura para a central que procede ao reencaminhamento dos dados até ao servidor. O servidor analisa os dados recebidos e através de um algoritmo deteta se deve tomar uma ação, caso aconteça, uma mensagem é enviada até à central e reencaminhada até ao dispositivo que lê a mensagem e verifica se é uma mensagem para executar a ação de cortar a energia.

5.2.1 Dispositivo

Nesta secção serão apresentados vários diagramas abstratos para uma implementação do sistema nos dispositivos, utilizando qualquer uma das tecnologias referidas anteriormente como possíveis de serem utilizadas, tanto a nível de *hardware* (Arduino, Waveshare), como a nível de tecnologia de comunicação (Bluetooth, Wi-fi).

As principais funcionalidades necessárias no dispositivo são:

- Gerir o dispositivo de forma a conectar-se a uma rede (Bluetooth, Wi-Fi ou outra).
- Comunicar os valores a partir de uma determinada tecnologia.
- Receber valores que indiquem a forma como deve operar (Ex: cortar a corrente elétrica).
- Gestão das leituras dos vários sensores que podem estar acoplados.

A partir destas funcionalidades foi criado o diagrama de componentes apresentado na Figura 24, onde são mostrados os vários *packages* para o sistema do dispositivo.

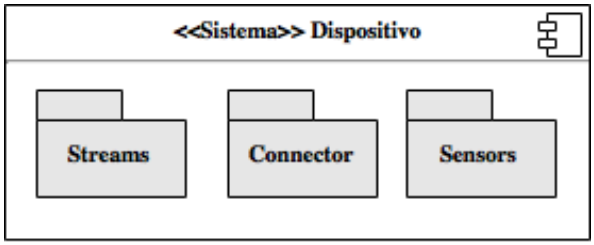

Figura 24 - Diagrama de componentes do dispositivo

Na Figura 25 é possível ver um modelo de domínio que dará origem ao diagrama de classes. Cada dispositivo tem um identificador único e pode ter um ou vários sensores conectados. Cada um desses sensores tem um tipo que o identifica (Ex: sensor de temperatura, movimento, proximidade) e um identificador que o torna único dentro de um dispositivo.

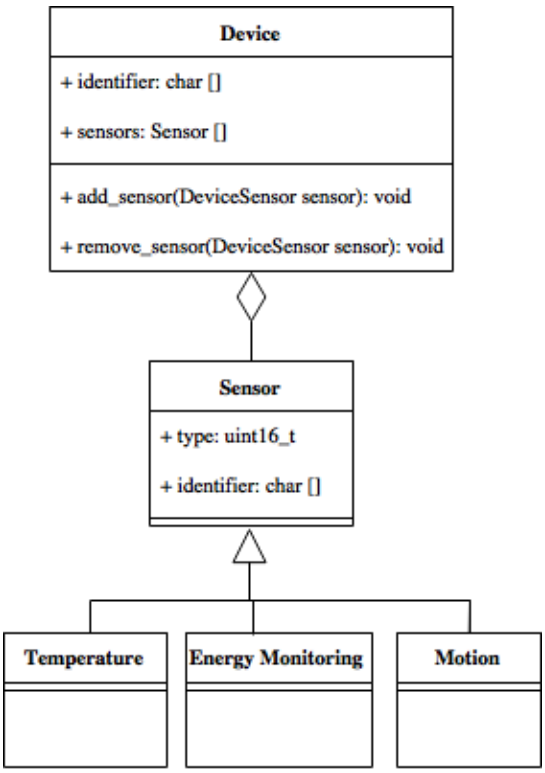

Figura 25 - Modelo de domínio do dispositivo segundo diagrama de classes (UML)

Da análise das funcionalidades em conjunto com o diagrama de componentes foi desenhado o diagrama de classes, representado na Figura 26.

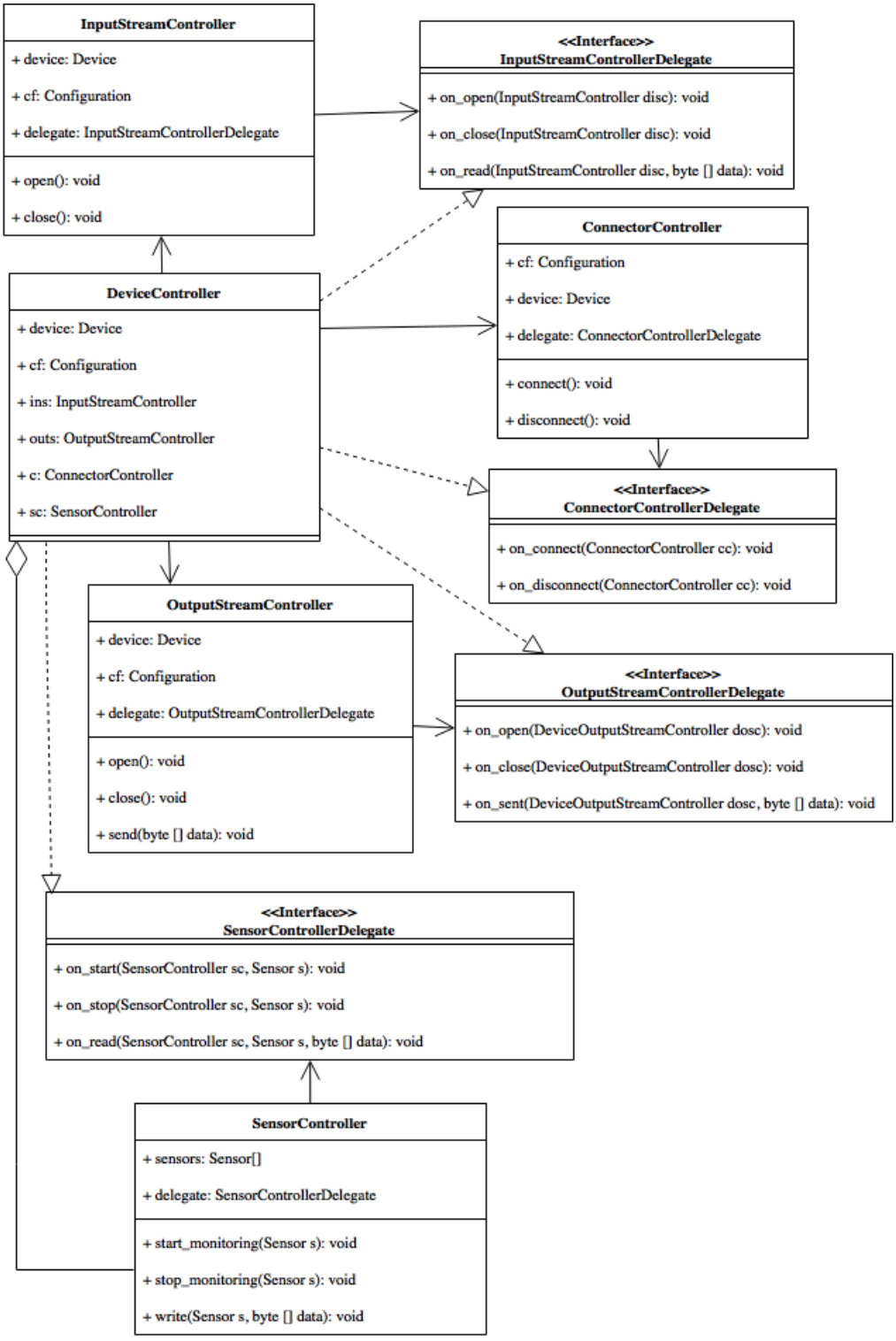

Figura 26 - Diagrama de classes do dispositivo

O diagrama foi baseado no padrão arquitetural MVC (Model View Controller), e este representa o controlador. Foi utilizado o padrão *delegate* de forma a conseguir notificar outras classes de eventos que acontecem. Existe um controlador do dispositivo que controla o envio e receção de dados através das classes *InputStreamController e OutputStreamController*.

Além dessas tarefas esse controlador tem a possibilidade de se conectar à rede através do *ConnectorController* ou de efetuar a gestão dos sensores pela classe *SensorController*.

É importante que a comunicação seja fiável e com baixos valores de latência para uma melhor análise do meio, aumentando assim a precisão e rapidez com que os dados são obtidos.

5.2.1.1 Conexão à rede

Cada dispositivo deve iniciar a conexão a uma rede para enviar mensagens em *broadcast* ou em determinadas tecnologias ser o ponto de acesso ao qual a central se deve ligar. Essa conexão é apresentada no diagrama da Figura 27. Para essa conexão são utilizadas as configurações que existem na classe *ConnectorController*.

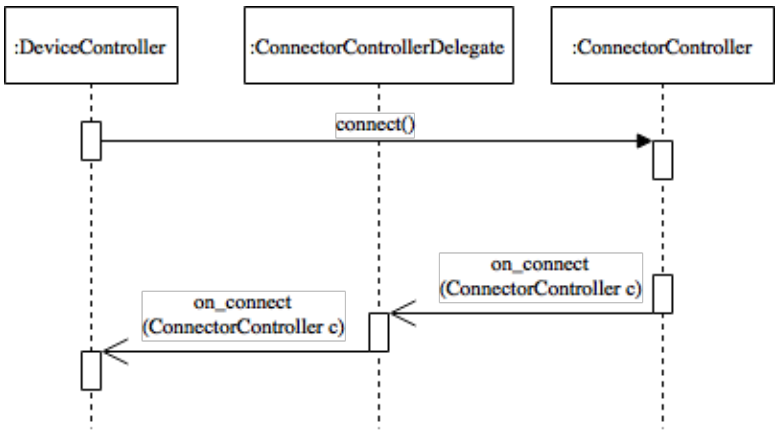

Figura 27 - Diagrama de sequência de conexão

5.2.1.2 Monitorizar sensor

Uma das responsabilidades do dispositivo é monitorizar os vários sensores que podem estar emparelhados. No diagrama de sequência representado na Figura 28, é possível ver a forma como é iniciada uma monitorização de um sensor.

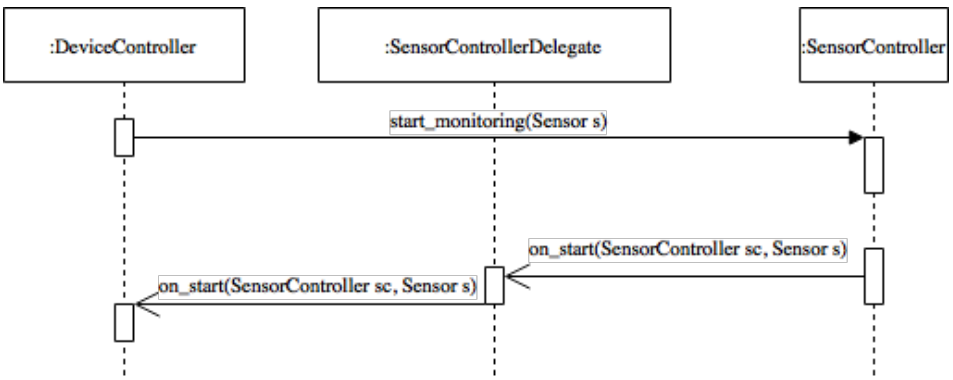

Figura 28 - Diagrama de sequência de início de monitorização de sensor

5.2.1.3 Receção dos valores dos sensores

Depois de iniciada a monitorização do sensor, o *DeviceController* fica à escuta dos valores obtidos pelos sensores, essa funcionalidade é representada na Figura 29. Os dados posteriormente serão enviados para a central.

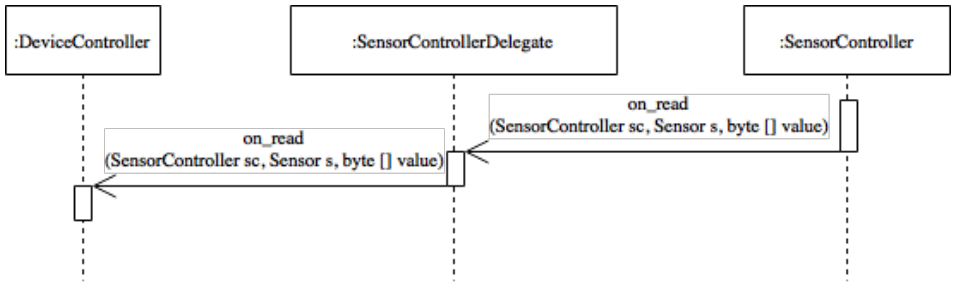

Figura 29 - Diagrama de sequência de receção de valores dos sensores

5.2.1.4 Envio de dados

O dispositivo depois de receber os valores no controlador, deve ter a capacidade de enviar os dados através da *stream* de *output* como é representado na Figura 30. Deve ser aberta a *stream* de comunicação e depois começar a transmissão dos dados.

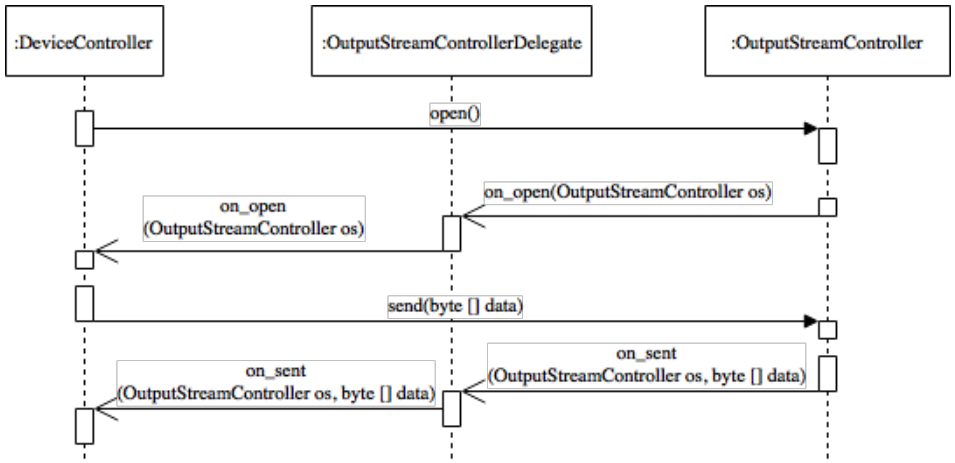

Figura 30 - Diagrama de sequência de envio de dados do dispositivo

5.2.2 Central

Nas próximas secções, serão apresentados diagramas que representam a forma como é desenhada a central, ignorando artefactos específicos à implementação.

Para o desenho da central foi necessário proceder ao levantamento das principais funcionalidades a serem implementadas que são:

- Controlar um ou vários dispositivos.
- Procurar por dispositivos.
- Conectar a dispositivos.

- Enviar e receber dados de dispositivos.

A partir dessas funcionalidades, foi desenhado o diagrama de componentes representado na Figura 31, onde são visíveis os vários *packages* para o sistema da central.

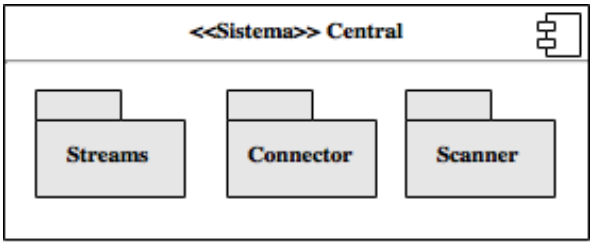

Figura 31 - Diagrama de componentes da central

Na Figura 32 é possível ver um modelo de domínio que é a representação do modelo que a central necessita de criar para armazenar os dispositivos encontrados. Cada dispositivo encontrado tem um ou vários sensores.

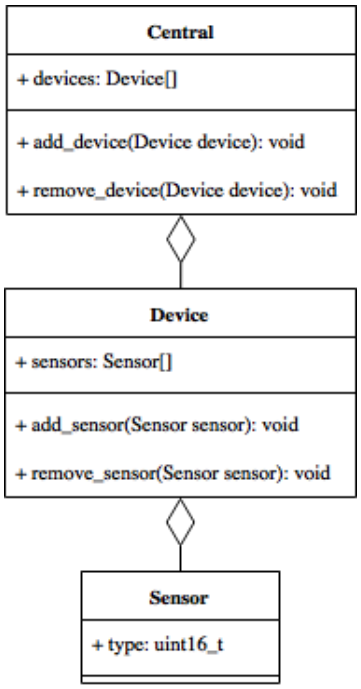

Figura 32 - Modelo de domínio da central segundo diagrama de classes (UML)

O diagrama de classes visível na Figura 33 representa as classes do controlador da central, e é possível ver em maior detalhe o que se pretenda desta implementação.

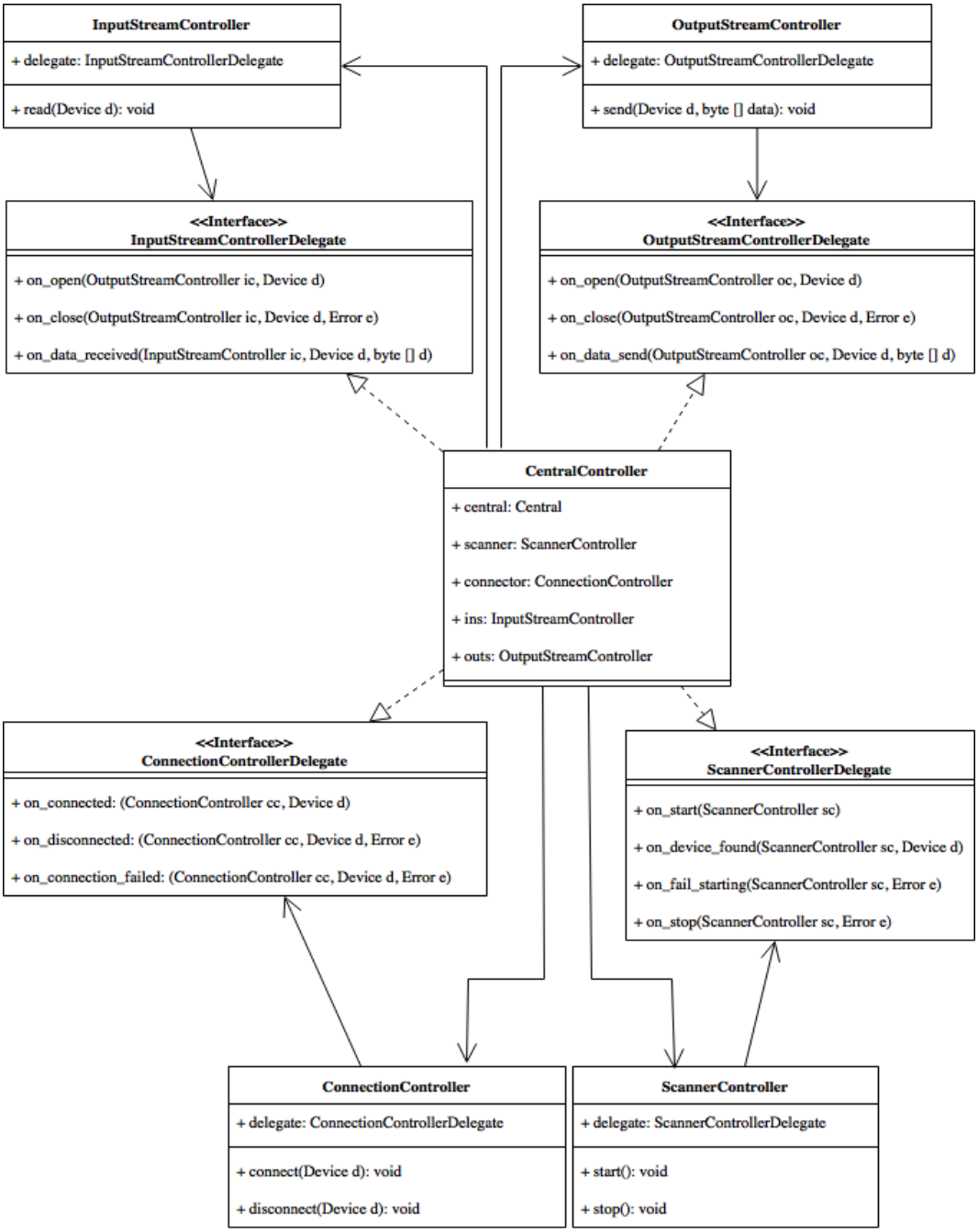

Figura 33 - Diagrama de classes da central

Tal como para o dispositivo, foi criado um controlador por cada módulo que é necessário implementar. Existe um *CentralController* que gere todos os módulos da central. O *ScannerController* permite pesquisar por dispositivos na proximidade enquanto que o *ConnectionController* tem a responsabilidade de ligar e desligar de um dispositivo ou rede, por fim, depois de estabelecida a conexão é possível utilizar o *InputController e OutputController,* que ficam com a responsabilidade de receber e enviar dados dos dispositivos respetivamente.

A implementação das funcionalidades referidas anteriormente, é descrita em maior detalhe nas secções seguintes.

5.2.2.1 Procurar por dispositivos

No diagrama apresentado na Figura 34 é mostrado a forma como é implementada a funcionalidade de procura de dispositivos por parte da central.

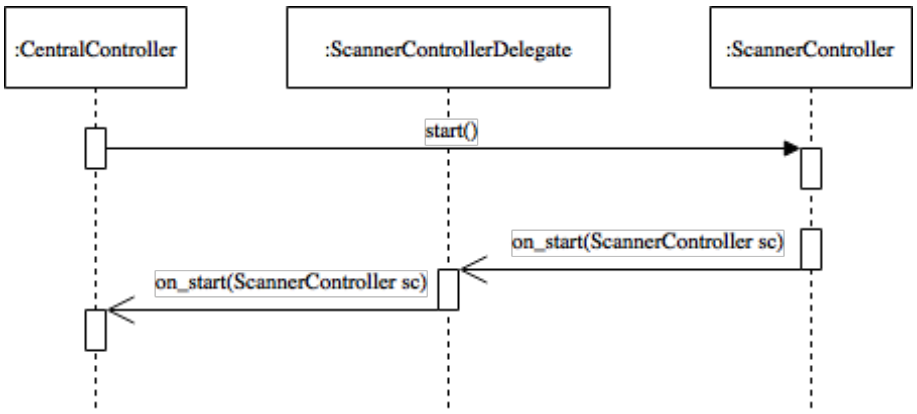

Figura 34 - Diagrama de sequência de procura de dispositivos

5.2.2.2 Conectar a dispositivo

Após um dispositivo ser encontrado, deve ser feita a conexão ao mesmo, para a partir desse momento começar a receber valores das leituras dos sensores. A sequência que representa essa funcionalidade é apresentada na Figura 35.

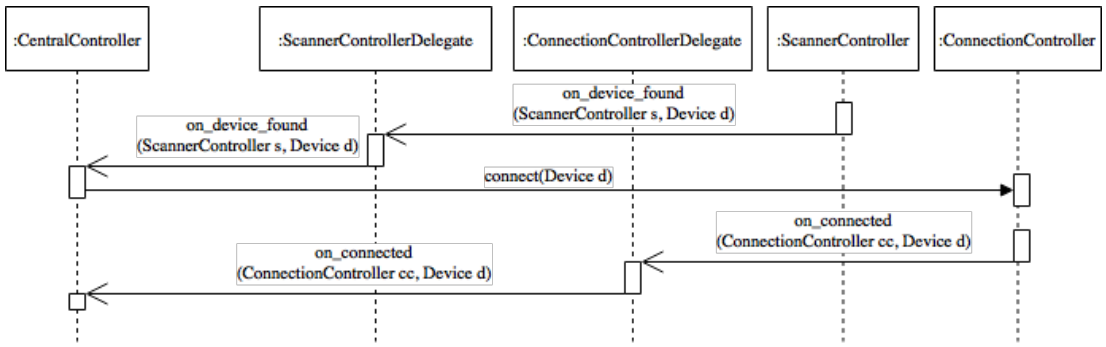

Figura 35 - Diagrama de sequência de conexão de dispositivos

5.2.2.3 Receber dados

Após a conexão do dispositivo ter sido concluída com sucesso, é necessário abrir a *stream* de *input* para a partir desse instante estar disponível para receber dados que sejam enviados. Essa representação é mostrada no diagrama da Figura 36.

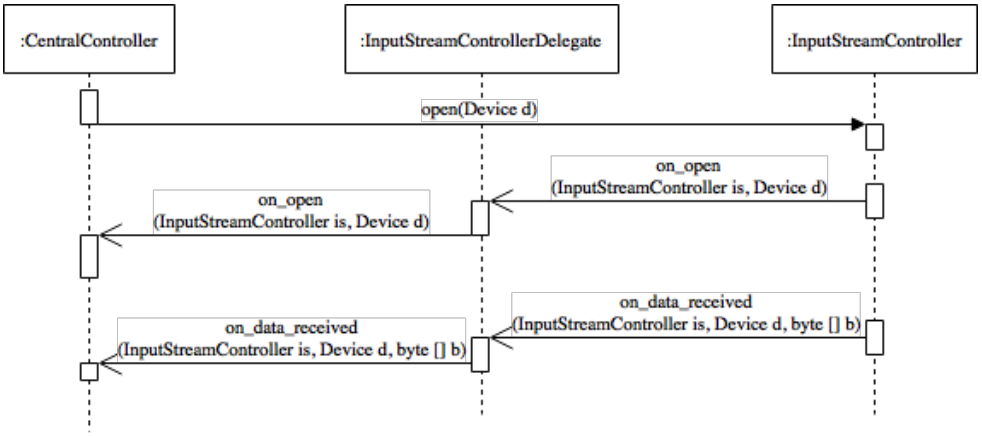

Figura 36 - Diagrama de sequência de receção de dados

5.2.3 Servidor

Nas secções seguintes são apresentados os vários diagramas com vista ao desenho do servidor. Para isso foi feito um levantamento das várias funcionalidades listadas em seguida:

- Registo e login de utilizadores.
- Comunicar bidirecionalmente com a central.
- Persistência dos dados que recebe da central.
- Tomar decisões.

Tal como para os sistemas anteriores, a partir destas funcionalidades foi criado o diagrama de componentes visível na Figura 37.

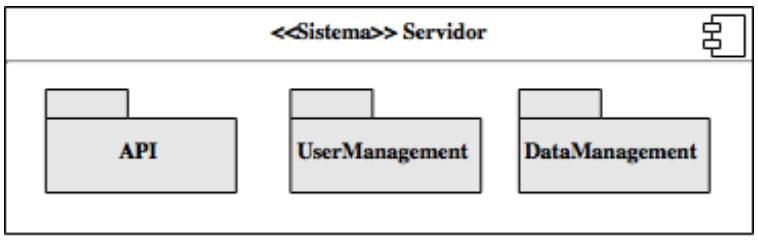

Figura 37 - Diagrama de componentes do servidor

Foi desenhado um modelo de domínio, apresentado na Figura 38, onde é representado o modelo que o servidor necessita de criar para armazenar os dados dos dispositivos. O sistema suporta que um utilizador possa ter várias centrais e cada uma dessas centrais ter um ou vários dispositivos associados. Cada um desses dispositivos pode ter sensores de monitorização energética, de movimento, de temperatura, ou outros, e todos eles recebem e enviam dados.

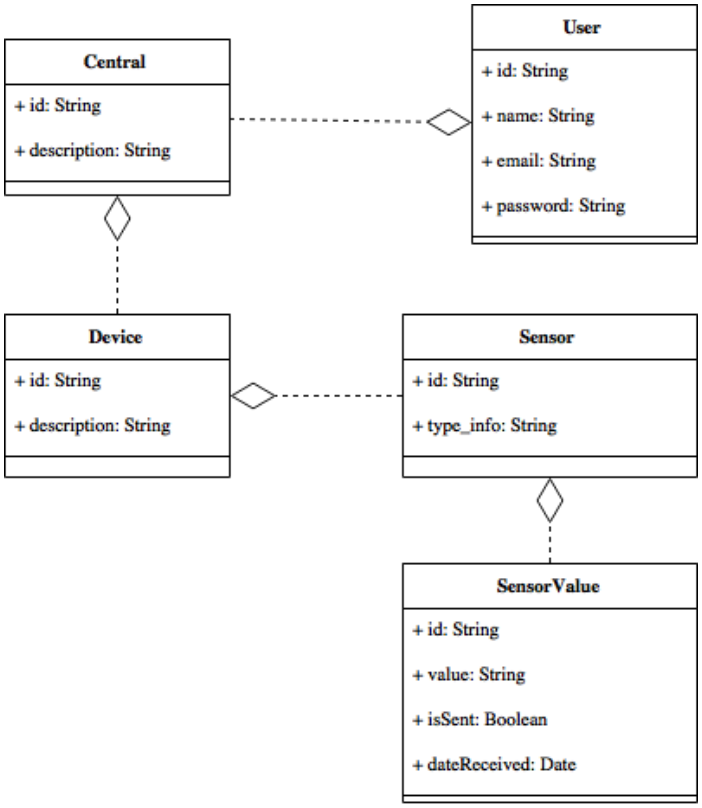

Figura 38 - Modelo de domínio do servidor segundo diagrama de classes (UML)

O servidor deve ter um API, por exemplo REST ou SOUP, que permita às centrais efetuarem pedidos HTTP para esses dados serem persistidos. O servidor deve também enviar valores, para essas centrais, valores que indiquem como gerir os equipamentos, esse envio pode ser feito com WebSockets ou a partir de pedidos REST utilizando AJAX em que a central fica continuamente à espera de resposta (*long polling*).

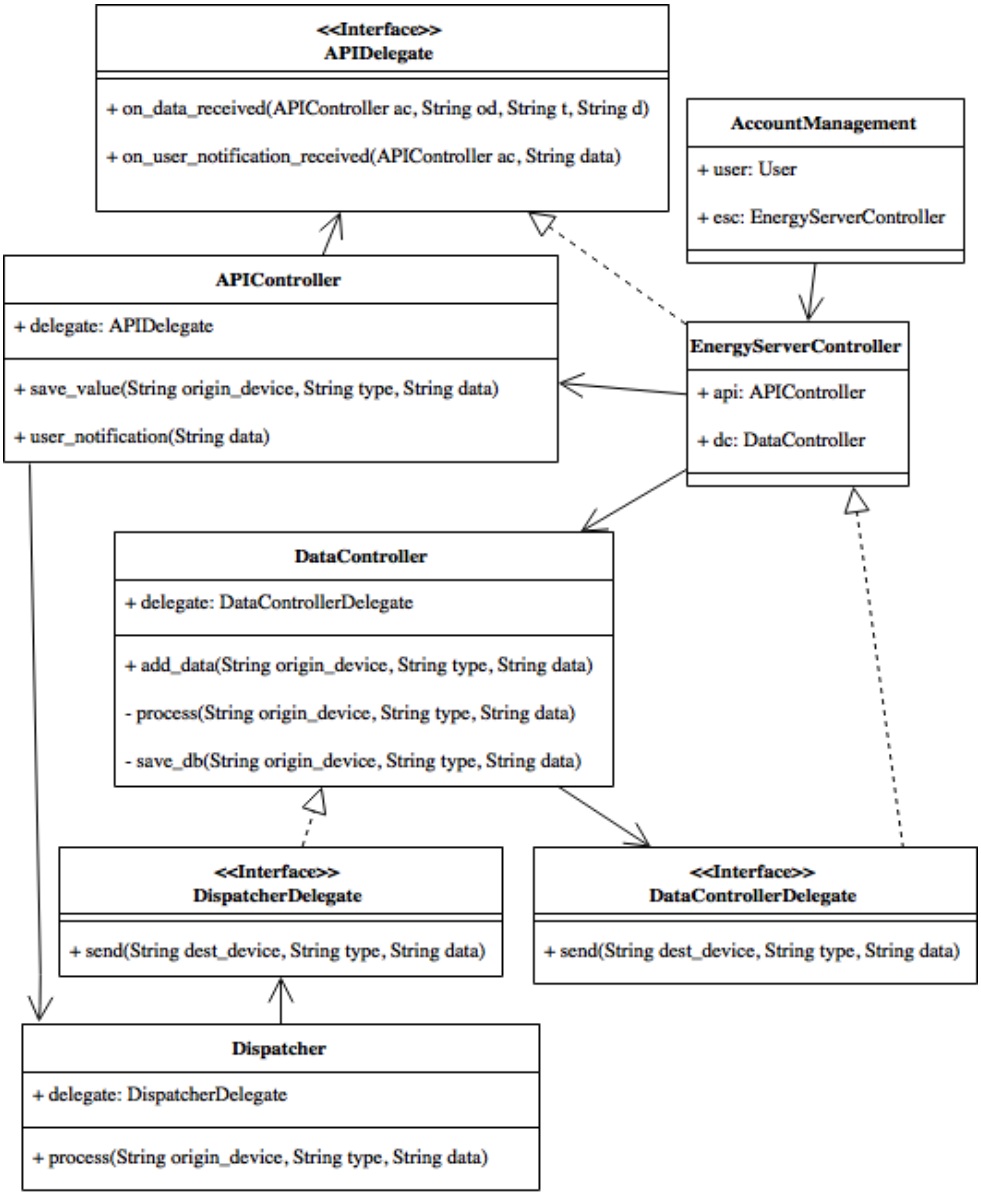

Figura 39 - Diagrama de classes do servidor

Existe um controlador principal que é o *EnergyServerController* que é responsável por gerir as restantes classes. A *APIController* gere os pedidos do dispositivo para o servidor e vice-versa. Quanto à classe *DataController* permite que sejam adicionados dados que podem ser guardados numa base de dados e enviados à classe *Dispatcher* para os processar com vista a encontrar ações que possam ser tomadas. Para a gestão de contas de utilizador existe a classe *AccountManagement*.

Nas secções seguintes são apresentadas em maior detalhe as funcionalidades que se pretende que sejam suportadas pelo servidor.

5.2.3.1 Persistir dados

Assim que recebe dados das centrais, o servidor deve persisti-los na base de dados de forma a conseguir gerar tabelas ou gráficos de utilização de equipamentos, ao longo do tempo. É

possível ver no diagrama de sequência da Figura 40 a forma como as diferentes classes do servidor interagem para persistir os dados.

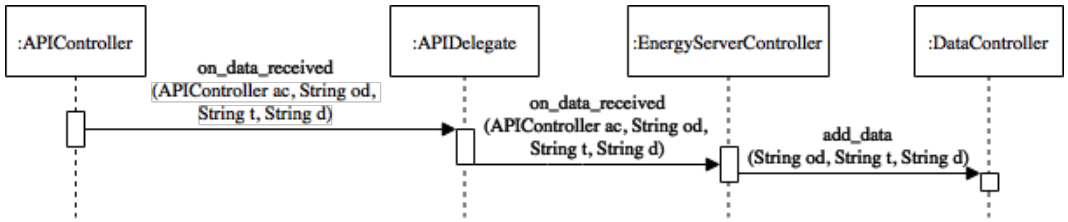

Figura 40 - Diagrama de sequência da persistência de dados

5.2.3.2 Detetar padrões

O servidor deve ser capaz de analisar os vários dados recebidos de forma a detetar padrões de utilização do espaço ou dos equipamentos, isto é, se um equipamento é ligado todos os dias somente entre 16h as 18h, o sistema é capaz de cortar o fornecimento de energia elétrica nas restantes horas de forma a poupar no consumo em *standby*. Na Figura 41 quando os dados são enviados para a *Dispatcher* estes são analisados através de um algoritmo que assim que deteta que deve tomar uma ação envia para a *DataController* os dados a enviar para um determinado dispositivo.

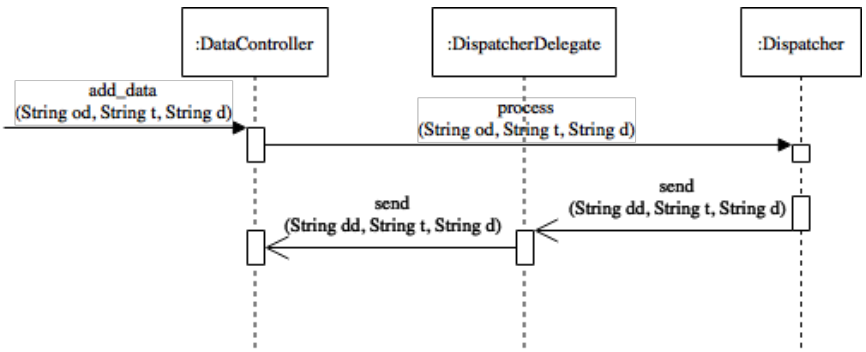

Figura 41 - Diagrama de sequência de deteção de padrões

5.3 Protocolo de Comunicação

A comunicação é um fator importante neste sistema visto que vários dispositivos interagem entre si. Entre o servidor e a central essa comunicação será feita através de HTTP. Quanto à comunicação entre central e dispositivo ou vice-versa, será através de mensagens em *broadcast* se a tecnologia permitir, isto porque não é necessário colocar o dispositivo que tem um baixo poder de processamento a fazer operações como conectar a um dispositivo específico. Caso isso não seja possível, o dispositivo deve criar um ponto de acesso onde a central poderá se conectar para receber e enviar dados.

Para a comunicação ser realizada entre os dispositivos e a central, é necessário um protocolo de comunicação. Este protocolo deve ser de tamanho reduzido para não exigir grande poder processamento, nem ultrapassar o máximo de bytes por pacote que alguma das tecnologias disponíveis suporta, de modo a não ser necessária a segmentação do mesmo.

Na Tabela 5 é possível ver o desenho desse protocolo que inclui 5 campos, os 4 primeiros com valor fixo de 1 *byte* e um último campo de tamanho variável.

- **Origem -** é o que identifica o dispositivo ou a central que enviou este pacote de dados.
- **Destino -** é o que identifica o dispositivo ou central que deve analisar estes dados.
- **Tipo -** representa o tipo de dados que vai ser enviado nesta mensagem.
- **Tamanho dos dados -** identifica o tamanho dos dados que vão aparecer no campo a seguir.
- **Dados -** são os dados enviados pelo dispositivo.

Bytes				
1	1	1	1	N
Origem	Destino	Tipo	Tamanho dos dados	Dados

Tabela 5 - Protocolo de comunicação

De realçar que os campos que só permitem 1 byte, têm disponíveis 256 combinações possíveis de valores, portanto o campo de dados como depende do valor introduzido no campo tamanho dos dados, pode ter um máximo de 256 bytes, sendo o tamanho máximo total do pacote de 260 bytes.

5.4 Sumário

Neste capitulo foram apresentados o desenho dos 3 módulos do sistema, o servidor, a central e os dispositivos.

Cada dispositivo deve ter um identificador único, isto para permitir que a central consiga agrupar os dados de acordo com os dispositivos que os enviam. Esse identificador não deve variar caso o dispositivo seja reiniciado, é então necessário que o mesmo seja incluído no *firmware*. Para o caso de ser preciso detetar se o dispositivo se encontra em *standby* é preciso aplicar técnicas de *machine learning* em que o sistema iria aprender quando o consumo é suficientemente baixo para o considerar em *standby*.

Após o desenho já é possível criar uma implementação para o sistema.

6 Implementação

Neste capítulo pretende-se explicar as escolhas de tecnologias feitas para o protótipo do sistema assim como mostrar a implementação.

6.1 Escolha de tecnologias

Para a implementação de um protótipo do sistema foi necessário proceder a análise de várias tecnologias existentes e referidas no capítulo 2, para escolher a mais adequada.

6.1.1 Comunicação

Existem várias possibilidades para os dispositivos comunicarem com a central, mas como o objetivo é existir uma maior liberdade para o utilizador colocar os dispositivos espalhados pelo espaço, optou-se por só abranger soluções de comunicação sem fios como Wi-Fi e Bluetooth, podendo ser utilizadas para ambientes industriais soluções com fio devido às interferências do meio e normas que seja preciso respeitar.

De entre as várias soluções Wi-Fi e Bluetooth analisadas todas elas apresentam velocidades suficientes para as necessidades do projeto, sendo que, só existirá necessidade de imitir pequenas quantidades de dados.

Analisando o parâmetro de alcance, Wi-Fi é a tecnologia com melhor desempenho, Wi-Fi Direct atinge cerca de 200 metros.

Quanto aos consumos energéticos de cada um, Bluetooth *Low Energy* destaca-se pelos seus baixos consumos energéticos, sendo os valores dependentes do restante *hardware* que é utilizado. Segundo os estudos referidos na secção 2.2.3, é o que apresenta os valores mais baixos das quatro tecnologias.

Com isto, foi selecionado a tecnologia Bluetooth *Low Energy* por ter a vantagem dos consumos energéticos baixos como descrito nos requisitos não funcionais apresentados na secção 4.3, sendo que o problema do alcance inferior poderia ser resolvido com o recurso a um protocolo de redes Mesh que fizesse o reenvio dos dados através de outros dispositivos que estivessem próximos ou com a utilização de mais centrais. À medida que o número de equipamentos assim como o espaço aumenta, poderá fazer sentido utilizar tecnologias com maior alcance como o caso do Wi-Fi.

6.1.2 Dispositivo

Para uma melhor gestão do consumo energético de um espaço, é preciso obter diversas informações com a finalidade de conseguir identificar e decidir onde é possível reduzir o consumo energético. Os dispositivos vão ter um papel importante para a análise dos dados e a tomadas de decisão que o servidor pode ter que efetuar. Vão estar espalhados por um determinado espaço e irão recolher vários dados e comunicar com a central. No caso dos dispositivos de monitorização energética podem receber dados da central com indicações para ligar ou desligar a passagem de corrente elétrica para o equipamento.

Para cumprir os requisitos não funcionais, apresentados na secção 4.3, os dispositivos devem permitir a conexão a sensores e o seu consumo energético ser reduzido. Visto a escolha da comunicação ter recaído sobre Bluetooth *Low Energy*, o dispositivo deve suportar essa tecnologia. Para isso foi utilizada a placa Waveshare BLE 400 que tem incorporada um chip Nordic 51822. Esta placa permite ainda que sejam acoplados vários sensores através de interfaces I2C, UART ou SPI. A escolha desta placa deveu-se essencialmente ao seu baixo custo.

A linguagem de programação escolhida para desenvolver nesta plataforma foi C, visto a documentação existente ser abundante, e existir um maior conhecimento pessoal em detrimento de outras.

Para interação com a interface BLE será utilizada a biblioteca arduino-BLEPeripheral, que simplifica o processo de anúncio e envio de dados na rede a partir desta tecnologia.

6.1.3 Sensores

Ligado ao dispositivo pela interface I2C, vai ser utilizado um sensor de monitorização de temperatura. O sensor é um TMP102 da Sparkfun.

6.1.4 Central

A central é o equipamento que vai fazer a transmissão dos dados entre os dispositivos e o servidor, para isso foi escolhido o Raspberry Pi 3 Model B. Esta decisão deveu-se ao seu baixo custo, aliado ao baixo consumo energético tendo disponível a capacidade de comunicar através de Bluetooth *Low Energy* com o dispositivo que foi a tecnologia de comunicação

escolhida. Pode estar ligada com ou sem fios a um router com ligação à internet para desta forma enviar os dados para um servidor.

O sistema operativo escolhido foi o Raspbian que é uma distribuição baseada em Linux e utilizada a API de BlueZ para a interação com o driver de Bluetooth *Low Energy*, assim como a biblioteca GDBus que permite uma simplificação no uso da API de BlueZ. O desenvolvimento de software será na linguagem C, devido a um maior conhecimento pessoal acerca da mesma.

Para interação com o servidor será utilizada a biblioteca curl[7] que pode ser utilizada na linguagem C e permite fazer pedidos HTTP de forma simples.

6.1.5 Servidor

Como servidor será utilizado o mesmo Raspberry Pi por uma questão de recursos e porque a quantidade de dados a serem recebidos não exigem um grande poder de processamento.

Será utilizada a Framework Django para o desenvolvimento e MySQL para a persistência de dados, ambas escolhidas também devido ao conhecimento pessoal. Para comunicação da central para o servidor foi escolhido o padrão REST, que como apresentado no estado da arte, utiliza JSON que é mais leve que o XML utilizado pelo SOAP e é relativamente mais simples a implementação.

6.2 Diagramas de domínio

Para esta implementação os diagramas são baseados nos mostrados no capítulo 5, mas concretos à tecnologia e aos dispositivos utilizados, isto é, usa as classes já apresentadas como abstratas e adiciona algumas de forma a cumprir os requisitos da tecnologia.

6.2.1 Dispositivo

O diagrama de classes desta implementação, representado na Figura 42, faz uso das mesmas classes mostradas no desenho. São estendidas algumas classes devido à forma como é feita a comunicação através da tecnologia Bluetooth *Low Energy*.

[7] https://github.com/curl/curl

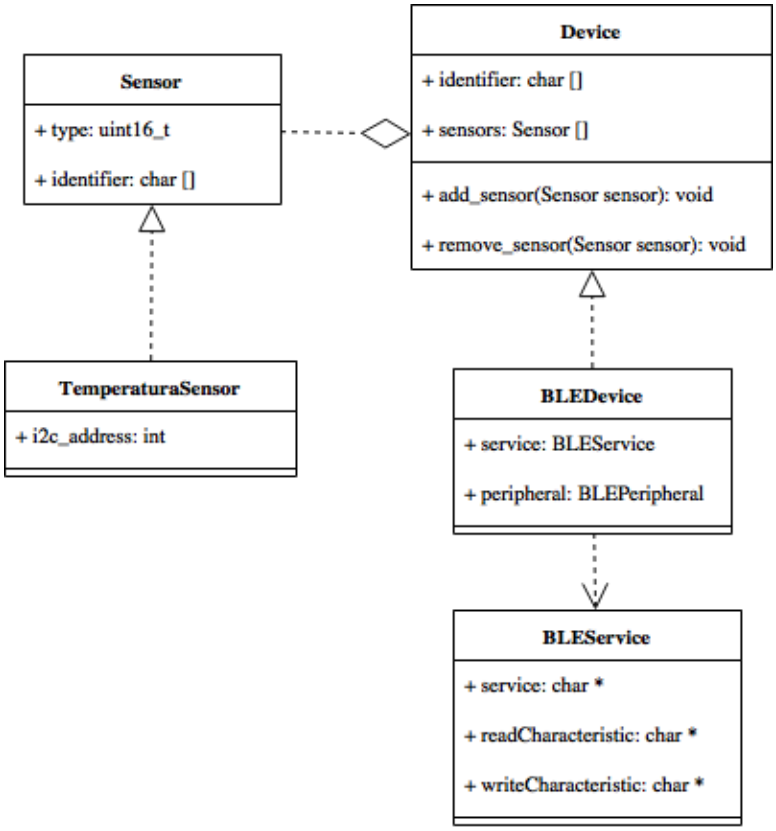

Figura 42 - Diagrama de classes de domínio (Dispositivo)

Um dispositivo que utilize BLE necessita de um serviço identificado no diagrama como BLEService. Este serviço é representado por um UUID anunciado pelo dispositivo, tendo esse serviço duas características, uma de leitura, outra de escrita que permitem enviar e receber dados respetivamente. Além disso é necessário também o uso de um BLEPeripheral, que é um objeto da biblioteca arduino-BLEPeripheral que permite iniciar o anúncio do dispositivo na rede e introduzir algumas configurações.

6.2.2 Central

Para a implementação desta central foram utilizadas as classes que foram apresentadas no capítulo 5, para o desenvolvimento do desenho. Essas classes são utilizadas como abstratas e foram acrescentadas outras especificas à implementação por Bluetooth *Low Energy*, que podem ser visualizadas na Figura 43.

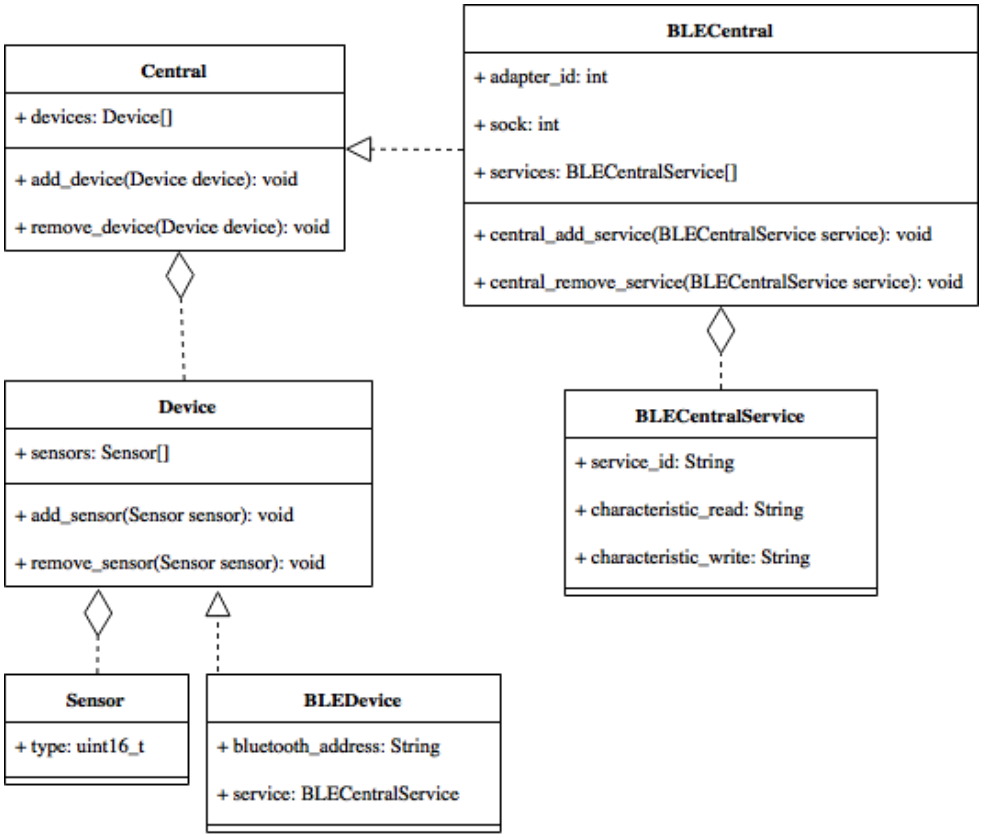

Figura 43 - Diagrama de classes de domínio (Central)

O dispositivo BLE tem um endereço Bluetooth associado assim como o serviço que está a utilizar para comunicar com a central. A central por sua vez tem um identificador do adaptador, que é utilizado para interações com esse adaptador, um identificador do *socket* aberto para receber pacotes e uma lista de serviços associados. Cada um desses serviços tem caraterísticas de leitura e de escrita.

6.2.3 Servidor

Em relação ao servidor, foi mantida a mesma estrutura de classes que as representadas na Figura 38, não existindo qualquer alteração. Essa estrutura é também a que vai ser persistida na base de dados *MySql*. Este fato deve-se ao servidor não precisar de uma implementação específica dependo do tipo de tecnologia de comunicação a ser utilizado.

6.3 Diagramas de classes

Os diagramas apresentados em seguida, tal como os de domínio, são baseados nos apresentados no capítulo de Desenho, isto é, utilizam as suas classes como abstratas e adicionam uma camada de classes especificas à implementação, neste caso de Bluetooth *Low Energy*.

6.3.1 Dispositivo

O diagrama de classes do dispositivo é semelhante ao abstrato não sofrendo muitas alterações a nível de representação. As alterações passam pelo uso do *BLEDevice* em detrimento da classe abstrata *Device*. Foram criadas classes especificas a BLE de *Connector*, *InputStream* e *OutputStream*. Na Figura 44 é visível um diagrama de classes que representa a implementação a ser feita no dispositivo.

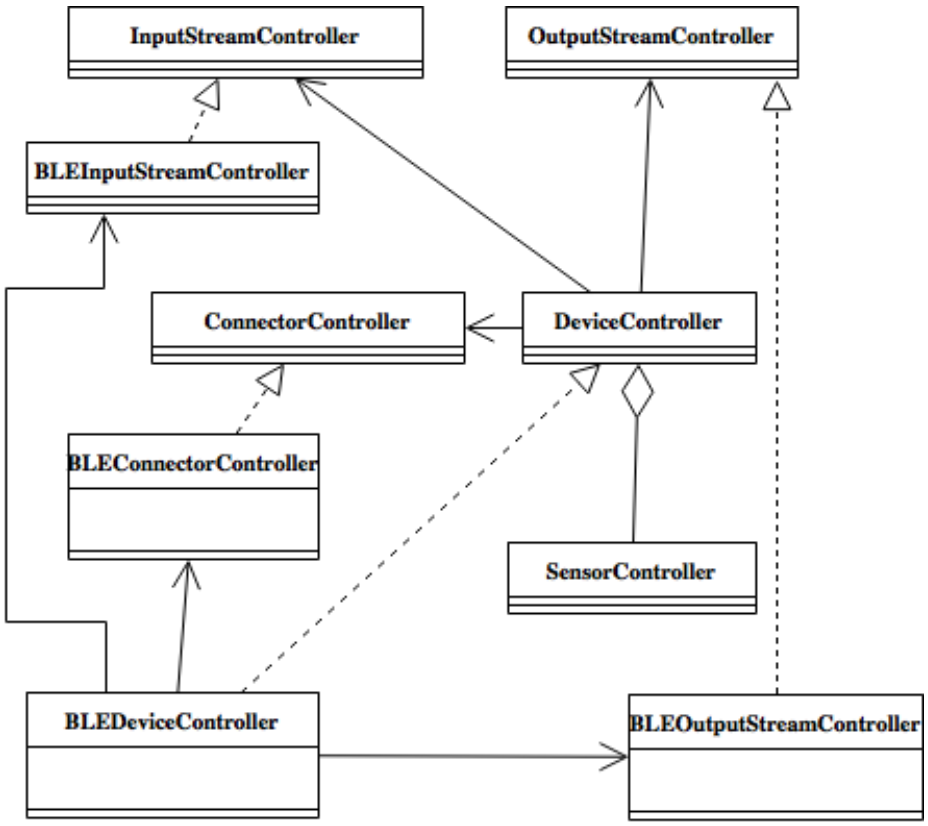

Figura 44 - Diagrama de classes do controlador (Dispositivo)

Para efeitos de simplificação do diagrama e de uma melhor compreensão foram retirados as variáveis, *interfaces* e métodos das classes abstratas já representadas no Desenho na Figura 26. Os métodos das classes abstratas são reescritos na implementação específica, por exemplo o connect() e o disconnect() do *Connector* são concretos a Bluetooth *Low Energy* tendo por isso que ser reescritos.

6.3.2 Central

O controlador da central também sofreu algumas alterações principalmente devido a forma como é feita a comunicação com dispositivos através de Bluetooth *Low Energy* em que são utilizadas características em vez dos mais habituais *sockets*. Essas alterações são visíveis na Figura 45.

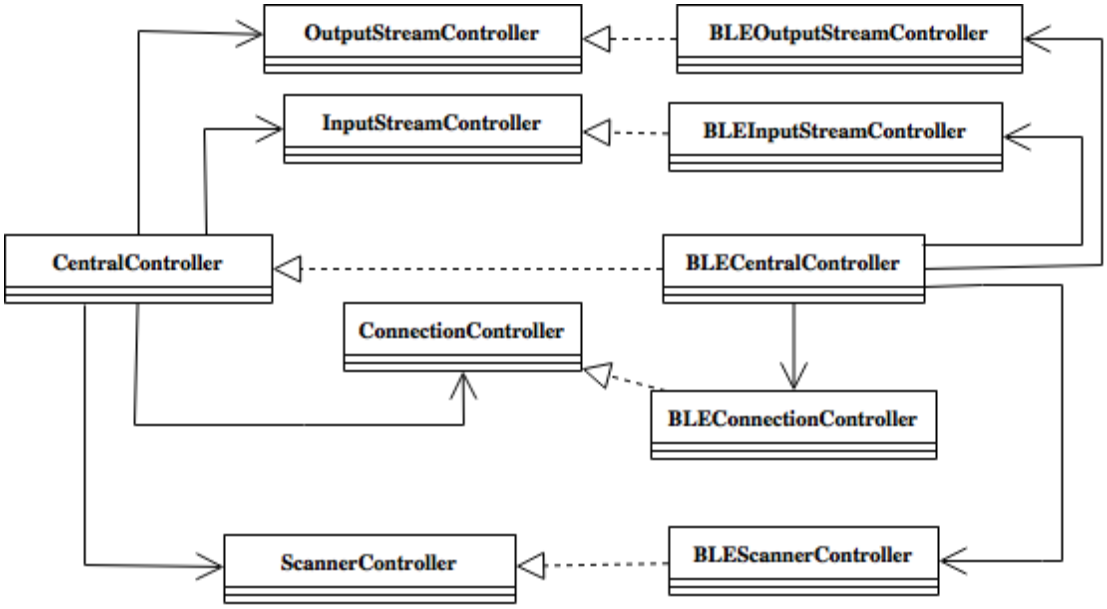

Figura 45 - Diagrama de classes do controlador (Central)

Tal como para o dispositivo, para efeitos de simplificação foram retirados os métodos, variáveis e *interfaces* já apresentados na Figura 33. Todos os métodos dos *controllers* são reescritos para a tecnologia em específico.

6.3.3 Servidor

O controlador do servidor tal como para as implementações referidas anteriormente, utilizou a implementação da Figura 39 como classes abstratas e foram acrescentadas algumas especificas a esta implementação com a utilização de REST. A representação do diagrama de classes é visível na Figura 46.

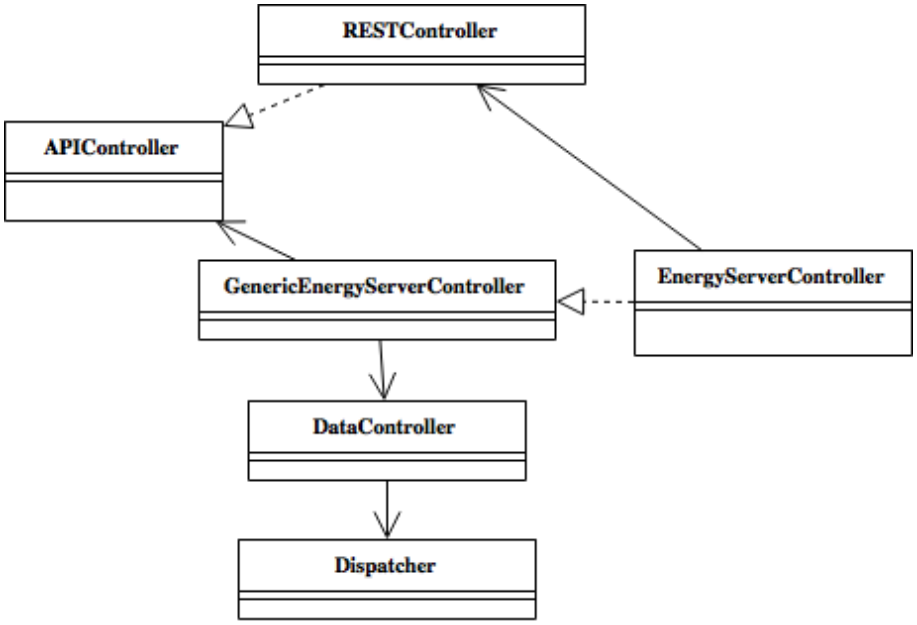

Figura 46 - Diagrama de classes do controlador (Servidor)

6.4 Configurações iniciais

Nesta secção são apresentadas as configurações que tem que ser adicionadas a cada plataforma para a implementação de cada caso de uso.

6.4.1 Dispositivo

Todos os dados recebidos e enviados pelo dispositivo, deve seguir o protocolo apresentado na Tabela 5.

6.4.2 Central

Todos os dados recebidos e enviados para o dispositivo, deve seguir o protocolo apresentado na Tabela 5. Os dados trocados com o servidor utilizam um protocolo em JSON apresentado na Listagem de código 1.

```
{
	"central_id": "central_id",
	"array_dados": [{
		"origem": "dispositivo_de_origem",
		"tipo": "tipo de dados",
		"dados": "valores"
	}]
}
```

Listagem de código 1 - Exemplo do protocolo JSON entre Central e Servidor

6.4.3 Servidor

O servidor foi implementado usando a Framework Django, tem alguns detalhes de configurações que serão descritos. A começar pelo modelo, o Django permite uma abstração sobre a criação da base de dados com base em query's SQL, sendo somente necessário criar um ficheiro com classes que representem esse modelo. O modelo para este projeto é apresentado na Listagem de código 2 e está de acordo com o apresentado na Figura 38.

```
from __future__ import unicode_literals

from django.db import models
from django.utils import timezone
from django.contrib.auth.models import User

class Central(models.Model):
          central_id = models.CharField(primary_key=True, max_length=200)
          user_id = models.ForeignKey(User, on_delete=models.CASCADE)
          description = models.CharField(max_length=200)

class Device(models.Model):
          device_id = models.CharField(primary_key=True, max_length=200)
          central_id = models.ForeignKey(Central, on_delete=models.CASCADE)
          description = models.CharField(max_length=200)

class Sensor(models.Model):
          sensor_id = models.AutoField(primary_key=True)
          device_id = models.ForeignKey(Device, on_delete=models.CASCADE)
          type_info = models.CharField(max_length=200)

class SensorValue(models.Model):
          sensor_value_id = models.AutoField(primary_key=True)
          sensor_id = models.ForeignKey(Sensor, on_delete=models.CASCADE)
          value = models.CharField(max_length=200)
          issent = models.BooleanField(default=False);
          date = models.DateTimeField(default=timezone.now)
```

Listagem de código 2 - Modelo da base de dados

Executando o comando apresentado na Listagem de código 3 as tabelas são geradas e incluídas no sistema de gestão de base de dados, neste caso o MySQL.

```
PC:~ user1$ python manage.py makemigrations blog
```

Listagem de código 3 - Comando para gerar a base de dados

6.5 Casos de uso

Nas secções seguintes são mostradas como foram feitas as implementações de cada caso de uso no sistema.

6.5.1 Registar Utilizador

O registo de utilizador (UC01) é feito no servidor, e para isso o utilizador deve abrir a página HTML, selecionar a opção registar e introduzir o nome, o email e uma password.

Numa primeira fase foi adicionado à Framework Django o caminho para abrir essa página de registo, isto é, ao introduzir, por exemplo, www.energysite.pt/register como endereço, deve chamar a função *register* incluída no ficheiro de views, como se pode ver na Listagem de código 4.

```
urlpatterns = [
         url(r'^register/', views.register),
]
```

Listagem de código 4 - Excerto para adicionar página ao Django

Em seguida é necessário adicionar ao ficheiro de views a função *register* que é despoletada quando o utilizador confirma os dados ou carrega a página. Esse excerto de código é visível na Listagem de código 5.

```
def register(request):
    if request.method == 'POST':
        form = UserRegistrationForm(request.POST)
        if form.is_valid():
            userObj = form.cleaned_data
            username = userObj['username']
            email =  userObj['email']
            password =  userObj['password']
            if not (User.objects.filter(username=username).exists() or
User.objects.filter(email=email).exists()):
                User.objects.create_user(username, email, password)
                user = authenticate(username = username, password = password)
                login(request, user)
                return HttpResponseRedirect('/')
            else:
                raise forms.ValidationError('Username with that email already
exists')
    else:
        form = UserRegistrationForm()
    return render(request, 'energysite/register.html', {'form' : form})
```

Listagem de código 5 - Excerto para criar utilizador

É possível ver o uso de modelo correspondente ao formulário que foi criado para ser recebido como um pedido POST, que é mostrado na Listagem de código 6.

```
class UserRegistrationForm(forms.Form):
    username = forms.CharField(
        required = True,
        label = 'Username',
        max_length = 32
    )
    email = forms.CharField(
        required = True,
        label = 'Email',
        max_length = 32,
    )
    password = forms.CharField(
        required = True,
        label = 'Password',
        max_length = 32,
        widget = forms.PasswordInput()
    )
```

Listagem de código 6 - Excerto do formulário de registo de utilizador

Por fim foi necessário criar uma página HTML com um formulário que suporte a introdução dos campos pedidos acima.

6.5.2 Efetuar Login

Para implementar a funcionalidade de efetuar login (UC02) o processo é semelhante ao referido em 6.5.1.

```
from django.contrib.auth import views as auth_views

urlpatterns = [
    url(r'^login/$', auth_views.login),
]
```

Listagem de código 7 - Excerto adicionar página de login

Foi utilizada uma view disponível no Django designada django.contrib.auth.views[8], que já faz a validação do nome do utilizador e password só sendo necessário adicionar aos *urls* disponíveis a função de login como se pode ver na Listagem de código 7.

6.5.3 Registar Central

Para registar a central (UC03) o utilizador deve previamente efetuar o login, em seguida encontrará uma opção para registar a central. Posteriormente terá que preencher um formulário em que deve indicar o identificador que se encontra na central assim como uma descrição para a mesma.

[8] https://docs.djangoproject.com/en/1.8/_modules/django/contrib/auth/views/

Como para os casos de uso anterior foi registado previamente o caminho "/registercentral" no ficheiro de urls.

A função que será invocada, no momento em que é feito um pedido a esse endereço é apresentada na Listagem de código 8.

```
def register_central(request):
    if request.method == 'POST':
        form = CentralRegistrationForm(request.POST)
        if form.is_valid():
            central_id = form.cleaned_data['centralid']
            description = form.cleaned_data['description']
            current_user_id = request.user.id
            if not (Central.objects.filter(central_id=central_id).exists()):
                central = Central(central_id, current_user_id, description)
                central.save()
                return HttpResponseRedirect('/')
            else:
                raise forms.ValidationError('A central já existe!')
    else:
        form = CentralRegistrationForm()
    return render(request, 'energysite/registercentral.html', {'form' : form})
```

Listagem de código 8 - Excerto para registar central

Caso seja um pedido por POST deve cumprir o modelo apresentado na Listagem de código 9. Caso não seja, será apresentada a página HTML com o formulário que deve ser preenchido.

```
class CentralRegistrationForm(forms.Form):
    centralid = forms.CharField(
        required = True,
        label = 'Identificador da central',
        max_length = 32
    )
    description = forms.CharField(
        required = False,
        label = 'Descricao',
        max_length = 200
    )
```

Listagem de código 9 - Excerto do formulário de registo da central

Essa central fica associada ao utilizador que esteja com o login efetuado.

6.5.4 Configurar Sistema

O caso de uso de configurar o sistema (UC04) é muito semelhante ao registar central mostrado na secção 6.5.3. Foi criada uma página nova e adicionada ao *urls* com o caminho "/configure", foi também concebida uma função *configure* que é invocada sempre que esse caminho for pedido. Como para formulários anteriores, é apresentado o formulário ou caso

seja um pedido de submissão do formulário são validados os campos assim como o login de utilizador. Se tudo estiver correto as configurações são persistidas na base de dados.

6.5.5 Ver estatísticas de utilização

Para ver as estatísticas de utilização (UC05), foi adicionado o caminho "/statistics" aos ficheiros de urls. Quando é feito um pedido a este caminho, existe uma função *statistics* no ficheiro de views que irá fazer uma *query* à base de dados por todos os dados recebidos pelas centrais registadas e é apresentado uma tabela com os dados.

6.5.6 Notificar o utilizador de gastos excessivos

O utilizador deve ser notificado sempre que existirem gastos excessivos (UC06). Quando são recebidos dados estes são analisados pelo servidor e caso exista uma variação anómala é enviado para o endereço de email com que o utilizador se registou um aviso. Para isso foram adicionadas as configurações apresentadas na Listagem de código 10 ao ficheiro de *settings*.

```
EMAIL_USE_TLS = True

EMAIL_HOST = 'smtp.gmail.com'

EMAIL_PORT = 587

EMAIL_HOST_USER = 'enderecodeemail@gmail.com'

EMAIL_HOST_PASSWORD = 'password'
```

Listagem de código 10 - Configurações de email

Sempre que é recebido dados e depois de efetuada a sua análise deve ser invocada a função *send* que utiliza o módulo django.core.mail[9] do Django e que permite o envio de uma mensagem de forma simples como mostrado na Listagem de código 11.

```
from django.core.email import send_email

send_email('Assunto', 'Mensagem', 'origem@gmail.com',

['destino@gmail.com'], fail_silently=False)
```

Listagem de código 11 - Envio de email

Para uma implementação definitiva poderia ser utilizado uma plataforma como o sendgrid[10] que seria o mais indicado para um sistema em grande escala, que fosse pretendido, por exemplo, efetuar campanhas de marketing por email.

[9] https://docs.djangoproject.com/en/1.11/topics/email/

6.5.7 Enviar instruções para a central

Por limitações de tempo, e porque este protótipo só tem um sensor de temperatura, este caso de uso não foi implementado, ou seja, a comunicação acontece só no sentido da central para o servidor.

6.5.8 Notificar o servidor de valores obtidos dos dispositivos

Para a central notificar o servidor dos valores que obteve dos dispositivos (UC08), o servidor vai utilizar o estilo arquitetural REST para permitir que sejam feitos pedidos POST e estes sejam persistidos.

Para isso o servidor deve criar um ponto de entrada para essa API, neste caso registando o endereço "/apicentral" no ficheiro de urls. Para esse ponto de entrada existe uma função apicentral que é invocado quando existe um pedido por esse endereço. Na Listagem de código 12 é visível um exemplo de uma função que poderia ser usada para tratar esse pedido. Caso a central não exista, é retornado erro, caso contrário é chamada a função *parsedata* que analisa os dados recebidos e persiste nos modelos correspondentes que foram apresentados na Listagem de código 2.

```
def apicentral(request):

    if request.method == 'POST':

        central_id = request.POST.get('central_id')
        data = request.POST.get('array_dados')
        if not (Central.objects.filter(central_id=central_id).exists()):
            raise HttpResponse('Central não registada')
        else:
            parsedata(data);
```

Listagem de código 12 - Exemplo de pedido POST

A central por sua vez deve fazer pedidos ao endereço disponibilizado pelo servidor enviando os valores que foram recebidos dos dispositivos. Para efetuar esses pedidos vai ser utilizada a biblioteca curl. Inicialmente é criada a *String* em JSON que vai ser enviada no pedido POST, assim como são definidos vários parâmetros na biblioteca curl para em seguida ser feito o pedido. Na Listagem de código 13 é mostrado um exemplo de código de um pedido POST.

[10] https://sendgrid.com

```
static const char *post_msg= "{\"central_id\": \"central_id\",\"array_dados\":
[{\"origem\": \"dispositivo_de_origem\",\"tipo\": \"tipo de dados\",\"dados\":
\"valores\"}]}"

handle = curl_easy_init();

if(handle) {
    curl_easy_setopt(handle, CURLOPT_URL, "http://www.energysite/apicentral");
    curl_easy_setopt(handle, CURLOPT_POSTFIELDS, post_msg);
    /* ..................... */
}

CURLcode res = curl_easy_perform(handle);
if(res != CURLE_OK)
{
    //ERROR
}
curl_easy_cleanup(handle);
```

Listagem de código 13 - Exemplo de pedido curl na linguagem C

6.5.9 Enviar para a central os valores obtidos dos sensores

Depois do dispositivo iniciar a monitorização dos sensores deve emitir os valores recolhidos pela rede (UC09). Essa emissão para o caso de Bluetooth *Low Energy* será através da criação de um serviço a que serão adicionadas duas características:

- A característica com propriedades de leitura onde são colocados os dados que se pretende que sejam enviados. A central subscreve essa característica e recebe uma notificação sempre que existirem alterações.
- A característica com propriedades de escrita que é utilizada quando a central pretende enviar dados para o dispositivo.

Na Listagem de código 14 é possível ver um extrato de código onde, utilizando a biblioteca arduino-BLEPeripheral, é inicializado o anúncio de dados na rede.

```
BLEPeripheral blePeripheral= BLEPeripheral();
BLEService sensorService = BLEService("6E400001-B5A3-F393-E0A9-E50E24DCCA9E");
BLECharacteristic sensorCharacteristicTX = BLECharacteristic("6E400002-B5A3-
F393-E0A9-E50E24DCCA9E", BLERead | BLEWrite, 20);
BLECharacteristic sensorCharacteristicRX = BLECharacteristic("6E400003-B5A3-
F393-E0A9-E50E24DCCA9E", BLERead | BLENotify, 20);
BLEDescriptor sensorDescriptor = BLEDescriptor("2901", "value");

void setup_ble()
{
    blePeripheral.setLocalName("temp_sensor");
    blePeripheral.setDeviceName("temp_sensor");
    blePeripheral.setAdvertisedServiceUuid(sensorService.uuid());

    blePeripheral.addAttribute(sensorService);
    blePeripheral.addAttribute(sensorCharacteristicTX);
    blePeripheral.addAttribute(sensorCharacteristicRX);
    blePeripheral.addAttribute(sensorDescriptor);

    sensorCharacteristicTX.setValue("");

    blePeripheral.begin();
}
```

Listagem de código 14 - Configurações de serviços e características

Para o envio dos dados, e visto a central ter que subscrever a característica de leitura que o dispositivo disponibiliza, só é necessário alterar o valor que se encontra na característica através de uma função que é apresentada na Listagem de código 15.

```
void sendTemp(char * value)
{
    blePeripheral.poll();
    sensorCharacteristicRX.setValue(value);
}
```

Listagem de código 15 - Alterar dados das características

Quanto à central, terá que numa primeira fase procurar por dispositivos que se encontrem na proximidade e quando encontrar, verificar se é um dispositivo com o mesmo UUID usado nesta implementação e em caso afirmativo subscrever a característica de leitura para ser notificado sempre que existir uma alteração.

Para iniciar a pesquisa é necessário definir os parâmetros de pesquisa do HCI (Host Controller Interface) tais como, identificador do *socket* aberto, tipo de pesquisa, intervalo entre pesquisas entre outros, visíveis na Listagem de código 16.

```
uint8_t scan_type = 0x00;
uint16_t interval = htobs(0x0010);
uint16_t window = htobs(0x0010);
uint8_t own_type = 0x00;
uint8_t filter_policy = 0x00;

if(hci_le_set_scan_parameters(sock, scan_type, interval, window, own_type,
filter_policy, 1000) < 0)
{
    //ERROR
}
```

Listagem de código 16 - Definir parâmetros da pesquisa

Depois de definidos os parâmetros é necessário habilitar a pesquisa através de HCI, onde são definidos parâmetros como, identificador do *socket*, se deve ser iniciada ou parada a pesquisa, se devem ou não ser filtrados os resultados da pesquisa duplicados e por fim um *timeout* para a pesquisa. Essa definição de parâmetros é apresentada na Listagem de código 17.

```
if (hci_le_set_scan_enable(sock, 1 /* 1 - ligado, 0 - desligado */, 0 /* 0-
filtro desabilitado, 1-filtrar duplicados */, 1000  /* timeout */) < 0)
{
       //ERROR
}
pthread_t thread1;
int i1 = pthread_create(&thread1, NULL, &central_scanner_scanning,
(void*)scanner);
if(i1)
{
    //ERROR
}
else
{
    //Started
}
```

Listagem de código 17 - Habilitar HCI

Para concluir a pesquisa foi criada uma função, que é executada numa *thread* em separado, que é iniciada, e sempre que são encontrados novos dispositivos, são processados os dados anunciados pelos mesmo. Para isso é extraído o UUID do serviço anunciado e comparado com o UUID que se pretende encontrar, em caso de sucesso é notificado o *delegate* através da função *on_device_found*. Na Listagem de código 18 é possível ver um excerto da função de pesquisa.

```
static void * central_scanner_scanning(void * scr)
{
    BLEScannerController * scanner = (BLEScannerController *) scr;
    unsigned char buf[HCI_MAX_FRAME_SIZE];
    while(1)
    {
        memset(buf, 0, sizeof(buf));
        int buf_len = recv(scanner->sock, buf, sizeof(buf), 0);
        if (-1 == buf_len)
        {
            /* ......................... */
            //ERROR
            break;
        }

        /* ......................... */

        evt_le_meta_event *meta = (void *)(buf + (1 + HCI_EVENT_HDR_SIZE));
        le_advertising_info *info = (le_advertising_info *) (meta->data + 1);

        /* ......................... */

        process_info(scanner, info);
    }

    pthread_exit(NULL);
}
```

Listagem de código 18 - Exemplo de função de pesquisa

Por fim, depois de ser encontrado um dispositivo a característica de leitura deve ser subscrita, o que significa a abertura da *stream* de *input*. Para esta parte da implementação foi utilizada a biblioteca GDBus que simplifica as chamadas à API de BlueZ, na Listagem de código 19 é possível ver um exemplo dessa subscrição, em que sempre que existir uma alteração da característica por parte do dispositivo a função *OnCharacteristicPropertiesChanged* é invocada.

```
static void subscribe(InputStreamController * input_controller, GDBusProxy
*proxy)
{
    GError *err = NULL;

    g_signal_connect(proxy,
                     "g-properties-changed",
                     G_CALLBACK(OnCharacteristicPropertiesChanged),
                     input_controller);

    GVariant * ret = g_dbus_proxy_call_sync(proxy,
                                            "StartNotify", NULL,
                                            G_DBUS_CALL_FLAGS_NONE, -1, NULL,
&err);

    if (ret == NULL)
    {
        //ERROR
    }
    //Stream Opened
```

Listagem de código 19 - Subscrever uma característica

6.5.10 Envio de instruções para o dispositivo

A central deve conseguir enviar dados para o dispositivo (UC10). O conceito de envio de dados por Bluetooth *Low Energy* da central para o dispositivo é semelhante ao apresentado no 6.5.9, só que para este caso de uso, a central deve escrever na característica do dispositivo que tem propriedades de escrita, que é possível ver na Listagem de código 14. Esta escrita de dados foi implementada também usando a biblioteca GDBus e na Listagem de código 20 é apresentado um extrato desse código para escrever na característica.

```
static void write(OutputStreamController * output_controller, char * data, int
length, GDBusProxy * wc)
{
    GError *err = NULL;
    GVariant * message = g_variant_new_fixed_array(G_VARIANT_TYPE_BYTE, data,
length, sizeof(char));
    GVariant * ret = g_dbus_proxy_call_sync(wc, "WriteValue",
                            g_variant_new_tuple(&message, 1),
G_DBUS_CALL_FLAGS_NONE, -1, NULL, &err);

    if (ret == NULL)
    {
        //ERROR
    }
}
```

Listagem de código 20 - Exemplo da escrita na característica do dispositivo

O dispositivo, depois de cada *sleep*, ao iniciar um novo ciclo de execução, deve verificar se novos dados foram colocados na característica.

6.6 Sumário

Neste capítulo mostrou-se a implementação concreta do sistema, fazendo uso do desenho descrito anteriormente. Após esta implementação é importante identificar se o sistema é eficiente e cumpre os objetivos para que foi desenhado.

7 Avaliação

Neste capítulo pretende-se avaliar o projeto realizado e as perguntas de investigação que se pretendem testar e o seu resultado.

7.1 Avaliação do projeto

Para avaliar este projeto numa primeira fase foi pensado utilizar grandezas como:

- *Precison* e *recall*, para medir as falhas que o sistema pode ter, por exemplo, desligar a televisão quando o utilizador está a ver uma emissão na mesma. Com estas medidas é possível calcular os falsos positivos e qual a percentagem de vezes que o sistema funciona corretamente.
- A percentagem de satisfação dos utilizadores perante a poupança que obtiveram. Para calcular essa satisfação vão ser feitos inquéritos ao utilizador e verificar calculando a percentagem, qual o grau de satisfação.
- O tempo para verificar se uma bateria perdeu uma maior ou menor capacidade em relação a um dispositivo que não é monitorizado pelo sistema. Para fazer essa verificação serão comparados valores do número de ciclos de carga e autonomia máxima.
- O consumo energético para verificar se existiu efetivamente um decréscimo do consumo. Para isso será utilizado um teste A/B, em que este teste tem como propósito que um dos grupos A ou o B seja sujeito a um teste, o outro é considerado o grupo de controlo. Para este caso o grupo B vai ser sujeito à utilização do sistema nas mesmas condições e com o mesmo equipamento que foi usado no teste A. O resultado destes dois grupos vai ser comparado usando o T-Test, para verificar se efetivamente existiu um decréscimo do consumo energético.

Pelo facto de o protótipo implementado não incluir equipamentos que permitem a monitorização energética, não foi possível avaliar os falsos positivos ou a percentagem de utilizadores satisfeitos, sendo que este protótipo só inclui a monitorização da temperatura.

A avaliação foi feita com o auxílio de um simulador onde é possível testar diversas configurações para diversos espaços e verificar se é sempre eficiente ou em que casos não o é, isto é, verificar se existem situações que o sistema é ineficiente, por exemplo, no caso em que realiza a monitorização de uma televisão sem a utilização de qualquer sensor de movimento. Com isto, são obtidos os valores do consumo energético com e sem o sistema, sendo feita posteriormente a comparação de resultados.

7.2 Simulador

Para além da implementação do sistema de gestão energética mostrado anteriormente, para validar se esse mesmo sistema era eficiente foi necessário desenvolver um simulador. Este simulador foi implementado na linguagem Java e através da introdução de variáveis que indiquem como se comporta um equipamento no espaço, identifica se o sistema vai ser viável nessas condições. As variáveis que devem ser introduzidas no simulador são:

- As horas de utilização do equipamento, que no caso de ser um equipamento do sistema, deve ser o total de horas que se pretende analisar.
- A potência usada em standby.
- A potência usada enquanto se encontra e utilização.

Essa introdução é mostrada na Listagem de código 21, onde se pode ver a criação de dois dispositivos, um do sistema com o nome Raspberry e que se encontra ligado as 24h no dia que se pretende simular, e outro um equipamento que se encontra a ser monitorizado, neste caso uma TV de nome LG, com um total de 5h em funcionamento, um consumo em funcionamento de aproximadamente 43Wh e em *standby* de 0,23Wh.

```
public static Device getRaspberry()
{
       ArrayList<Time> upTimes = new ArrayList<Time>();
       upTimes.add(createTime("15-09-2017 00:00:00", "16-09-2017 00:00:00"));

       Device rasp = new Device("Raspberry", upTimes, 1.282, 0,
       Device.Type.SystemDevice);
       return rasp;
}

public static Device getTVLG651V()
{
    ArrayList<Time> upTimes = new ArrayList<Time>();
    upTimes.add(createTime("15-09-2017 12:00:00", "15-09-2017 13:00:00"));
    upTimes.add(createTime("15-09-2017 16:00:00", "15-09-2017 20:00:00"));

    Device tv = new Device("LG", upTimes, 43.177, 0.238, Device.Type.Device);
    return tv;
}
```

Listagem de código 21 - Introdução de equipamentos no simulador

O simulador recebe essa lista de dispositivos e calcula os custos energéticos dos equipamentos do sistema e os custos dos equipamentos que se encontram no espaço. Para

uma simulação de um funcionamento normal deve ser calculado o custo em execução, por exemplo, para o caso apresentado anteriormente, a ser executado durante 24h, seria para a TV:

$$(43{,}177 \text{ x } 5) + (0{,}238 \text{ x } 19)$$

Isto é, durante 5h a TV está em funcionamento e outras 19h está em *standby* e isso dá o total de potência consumida, sem o auxílio do sistema. Com a utilização do sistema para o mesmo caso o cálculo seria:

$$(43{,}177 \text{ x } 5) + (1{,}282 \text{ x } 24)$$

A TV é utilizada durante 5h, o custo da mesma em *standby* é descartado, pois o sistema detetará que se encontra em *standby* e cortará o funcionamento de energia. A esse funcionamento da TV é adicionado o custo do sistema durante as 24h.

Na Listagem de código 22 é visível a forma como é feita o cálculo do custo.

```
for(int i = 0; i < getDevices().size(); i++)
{
    Device d = getDevices().get(i);
    long standbyUpTime = calculateStandbyUpTime(d.getTotalUpTime());
    long upTime = d.getTotalUpTime();
    double consumption = d.getTotalConsumption();
    double standbyConsumption = d.getStandbyConsumption() *
Utils.convertMsToHour(standbyUpTime);

    if(d.getType() == Device.Type.Device){
        simulatorDeviceConsumption += consumption;
        simulatorDeviceStandbyConsumption += standbyConsumption;
    }
    else
    {
        simulatorSystemConsumption += consumption;
        simulatorSystemStandbyConsumption += standbyConsumption;
    }
    //.........
}
```

Listagem de código 22 - Cálculo dos custos

O simulador quando termina, notifica através de um *delegate* os resultados obtidos na simulação.

7.3 Medições

Foram recolhidos dados de vários equipamentos com o intuito de, posteriormente através do simulador apresentado na secção 7.2, mostrar os resultados e indicar se o sistema é eficiente ou em que casos o é.

Para esta recolha de resultados foi utilizado o equipamento de monitorização energética, o UNI-T UT230B-EU que foi apresentado no estado da arte e que nos permite efetuar a medição

de vários parâmetros da eletricidade, como por exemplo, a intensidade da corrente (I), diferença de potencial (D.D.P.), frequência (Freq.), fator de potência (F.P.) e a potência (P.).

Com apenas monitor energético ligado à tomada, sem nenhum equipamento ligado, obtivemos os valores apresentado na Tabela 6.

	I (A)	**D.P. (V)**	**FREQ. (HZ)**	**F.P.**	**P. MEDIDA (W)**	**P. ATIVA (W)**
UNI-T UT230B-EU	0,006	233	50	0	0	1,398

Tabela 6 - Consumos do equipamento de monitorização

A P. Medida (Potência medida), é a potência que o equipamento de monitorização nos indica estar a consumir e a P. Ativa (Potência ativa) foi calculada com recurso à formula da potência ativa descrita na secção 2.7.

Na Tabela 6 é possível ver que o monitor de corrente apesar de não se encontrar ligado a nenhum equipamento tem uma intensidade de corrente de 0,006 Amperes, que foi retirada a todas as medições feitas posteriormente. A potência medida pelo equipamento e a potência ativa diferem, portanto, assume-se que a potência ativa tem em conta o valor de corrente gasto pelo equipamento de monitorização. Em futuras medições será desprezado o valor de potência medido pelo dispositivo, visto não ter precisão abaixo de 0,1W, e só será mostrado o valor de potência ativa. A diferença de potencial também sofreu variações entre e durante as diversas medições, sendo o valor mais comum cerca de 233 Volts e como tal, foi considerado o valor de referência para todas as medições.

Numa primeira fase foram feitas medições aos equipamentos que foram propostos, ou que compõem o protótipo e os resultados dessas medições são apresentados na Tabela 7. De notar que como dispositivo foi utilizado o Waveshare BLE400, mas poderia ter sido utilizado um Arduino em conjunto com um módulo Wi-Fi, ou o Wemos D1, enquanto que o Raspberry foi o equipamento escolhido para a funcionalidade de central do sistema.

	I (A)	D.D.P. (V)	FREQ. (HZ)	F.P.	P. ATIVA (WATTS)	OBSERVAÇÕES
WAVESHARE BLE400	0,001	233	50	0,12	0,028	Sensor de temperatura acoplado, a executor protótipo
ARDUINO	0,001	233	50	0,27	0,063	Inativo
ARDUINO + ESP8266	0,004	233	50	0,39	0,363	Inativo, só com o módulo ESP8266 acoplado e a funcionar como Access-Point
WEMOS D1	0,002	233	50	0,32	0,149	A funcionar como Access-Point
WEMOS D1	0,003	233	50	0,32	0,224	Com um equipamento ligado a executar PING
RASPBERRY	0,01	233	50	0,51	1,188	Inativo
RASPBERRY	0,009	233	50	0,55	1,153	Inativo e HDMI desligado
RASPBERRY	0,01	233	50	0,55	1,282	A executar protótipo e HDMI desligado

Tabela 7 - Monitorização de equipamentos do sistema

Nessa tabela é possível verificar que o dispositivo utilizado no protótipo para monitorizar o espaço, consome um total de energia de 0,028W aplicando a fórmula da potência ativa. Por outro lado, o Raspberry a executar o protótipo implementado consome um total de 1,282W.

É possível então concluir, que o sistema no seu todo consome cerca de 1,31W com um Raspberry e com um dispositivo a monitorizar a temperatura, estando ambos a executarem o protótipo implementado. Por cada dispositivo acrescentado ao sistema a potência consumida vai aumentar em 0,028W. Através das medições efetuadas não foi possível verificar variações na potência consumida pelo Raspberry mesmo estando ligado a dois dispositivos.

Considerando que a monitorização energética vai ter um custo muito semelhante ao da monitorização da temperatura, visto que só varia o sensor acoplado ao dispositivo, o sistema será eficiente sempre que monitorizar um equipamento que consuma pelo menos 1,31W em *standby*, isto se, nesse caso o sistema cortar o fornecimento de energia elétrica.

Para o caso de se considerar que o sistema deve detetar quando o utilizador pretende ligar o equipamento, por exemplo através de um sensor de movimentos, o valor do *standby* poderá não ser completamente nulo, isto porque, poderá existir falsos positivos, por exemplo, um utilizador entrar no espaço e não utilizar o equipamento monitorizado, vai fazer com que o sistema ative o fornecimento de corrente elétrica, e exista um consumo em *standby* do mesmo. Mas o tipo de reação do sistema, irá sempre depender de um cenário mais concreto e do tipo de sensores ou algoritmos utilizados.

Alguns equipamentos também foram monitorizados para posteriormente proceder à simulação. Esses resultados da monitorização são apresentados na Tabela 8.

	I (A)	D.D.P. (V)	FREQ. (HZ)	F.P.	P. ATIVA (WATTS)	OBSERVAÇÕES
LG UJ634V	0,213	233	50	0,87	43,177	Ligada
LG UJ634V	0,051	233	50	0,02	0,238	Em standby
BOX CISCO ISB2231	0,07	233	50	0,64	10,438	Em funcionamento
BOX CISCO ISB2231	0,052	233	50	0,61	7,391	Em standby
BOX + TV + ROUTER + TELEFONE	0,464	233	50	0,88	95,139	Todos os equipamentos em funcionamento
BOX + TV + ROUTER + TELEFONE	0,147	233	50	0,49	16,783	Box e Tv em standby, todos os outros em funcionamento.

Tabela 8 - Monitorização do espaço 01

De notar que todas as medições foram obtidas depois de os equipamentos se encontrarem em *standby* há pelo menos 30 minutos, isto porque alguns, como o caso das Box de televisão por vezes ficam a consumir mais energia durante uns minutos de modo a permitir uma inicialização rápida. O telefone é da marca Philips, modelo M330 enquanto que o Router é da marca Huawei, modelo HG8247Q.

É visível que alguns equipamentos têm um consumo em standby considerável como o caso da box. Se considerarmos um sistema com uma televisão, box, router e telefone os valores em standby são ainda mais altos.

Outras medições foram efetuadas em equipamentos de diferentes tipos e apresentados na Tabela 9.

	I (A)	D.D.P. (V)	FREQ. (HZ)	F.P.	P. ATIVA (WATTS)	OBSERVAÇÕES
MICROONDAS BLUESKY 900W	0,016	233	50	0,65	2,423	Em standby
MAQ. CAFÉ DELONGHI PIXIE CLIPS EN 126	0,017	233	50	0,16	0,634	Em standby
DESPERTADOR SONY ICF-DS15IPN	0,002	233	50	0,16	0,075	-
IPHONE 5S	0,039	233	50	0,66	5,997	A carregar
IPHONE 5S	0,001	233	50	0,06	0,014	Carregado

Tabela 9 - Monitorização do espaço 02

É possível observador que um micro-ondas desligado, tendo somente um pequeno monitor que apresenta as horas, consome cerca de 2,4W, praticamente o dobro do sistema. Quanto à maquina de café e o despertador os consumos são baixos, mas pode compensar ter um sistema a monitorizar o espaço, dependendo dos equipamentos que se incluem no sistema para além destes. O carregamento de um telemóvel exige cerca de 6W de potência, e o valor consumido depois de carregado é de cerca de 0,01W.

Com isto, é possível concluir que a potência total do equipamento em *standby* (PTES) é a multiplicação da potência do equipamento em standby (PES) pelas horas em *standby* (HS):

$$PTES = (PES * HS)$$

A potência total do sistema (PTS) é a multiplicação da potência do sistema (PS) pelas horas de utilização (HU):

$$PTS = (PS * HU)$$

Portanto, é possível estimar que, apesar das considerações que foram feitas ao sistema, o mesmo é viável quando o somatório da potência total dos equipamentos em *standby* do número de equipamentos (NE) for superior à potência total do sistema:

$$(\sum_{k=1}^{NE} PTES_k) > (PTS)$$

7.4 Simulação

Considerando os valores obtidos na secção 7.3 foram feitas algumas simulações. De notar que para estas simulações foi considerado o tempo total de 24 horas, em que caso um equipamento não esteja a ser utilizado está a consumir potência em *standby*.

A potência total consumida pelo equipamento sem a utilização do protótipo, representa a potência dos equipamentos consumida em *standby* a multiplicar pelo número de horas em *standby*, a que se soma a potência total do equipamento em utilização (PTEU) a multiplicar pelas horas de utilização:

$$PTE = (PTES * HS) + (PTEU * HU)$$

Em que, para esta simulação, o resultado da soma de HS com HU deve ser igual a 24 horas.

$$HS + HU = 24$$

Para o gasto de potência total com o sistema com o equipamento (PTSE) foi considerado que, os gastos de potência do equipamento enquanto se encontram em utilização, mais o gasto de potência do sistema durante 24 horas ignorando os gastos em *standby* dos equipamentos:

$$PTSE = (PTEU * HU) + (PTS * 24)$$

Com isto pretende-se simular um sistema que através da utilização deste sistema anula-se os custos dos *standby's*. Para tal foram criados 4 cenários:

- **Cenário 1 –** Utilização de uma TV LG, modelo UJ634V, durante 4 horas estando as restantes 20 horas em *standby*. O sistema é composto por um Raspberry que representa a central, e por um WaveShare BLE400 que é o dispositivo de monitorização.
- **Cenário 2 –** Utilização de uma box de televisão, modelo Cisco ISB2232, durante 2 horas sendo as restantes 22 horas em *standby*. O sistema é composto por um Raspberry que representa a central, e por um WaveShare BLE400 que é o dispositivo de monitorização.
- **Cenário 3 –** Utilização de uma box de televisão, modelo Cisco ISB2232, durante 6 horas sendo as restantes 18 horas em *standby*. O sistema é composto por um Raspberry que representa a central, e por um WaveShare BLE400 que é o dispositivo de monitorização.
- **Cenário 4 –** Monitorização de um conjunto de equipamento através de uma tomada com 4 entradas, com uma utilização conjunta de 3 horas sendo as restantes 21 horas em standby. O sistema é composto por um Raspberry que representa a central, e por um WaveShare BLE400 que é o dispositivo que ficará ligado à tomada de 4 entradas de forma a monitorizar os 4 equipamentos em simultâneo.

Na Tabela 10 é possível verificar os resultados desses cenários, onde é apresentado o número do cenário (Cen.), o equipamento monitorizado (Equip.), o número de horas de utilização, os dispositivos que compõem o sistema, o gasto total sem sistema e o gasto total com a utilização do sistema, assim como a diferença dos gastos utilizando o sistema.

CEN.	EQUIP.	UTILIZAÇÃO (HORAS)	SISTEMA	GASTO TOTAL S/SISTEMA (W)	GASTO TOTAL C/SISTEMA (W)	DIFERENÇA DE GASTOS (%)
1	TV	4	Central+ Dispositivo	177,468	204,148	15,03%
2	Box	2	Central+ Dispositivo	183,478	52,316	-71,49%
3	Box	6	Central+ Dispositivo	195,666	94,068	-51,92%
4	Box+TV+ Telefone+ Router	3	Central+ Dispositivo	637,86	317,857	-50,17%

Tabela 10 - Simulação 01

Com estas simulações é possível concluir que em alguns casos é possível poupar cerca de 50% da potência consumida. No primeiro caso isso não acontece, pois, o consumo da televisão em *standby's* é inferior ao do gasto de todo o sistema de monitorização. É também de notar que à medida que o número de horas de utilização aumenta a eficiência do sistema diminui, isto porque os ganhos do sistema acontecem só quando o equipamento está em standby.

Apesar do limitado alcance da tecnologia BLE, em determinadas habitações seria possível monitorizar vários equipamentos em diferentes divisões o que nos permite misturar as simulações com outro tipo de equipamentos:

- **Cenário 5 –** Foi considerado que a box, TV, telefone e router estiveram em utilização durante 3 horas sendo o restante tempo em *standby*, o micro-ondas foi utilizado durante 10 minutos com uma potência de 150W e o Iphone foi carregado durante 2 horas permanecendo as restantes 22 horas ligado à corrente elétrica.

O resultado deste cenário é apresentado na Tabela 11.

CEN.	EQUIP.	SISTEMA	GASTO TOTAL S/SISTEMA (W)	GASTO TOTAL C/SISTEMA (W)	DIFERENÇA DE GASTOS (%)
5	Box + TV + Telefone + Router + Micro-ondas + Iphone 5S	Central + Dispositivo (x3)	733,077	355,195	-51,55%

Tabela 11 - Simulação 02

Com esta simulação é possível mostrar que apesar do aumento do número de dispositivos de monitorização, como o seu custo em termos de potência é bastante reduzido, continua a compensar a sua utilização tendo uma poupança na ordem dos 51%.

Outros sensores podem ser utilizados permitindo desligar todos os equipamentos monitorizados, caso não existam movimentos no espaço. Outro dos exemplos seria utilizar um sensor de temperatura, e ligar ou desligar um aquecedor sempre que os valores de temperatura variarem entre valores aceitáveis para o utilizador, o que permitiria manter o espaço numa temperatura estável, não existindo quebras de temperatura e obrigando a um menor esforço energético.

7.5 Sumário

Neste capítulo foi apresentado um simulador, assim como medições a equipamentos para permitir que posteriormente fossem feitas simulações com os mesmos. Depois do capítulo de avaliação concluído, é importante tirar conclusões em relação ao sistema.

8 Conclusão

Este capítulo pretende apresentar conclusões ao projeto realizado, identificar dificuldades, apresentar o trabalho futuro e efetuar uma apreciação global.

Na secção 1.2 foram levantadas algumas questões de investigação que são importantes responder depois de ter sido realizada a avaliação do sistema. A partir dos resultados da simulação apresentados na secção 7.4, é possível responder a essas perguntas:

- O sistema permite na maioria dos cenários apresentados reduzir o consumo energético, dependendo sempre do tipo de equipamento a ser monitorizado.
- Nem todos os equipamentos controlados pelo sistema reduzem o seu consumo energético. Para os cenários apresentados se o consumo em standby for inferior ao do sistema ou se só tiverem curtos períodos em standby, pode não existir uma diminuição do consumo.
- Se não forem utilizados os sensores adequados, o sistema pode interferir com a habitual utilização dos equipamentos. Se utilizarmos somente sensores de monitorização energética os equipamentos são desligados, mas o sistema não conseguirá identificar quando os deve voltar a ligar, exigindo um processo manual, como por exemplo, inserir as horas de utilização do equipamento ou aceder ao sistema para indicar que é necessário voltar a fornecer corrente elétrica.
- O aumento de vida útil das baterias não foi simulado, contudo, se o sistema considerar que um equipamento completamente carregado se encontra em *standby*, visto a potência consumida reduzir significativamente como apresentando na Tabela 9, o sistema irá cortar o fornecimento de energia. Com base do descrito no estado da arte na secção 2.6, em que é afirmado que um dos problemas que afeta as baterias é quando ficam muito tempo a serem carregadas, o sistema provavelmente irá melhorar a vida útil das baterias, não se podendo identificar um valor ou o número de ciclos que carga que serão poupados comparativamente a uma utilização normal.

Para os cenários que o sistema pode não ser eficiente, é de realçar que o utilizador continua a usufruir de outras vantagens como monitorizar o espaço em tempo real, ou perceber que equipamentos consomem mais energia.

Graças ao desenho que foi desenvolvido para este projeto, para adicionar novas funcionalidades é somente necessário a integração de novos módulos, já estando implementadas as funcionalidades de trocas de dados que permitem a monitorização.

Este projeto, como foi mostrado, pode ter benefícios em vários setores e tem ainda a possibilidade de serem acrescentadas novas funcionalidades para além da eficiência energética o que permitirá chegar a novos mercados um pouco por todo o mundo.

8.1 Objetivos atingidos

No geral, os objetivos para este projeto foram concluídos com sucesso. Foi possível criar um protótipo do sistema capaz de através de uma rede de sensores monitorizar um espaço, e apresentar ao utilizador os dados recolhidos numa página HTML em tempo real.

Apesar de no protótipo apresentado não ter sido implementado a monitorização energética, por simulação, através de um algoritmo que toma decisões em relação ao fornecimento de energia elétrica, dado o consumo do equipamento num determinado instante, foi possível comprovar que com este sistema e através da implementação de um módulo de monitorização energética, é possível o sistema tomar decisões que de fato trazem vantagens em termos energéticos.

8.2 Dificuldades

Devido à dimensão do projeto surgiram algumas dificuldades, tanto a nível de *software*, como de *hardware*.

Em termos de dispositivos físicos (*hardware*), a comunicação entre o dispositivo e a central deve ser adaptada caso a caso. Utilizar Bluetooth *Low Energy*, num espaço onde não seja possível ter ligação por cabo, pode ser uma tecnologia bastante eficiente em termos de consumo energético, mas a velocidade de transferência de dados é relativamente baixa comparativamente a outras tecnologias sem fios. Em alguns espaços, devido ao ruído pode não ser possível a sua utilização, tendo a comunicação que ser feita através de redes cabladas.

Em termos de desenvolvimento de produto (*software*), todo o desenho teve que ser abstrato à tecnologia de comunicação, pois como referido na secção 6.1.1, o hardware necessário a utilizar no espaço pode variar de caso para caso. Foi por isso criado um desenho que permite abstrair essas tecnologias. A forma de implementação varia de tecnologia para tecnologia, onde por exemplo com Wi-Fi comunicamos através de *streams*, e em Bluetooth *Low Energy* a

comunicação é feita através de um conceito diferente, as características. Isso exigiu uma adaptação para a tecnologia a ser utilizada.

A API de BlueZ que foi utilizada na central está muito pouco documentada, e acaba por ser bastante complexo entender o funcionamento de algumas funções em determinados casos.

Foi necessário criar um protocolo de comunicação de forma a enviar os dados entre os dispositivos e a central que permitisse diminuir ao máximo o número de *bytes* ocupados.

8.3 Limitações

O sistema pode apresentar alguns tipos de limitações na monitorização energética, dependendo sempre do tipo de sensores que são utilizados, dado que, é necessário detetar em alguns casos quando o utilizador deseja utilizar os equipamentos, o que pode ser feito através de sensores ou de padrões de utilização. Para o tipo de espaços onde não existe um padrão de utilização dos equipamentos, as vantagens a nível de eficiência energética podem ser limitadas.

8.4 Trabalho futuro

No final deste projeto foi possível identificar algumas formas de o melhorar e criar novas funcionalidades que pudessem aumentar a eficiência do sistema assim como uma maior automatização de todo o processo de gestão.

Integrar este projeto no mercado industrial pode ser útil, existindo muitas empresas industriais que poderiam beneficiar de uma redução substancial nos gastos energéticos.

As futuras cidades inteligentes, podem ter uma grande vantagem em incluir um sistema capaz de gerir um conjunto de sensores, ligando e desligando a iluminação em determinados locais somente quando não há movimentos, ou outras aplicações como analisar o tráfego para determinar a sinalização a ser colocada em painéis eletrónicos.

A base deste projeto poderá ser utilizada para a criação de casas inteligentes em que os utilizadores as consigam monitorizar à distância, aumentando a segurança e o conforto dos proprietários. Opções como ligar e desligar o aquecimento, levantar ou baixar os estores assim como identificar intrusos, são algumas das opções que podem ser acrescentadas a um sistema deste tipo, permitindo aumentar o mercado em que se inclui.

Para uma simplificação da utilização deste software poderá ser desenvolvido no futuro uma funcionalidade em que é possível reconhecer o tipo de dispositivo que se encontra a consumir eletricidade. Os dispositivos que se encontram a monitorizar os consumos dos equipamentos podem conseguir detetar as variações dos consumos e aprender com elas através de técnicas de *machine learning* o que iria permitir que o equipamento ligado à corrente elétrica fosse

automaticamente e corretamente identificado, permitindo ao sistema tomar decisões mais corretas.

Outra funcionalidade interessante de ser implementada é o sistema ter a capacidade de produzir relatórios de utilização e avisar o utilizador de uma variação anómala dos consumos.

A interação direta de alguns equipamentos que utilizem baterias com os dispositivos, poderia permitir aumentar a vida útil das mesmas, desligando o fornecimento de corrente quando estas se encontrassem carregadas, impedindo também sobreaquecimentos como referido na secção 2.6.

Através da simulação foi possível detetar que em certos cenários, como por exemplo, o cenário 01 apresentado na Tabela 10, pode compensar a utilização de dispositivos que comuniquem diretamente com o servidor, visto que o maior gasto do sistema é com a central.

A integração do sistema com uma aplicação móvel seria uma funcionalidade interessante, onde seriam enviadas notificações para o telemóvel com informações sobre o espaço. Também seria útil o próprio telemóvel interagir diretamente com os dispositivos para conhecer os valores de temperatura ambiente, humidade, ou consumos do equipamento que se encontra a monitorizar.

A monitorização do espaço está dependente da tecnologia de comunicação escolhida, ou seja, pode não ser possível monitorizar um edifício inteiro devido às limitações de alcance das tecnologias. Criar uma rede de topologia *mesh* entre todos os dispositivos do sistema, aumenta o alcance e a dimensão da rede, permitindo a gestão de espaços amplos sem necessidade de incluir várias centrais para a recolha de dados.

A utilização de várias tecnologias de comunicação dentro do mesmo sistema, pode ser importante pela redundância que acrescenta a rede, permitindo a um dispositivo utilizar outra tecnologia caso outra falhe. Além disso, dependendo do espaço em que se pretende instalar o sistema, algumas tecnologias poderão ser mais eficientes que outras, devido a interferências e outros fatores, ou seja, a utilização de várias tecnologias permite uma maior adaptabilidade ao espaço em que vai ser instalado o sistema.

O protocolo criado para trocar mensagens entre os vários dispositivos que compõem o sistema pode ser alterado para acrescentar suporte para controlo de versões, em que 4 a 8 *bits* seria o ideal e assim seria possível manter retro compatibilidade entre versões diferentes do protocolo.

A segurança das comunicações também deverá ser feita utilizando encriptação de forma a não permitir a utilizadores mal-intencionados modificar as leituras ou até mesmo alterar o comportamento do sistema.

Seria interessante se no futuro os equipamentos tivessem uma API em que fosse possível obter estes valores, como o consumo energético, deixando de ser necessário o uso de dispositivos de monitorização.

9 Referências

[1] F. Pacheco, “Energias Renováveis : Breves Conceitos,” *Conjunt. e Planej.*, no. 149, pp. 4–11, 2006.

[2] Department for Environment Food & Rural Affairs UK, “Public attitudes and behaviours towards the environment - tracker survey A research report completed for the Department for Environment , Food and,” no. September, 2009.

[3] Union of Concerned Scientists, “Confronting the Realities.” [Online]. Available: http://www.ucsusa.org/global_warming#.WH6iM7GtFo4.

[4] American Petroleum Institute, “No Title.” [Online]. Available: http://www.api.org/oil-and-natural-gas/consumer-information/in-the-classroom/energy-resources/nonrenewable-energy-resources. [Accessed: 01-Jan-2016].

[5] U.S. Energy Information Administration, “What is U.S. electricity generation by energy source?,” 2016. [Online]. Available: https://www.eia.gov/tools/faqs/faq.php?id=427&t=3.

[6] U.S. Energy Information Administration, “How is electricity used in U.S. homes?” [Online]. Available: https://www.eia.gov/tools/faqs/faq.php?id=96&t=3.

[7] IEA, “Energy Performance Certification of Buildings - A Policy Tool to Improve Energy Efficiency,” p. 64, 2010.

[8] Adene, “O que é.” [Online]. Available: http://www.adene.pt/sce/o-que-e-1. [Accessed: 20-Jul-2010].

[9] Solar Energy Industries Association, “Solar Industry Data,” 2016. [Online]. Available: http://www.seia.org/research-resources/solar-industry-data.

[10] California Public Utilities Commission, “About the California ZNE Residential Action Planning Effort.”

[11] K. Tweed, “California Wants All New Homes to Be Net Zero in 2020,” 2015. [Online].

Available: https://www.greentechmedia.com/articles/read/California-Wants-All-New-Homes-to-be-Net-Zero-in-2020.

[12] D. A. S. C. Europeias, “18.12.2008,” pp. 45–52, 2008.

[13] J. Morgan, “A Simple Explanation Of ‘The Internet Of Things,’” 2014.

[14] I. Bluetooth SIG, “Bluetooth 5,” 2016. [Online]. Available: https://www.bluetooth.com/specifications/bluetooth-core-specification/bluetooth5.

[15] Laird Technologies, “Bluetooth Smart Ready vs . Bluetooth Smart devices Roles in Bluetooth Low Energy,” vol. d, no. August, pp. 1–2, 2012.

[16] B. Sig, “Bluetooth SIG 2014 Annual Report,” 2014.

[17] Bluetooth SIG Inc., “Attribute Protocol.” [Online]. Available: https://www.bluetooth.com/specifications/bluetooth-core-specification.

[18] M. Palmer, *Hands-On Networking Fundamentals*, 2nd ed. 2013.

[19] B. Low and E. Link, “Modeling the Maximum Throughput of,” vol. 15, no. 11, pp. 1187–1189, 2011.

[20] NordicSemiconductor, “No Title,” 2016. [Online]. Available: http://blog.nordicsemi.com/getconnected/things-you-should-know-about-bluetooth-range.

[21] J. Donovan, “Bluetooth Goes Ultra-Low-Power,” 01-Dec-2011.

[22] Apple, “No Title,” 2013. [Online]. Available: https://developer.apple.com/library/content/documentation/NetworkingInternetWeb/Conceptual/CoreBluetooth_concepts/CoreBluetoothOverview/CoreBluetoothOverview.html.

[23] Atlantic Wireless Telecommunications, “CONNECTING DEVICES ANYTIME, ANYWHER.” [Online]. Available: http://atlanticwireless.net/wi-fi/.

[24] Cisco, “802.11ac: The Fifth Generation of Wi-Fi Technical White Paper,” 2015. [Online]. Available: http://www.cisco.com/c/en/us/products/collateral/wireless/aironet-3600-series/white_paper_c11-713103.html.

[25] Silex technology america Inc., “Network Environment Examples.” [Online]. Available: http://www.iet.com.hk/English/support/jiten3.html.

[26] R. Pagoto, “No Title,” 2015. [Online]. Available: http://www.decom.ufop.br/imobilis/wi-fi-direct-tutorial-parte-1/.

[27] Wi-Fi Alliance, “How fast is Wi-Fi Direct?” [Online]. Available: http://www.wi-fi.org/knowledge-center/faq/how-fast-is-wi-fi-direct.

[28] Actiontec, “An Industry Look at Wi-Fi Direct Wi-Fi Direct : The Details.”

[29] R. Friedman and A. Kogan, “On Power and Throughput Tradeoffs of WiFi and Bluetooth in Smartphones,” pp. 900–908, 2011.

[30] P. Smith, “Comparing Low-Power Wireless Technologies,” 2011. [Online]. Available: https://www.digikey.com/en/articles/techzone/2011/aug/comparing-low-power-wireless-technologies.

[31] LinkLabs, “What Is Mesh Topology,” 2015. [Online]. Available: https://www.link-labs.com/blog/what-is-mesh-topology.

[32] Raspberry PI Foundation, “Raspberry PI 3 Model B.” [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.

[33] Arduino, “Introduction.” [Online]. Available: https://www.arduino.cc/en/Guide/Introduction.

[34] Arduino, “Interfacing with Hardware.” [Online]. Available: https://playground.arduino.cc/Main/InterfacingWithHardware. [Accessed: 20-Jul-2009].

[35] Espressif Systems, “ESP8266.” [Online]. Available: http://espressif.com/en/products/hardware/esp8266ex/overview. [Accessed: 20-Jul-2010].

[36] WEMOS Electronics, “D1.” [Online]. Available: https://wiki.wemos.cc/products:d1:d1. [Accessed: 20-Jul-2010].

[37] Waveshare, “BLE400.” [Online]. Available: https://www.waveshare.com/wiki/BLE400. [Accessed: 20-Jul-2010].

[38] D. Nedelkovski, “How PIR Sensor Works and How To Use It with Arduino,” 2015. [Online]. Available: http://howtomechatronics.com/tutorials/arduino/how-pir-sensor-works-and-how-to-use-it-with-arduino.

[39] Raspberry Pi Foundation, “Learning Resources.” [Online]. Available: https://www.raspberrypi.org/learning/parent-detector/worksheet/.

[40] Safewise, “The Beginner’s Guide to Motion Sensors.” [Online]. Available: http://www.safewise.com/resources/motion-sensor-guide.

[41] SPARKFUN, “SPARKFUN DIGITAL TEMPERATURE SENSOR BREAKOUT - TMP102.” [Online]. Available: https://www.sparkfun.com/products/13314. [Accessed: 20-Jul-2010].

[42] The Energy Detective, “TED PRO HOME.” [Online]. Available: http://www.theenergydetective.com/tedprohome.html.

[43] EDP, “EDP Re:Dy.” [Online]. Available: https://energia.edp.pt/particulares/servicos/redy/. [Accessed: 20-Jul-2010].

[44] TP-Link, “HS110.” [Online]. Available: http://www.tp-link.com/us/products/details/cat-5516_HS110.html.

[45] Belkin, “Wemo® Insight Smart Plug.” [Online]. Available: http://www.belkin.com/us/p/P-F7C029/.

[46] UNI-T, “UT230B Series Power Sockets.” [Online]. Available: http://www.uni-

trend.com/productsdetail_1939_1174_1174.html. [Accessed: 20-Jul-2010].

[47] Microsoft, “Windows 10 IoT Core.” [Online]. Available: https://developer.microsoft.com/pt-pt/windows/iot/Explore/IoTCore.

[48] Canonical Ltd, “Core.” [Online]. Available: https://developer.ubuntu.com/core.

[49] Canonical Ltd, “Get started.” [Online]. Available: https://developer.ubuntu.com/core/get-started.

[50] Raspbian, “Welcome to Raspbian.” [Online]. Available: https://www.raspbian.org/.

[51] c4learn.com, “C language history.” [Online]. Available: http://www.c4learn.com/c-programming/c-language-history.

[52] c4learn.com, “C++ History.” [Online]. Available: http://www.c4learn.com/cplusplus/cpp-history/.

[53] Oracle, “The History of Java Technology.” [Online]. Available: http://www.oracle.com/technetwork/java/javase/overview/javahistory-index-198355.html.

[54] Python Software Foundation, “History of the software,” 2001. [Online]. Available: https://docs.python.org/2.0/ref/node92.html.

[55] E. Dietrich, “C# Version History: Examining the Language Past and Present.” [Online]. Available: https://blog.ndepend.com/c-versions-look-language-history/. [Accessed: 20-Jul-2010].

[56] Microsoft, “Deploying an App.” [Online]. Available: https://developer.microsoft.com/en-us/windows/iot/Docs/appdeployment.

[57] BlueZ Project, “About.” [Online]. Available: http://www.bluez.org. [Accessed: 20-Jul-2009].

[58] Havoc Pennington, David Wheeler, John Palmieri, and Colin Walters, “D-Bus Tutorial.” [Online]. Available: https://dbus.freedesktop.org/doc/dbus-tutorial.html. [Accessed: 20-Jul-2010].

[59] S. Mistry, “arduino-BLEPeripheral.” [Online]. Available: https://github.com/sandeepmistry/arduino-BLEPeripheral. [Accessed: 20-Jul-2009].

[60] Arduino, “Wire.” [Online]. Available: https://www.arduino.cc/en/Reference/Wire. [Accessed: 20-Jul-2009].

[61] Django Software Foundation, “Meet Django.” [Online]. Available: https://www.djangoproject.com. [Accessed: 20-Jul-2009].

[62] Node.js Foundation, “Node.JS.” [Online]. Available: https://nodejs.org/en/. [Accessed: 20-Jul-2010].

[63] Stackify, “SOAP vs. REST: The Differences and Benefits Between the Two Widely-Used Web Service Communication Protocols.” [Online]. Available: https://stackify.com/soap-vs-rest/. [Accessed: 20-Jul-2009].

[64] D. Sheiko, “WebSockets vs Server-Sent Events vs Long-polling.” [Online]. Available: http://dsheiko.com/weblog/websockets-vs-sse-vs-long-polling. [Accessed: 20-Jul-2009].

[65] Battery University, “What’s the Best Battery?” [Online]. Available: http://batteryuniversity.com/learn/archive/whats_the_best_battery.

[66] Battery University, “How to Prolong Lithium-based Batteries,” 2017. [Online]. Available: http://batteryuniversity.com/learn/article/how_to_prolong_lithium_based_batteries.

[67] The Engineering ToolBox, “Active, Apparent and Reactive Power.” [Online]. Available: https://www.engineeringtoolbox.com/kva-reactive-d_886.html. [Accessed: 20-Jul-2010].

[68] Discovergy, “Discovergy.” [Online]. Available: https://discovergy.com.

[69] EMS3, “Smart Meters.” [Online]. Available: http://ems3.com/smart-meters/.

[70] ESightEnergy, “Cost Savings & Energy Usage Analysis.” [Online]. Available: http://www.esightenergy.com/esight-platform-2/cost-savings-energy-usage-analysis/.

[71] P. Koen, H. M. J. Bertels, and E. J. Kleinschmidt, “Managing the Front End of Innovation — Part I,” *Res. Technol. Manag.*, vol. 57, no. 3, pp. 34–43, 2014.

[72] C. M. Vigna, “Fuzzy Front End: Linha de Frente da Inovação,” 2015.

[73] P. Koen *et al.*, “Providing Clarity and a Common Language To the ‘Fuzzy Front End.,’” *Res. Technol. Manag.*, vol. 44, no. 2, pp. 46–55, 2001.

[74] A. K. Agrawal and Z. Rahman, *Roles and Resource Contributions of Customers in Value Co-creation*, vol. 3, no. 1. Holy Spirit University of Kaslik, 2015.

[75] R. Sanchez-Fernandez and M. A. Iniesta-Bonillo, “The concept of perceived value: a systematic review of the research,” *Mark. Theory*, vol. 7, no. 4, pp. 427–451, 2007.

[76] EDP, “Abordagem à Sustentabilidade.” [Online]. Available: http://www.edp.pt/pt/sustentabilidade/abordagemasustentabilidade/Pages/default_new.aspx.

[77] BusinessDictionary, “Stakeholder.” [Online]. Available: http://www.businessdictionary.com/definition/stakeholder.html. [Accessed: 01-Jan-2017].

[78] Scott W. Ambler, “Use Case Diagramming Guidelines.” [Online]. Available: http://www.agilemodeling.com/style/useCaseDiagram.htm#Actors. [Accessed: 15-Sep-2017].