Instituto Politécnico de Tomar

Escola Superior de Tecnologia de Tomar

Tiago Alexandre Martins Fernandes

SISTEMA INTEGRADO DE SEGURANÇA E GESTÃO DE ENERGIA (SISGE)

Dissertação de Mestrado

Orientado por:

Prof. Doutor Mário Gomes, ESTT – IPT

Prof. Doutor Paulo Coelho, ESTT - IPT

Dissertação apresentada ao Instituto Politécnico de Tomar
para cumprimento dos requisitos necessários
à obtenção do grau de Mestre
em Engenharia Eletrotécnica

vita.ipt
Vida Assistida
por Ambientes Inteligentes

RESUMO

Apresenta-se nesta dissertação o projeto de um sistema autónomo de gestão de segurança e energia.

O estado social atual implica que passemos grande parte das nossas vidas no interior de edifícios. Sempre que possível, preferimos que as nossas tarefas de simples e rotineira execução sejam efetuadas por sistemas autónomos. Juntando estes dois pontos à necessidade de gerir os recursos energéticos e a segurança obtemos a necessidade de sistemas de automação para edifícios, que agreguem monitorização de segurança e energia.

Devido à vasta oferta tecnológica de baixo custo e de boa qualidade disponível no mercado, o desenvolvimento deste tipo de sistema de monitorização torna-se cada vez menos complicado. As tecnologias chave destes sistemas são na maior parte das vezes de fonte de utilização livre. Estes sistemas tornam-se assim em alternativas aos sistemas existentes no mercado.

Nesta dissertação é descrito o estado da arte e apresentado o projeto de um sistema de baixo custo para gestão da segurança e energia em edifícios comercias e não comerciais.

Foi desenvolvida com o propósito de monitorizar consumos energéticos em edifícios, salvaguardar bens e pessoas e fornecer ao utilizador um sistema que guarda toda a informação e permite a análise da mesma. O sistema desenvolvido utiliza comunicações por fios e sem fios entre as unidades periféricas e a unidade central. Foi também desenvolvida uma interface de visualização em dispositivos com acesso à *Internet* dos dados em tempo real sobre o consumo e estado dos sensores de segurança.

Palavras-chave: Sistema, automação, gestão, segurança, energia, integração edifícios, domótica, inmótica, consumo de energia, consumos.

ABSTRACT

In this dissertation the project of an autonomous system of security and energy management is presented.

The current society state implies that we spend most of our lives inside buildings. Whenever possible, we prefer that our simple and routine tasks are performed by stand-alone systems. Joining these two points with the need to manage energy resources and security, we get need of an automation system for buildings with added security and energy monitoring.

Due to the wide range of low cost and good quality devices available on the market, the development of this type of monitoring system becomes less and less complicated. Key technologies used on these systems are mostly open source. These systems present a good alternative to existing systems on the market.

This dissertation describes the state of the art and presents the design of a low-cost system for security and energy management in commercial and non-commercial buildings.

The system was developed with the purpose to monitor energy consumption in buildings, safeguard goods and people, providing the user with a system that stores all the information and allows the analysis of the same. The system developed uses wireless and wire communications between peripheral units and the central unit. A visualization interface has also been developed that allows devices with Internet access to see in real-time data of consumption and security sensors.

Keywords: System, automation, management, security energy buildings, house automation, integration, energy consumption…

AGRADECIMENTOS

A todos os que me ajudaram, professores, amigos e colegas que contribuíram com o seu auxílio e apoio.

Em especial aos meus orientadores, Professor Doutor Mário Gomes e Professor Doutor Paulo Coelho, pela oportunidade e incentivo concedidos durante a realização deste trabalho e ao longo dos últimos anos enquanto meus docentes.

Ao Engenheiro Pedro Neves pela colaboração e disponibilidade durante a realização do protótipo.

Aos meus amigos e à minha família pelo apoio que me deram.

Ao Instituto Politécnico de Tomar (IPT) pelas condições proporcionadas para a realização do projeto;

Este trabalho teve apoio do projeto RIGMEI – *Red Iberoamericana de Generación Distribuida y Microrredes Eléctricas Inteligentes*, com referência RED-713RT04752012, financiado pelo CYTED - Programa Iberoamericano de Ciencia y Tecnologia para el Desarrollo.

Trabalho realizado no âmbito das atividades da Unidade VITA.IPT – Vida Assistida por ambientes Inteligente.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AC	Corrente Alternada
ADC	*Analog Digital Converter*
AVAC	Aquecimento, Ventilação e Ar Condicionado
BT	Baixa tensão
CAN	*Controller Area Network*
CSI	*Camera Serial Interface*
CSMA-CA	*Carrier Sense Multiple Access with Collision Avoidance*
CSV	Comma-separated values
DC	Corrente Contínua
DSI	*Display Serial Interface*
EEPROM	*Electrically-Erasable Programmable Read-Only Memory*
EU	Europa
EUA	Estados Unidos da América
GPIO	General Por
HDMI	*High-Defenition Multimedia Interface*
IEEE	*Institute of Electrical and Electronics Engineers*
I2C	*Inter-Integrated Circuit*
ICSP	*In-Circuit Serial Port*
IDE	*Integrated Development Environment*
IIR	*Infinite Impulse Response*
IOT	*Internet of Things*
IP	*Internet Protocol*
Ip	Corrente no Primário
IPT	Instituto Politécnico de Tomar
Is	Corrente no Secundário
ISO	*International Organization for Standardization*
Mac	*Media Access Control*
MT	Média tensão
MQTT	*Message Queue Telemetry Transmission*

QoS	*Quality of Service*
TCP	*Transmission Control Protocol*
THD	Taxa de distorção harmónica
LAN	*Local Area Network*
LBT	*Listen Before Talk*
M2M	*Machine to Machine*
MISO	*Master In Slave Out*
MOSI	*Master Out Slave In*
PIR	*Passive Infra Red*
PLC	*Power Line Comunication*
PoE	*Power over Ethernet*
PWM	*Phase Width Modelation*
RAM	*Random Access Memory*
Rb	Resistência *burden*
RF	Rádio Frequência
RMS	*Root Mean Square*
RX	*Receiver*
SCK	*Serial Clock*
SDRAM	*Synchronous Dynamic Random Access Memory*
SPI	*Serial Peripheral Interface*
SS	*Slave*
SSH	*Secure Shell*
TP0	*Transport Protocol 0*
TP1	*Transport Protocol 1*
TTL	*Transistor-Transistor Logic*
TX	*Transmitter*
U	Tensão
USB	*Universal Serial Bus*
Vs	*Versus*

LISTA DE SÍMBOLOS

%	Percentagem
A	Ampere
Ah	Ampere-hora
Bit/s	Bits por segundo
€	Euro
F	Farads
GHz	Giga-Hertz
Hz	Hertz
Kbits/s	Quilobits por segundo
KB	Quilobyte
Kbps	Quilobits por segundo
kW	Quilowatt
kWh	Quilowatt-hora
kVA	Quilovolt-ampere
mA	Miliampere
Mbits/s	Megabits por segundo
MHz	Megahertz
m	Metros
ms	Milissegundos
mW	Mili-watt
MW	Mega-watt
MWh	Megawatt-hora
Ω	Ohm
KΩ	Quilo-ohm
V	Volt
VA	Volt-ampere
W	Watt
µF	micro-Farads

Capítulo 1 - Introdução

1.1. Enquadramento

Nos dias que correm, a sociedade passa a maior parte do tempo no interior de edificações onde ao longo da sua existência tem procurado proteção e conforto. À medida que a tecnologia evolui, são desenvolvidas melhorias que conferem aumentos na segurança e no conforto dos edifícios. Uma das formas contemporâneas de proporcionar segurança e conforto é através da automação de tarefas rotineiras em edifícios.

A automação nasceu do desejo humano de desenvolver algo que replicasse as suas ações naturais para a sua recreação, para diminuição dos perigos inerentes a certas ações e para o aumento da produtividade. A automação é por isso fruto da vontade humana de melhorar a sua qualidade de vida e de levar a cabo tarefas que sem a automação seriam humanamente impossíveis.

Por definição, automação exprime a execução de tarefas através de técnicas mecanizadas ou computacionais sem a intervenção humana, o que permite a diminuição do erro inerente à ação humana, a agilização para além dos limites humanos e a execução de tarefas que de outro modo seriam fisicamente impossíveis de realizar pelo homem. Estes aspetos levam à dinamização e otimização da realização de tarefas.

Genericamente, a automação representa a simples substituição de mão-de-obra humana por mão-de-obra "mecanizada" com controlo autónomo. Esta definição não tem em conta a principal importância da automação, que é dotar o homem com sistemas que realizam tarefas de impossível realização para o mesmo, contudo, a intervenção do homem existe sempre, quer na sua conceção, quer na sua manutenção ou mesmo para o seu normal funcionamento.

A automação em edifícios como hoje a conhecemos teve origem na década de 1980 com o aparecimento do computador pessoal, mas tem sido nos últimos anos que o interesse por este tema tem aumentado [1]. A redução de custos associados aos edifícios e a procura constante de melhorar o conforto dos utilizadores têm sido os maiores contribuidores para o aumento da automação em edifícios.

As formas de providenciar conforto vão desde o controlo das variáveis ambientais dos espaços, como a luminosidade e temperatura, e podem estender-se até à segurança da integridade das pessoas e bens, como o controlo de acessos ou a deteção de incêndios.

As exigências económicas e ecológicas atuais obrigam cada vez mais a uma gestão dos recursos energéticos, quer estes sejam fontes consumidoras ou fontes produtoras de energia. Se por um lado a redução dos consumos beneficia diretamente o ambiente devido à diminuição dos impactes ambientais causados pela produção de energia, por outro lado a poupança económica providencia recursos financeiros que podem ser aplicados para uma ainda maior redução dos consumos.

A monitorização dos consumos energéticos dos edifícios origina uma fonte de informação importante que pode ser utilizada na tomada de decisões com vista à diminuição desses mesmos consumos, o que originará uma potencial redução dos custos energéticos do edifício.

De certa forma, ao providenciar segurança providencia-se também conforto devido ao efeito que a segurança tem sobre as pessoas. Por isso e devido à ocorrência de infortúnios acontecerem quando menos se espera, os sistemas de segurança tomam um papel importante quando se trada de garantir a proteção de pessoas contra ameaças em que a sua deteção prévia ou atempada pode evitar consequências de maior.

Os sistemas de monitorização de segurança, ao identificarem a ocorrência de falhas de segurança em tempo real, permitem uma resposta rápida necessária para impedir, reverter ou minimizar as consequências dessas mesmas falhas.

1.2. Objetivos

Este trabalho consiste no estudo e prova de conceito de um sistema inteligente para edifícios que congregue um sistema de gestão de energia e um sistema de segurança.

Pretende-se, quanto ao sistema de gestão de energia, medir e analisar os consumos de energia em edifícios com vista à diminuição do desperdício energético através de processos automáticos e manuais. Verificar a adequabilidade da tarifa energética contratada perante outras ofertas no mercado. Monitorização em tempo real dos custos e ainda possibilidade de monitorização da produção energética efetuada a partir de microgeração.

Quanto ao sistema de segurança, pretende-se assegurar a integridade das pessoas e bens através da monitorização de variáveis, como a presença de fumo e aumento da temperatura ambiente no caso de incêndio ou identificação de intrusão a partir da deteção de movimentos. Ambos os sistemas serão integrados numa central que tomará ações automáticas pré-configuradas, mediante os vários cenários e ajudará os utilizadores na

tomada de decisões. Esta central será munida de uma interface gráfica que permitirá aos utilizadores interagirem remotamente com o sistema.

1.3. Motivação

É evidente que os consumidores domésticos de eletricidade têm pouca influência direta no custo da energia, têm sim influência direta nos consumos que efetuam. Quer isto dizer que para poderem obter uma poupança financeira, no que toca aos gastos com energia, é necessário reduzirem diretamente os seus consumos.

A importância da redução dos gastos energéticos é sobretudo devida à necessidade de atenuar a tendência existente de aumento dos consumos de energia com os objetivos de poupar recursos naturais e diminuir os níveis de poluição causados pela produção de energia.

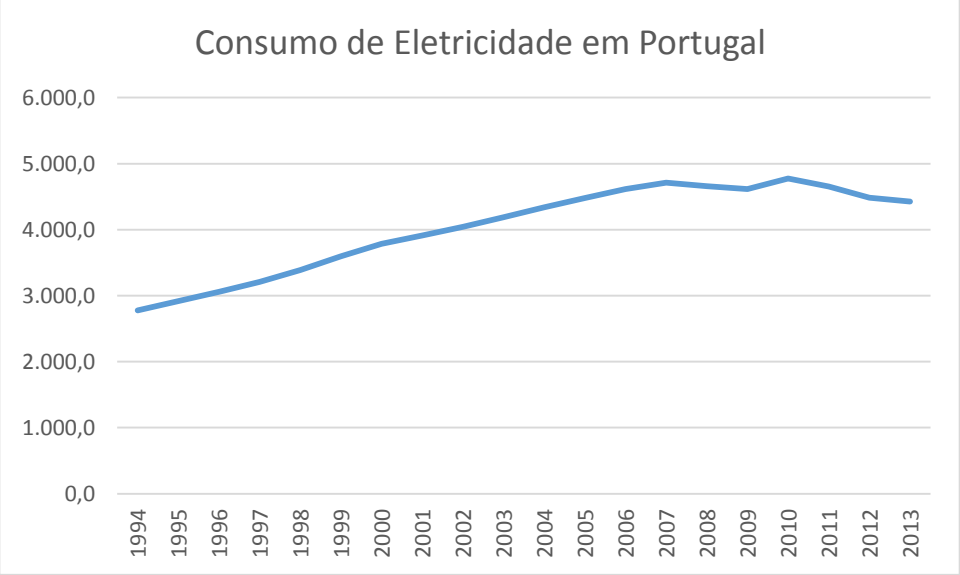


Figura 1: Evolução do consumo de eletricidade em Portugal desde 1994 [2].

Na maior parte dos casos, o consumidor apenas toma conhecimento dos consumos quando os mesmos já foram efetuados há algum tempo, o que ajuda muito pouco na identificação da fonte causadora dos mesmos. É por esta razão que os sistemas de monitorização de energia são importantes pois conseguem em tempo real ajudar na identificação das fontes causadoras de consumos potencialmente desnecessários.

Um sistema de gestão de energia para além de monitorizar os consumos ajuda o consumidor a tomar decisões perante a apresentação de informação sobre os gastos e

possíveis formas de melhoria. O objetivo da integração da monitorização da segurança com um sistema de gestão de energia é o aproveitar da informação e recursos com vista a uma melhor análise e atuação em ambos os sistemas.

A segurança é necessária para proteger em primeiro lugar as pessoas e depois os bens materiais, sendo a alarmística essencial para evitar ou minimizar quaisquer ocorrências que os possam por em risco. Os sistemas de segurança são cada vez mais essenciais devido à consciencialização do número de ameaças que existem nos meios envolventes.

1.4. Metodologia

A dissertação está organizada em cinco capítulos, incluindo o atual capítulo da Introdução.

No **Capítulo 1** faz-se uma Breve introdução ao tema, apresentação dos objetivos gerais do trabalho e da motivação para o desenvolvimento do mesmo.

No **Capítulo 2** apresentam-se fundamentos sobre o que é a automação em edifícios e como funciona. Abordagem às duas áreas deste trabalho, a gestão da Segurança e a gestão da Energia. Avaliação das principais vantagens de sistemas para este efeito.

O **Capítulo 3** corresponde ao estado da arte, onde são analisadas algumas das tecnologias existentes. Descrição dos principais meios de comunicação sem fios e com fios, apresentando-se quais as suas vantagens e desvantagens. Resumo de alguns meios físicos de comunicação. Revisão dos protocolos *KNX, LownWorks* e *X10*. Abordagem ao tema *Internet of Things*. Regulamentação existente para sistemas deste tipo. Apresentação são de produto similar já existente no mercado nacional.

No **Capítulo 4** descreve-se projeto SISGE, apresenta os estudos preliminares ao projeto, assim como o projeto em si. Dimensionamento do protótipo para medição de energia. Escolhas dos elementos de *hardware* e *software* e apresentação das razões e motivos que levaram a essas escolhas. Procedimentos para instalação das várias aplicações. Principais funções utilizadas na programação do sistema.

No **Capítulo 5** são feitas as considerações finais, onde se abordam os aspetos mais relevantes da dissertação. Apresenta ainda os resultados obtidos às simulações efetuadas, erros no sistema e suas causas e desenvolvimentos futuros.

Capítulo 2 - Automação em edifícios

2.1. Definição e conceito

O conceito da automação em edifícios sejam estes habitacionais ou não, nasceu na década de 1980 e tem vindo desde então a ser discutido. Nos últimos anos este tema tem sido cada vez mais abordado devido, sobretudo, às inovações tecnológicas que têm permitido tirar mais partido desde tipo de sistemas.

Hoje em dia o conceito da automação em edifícios (também conhecido como Domótica) é cada vez mais associado aos edifícios chamados de inteligentes. Contudo em termos tecnológicos a ideologia de edifícios inteligentes é mais generalista do que a automação de edifícios, devido a esta última abordar sobretudo sistemas de automação, controlo, gestão de energia e interfaces inteligentes [3].

No que diz respeito ao tipo de edifícios em que os sistemas de automação são aplicados dividem-se em habitacionais e não habitacionais. Nos edifícios habitacionais os sistemas de automação são baseados principalmente na geração de conforto ao utilizador, já nos edifícios não habitacionais os sistemas de automação dedicam-se principalmente à gestão energética e consequentemente à gestão de custos.

Todos os sistemas de automação em edifícios têm como base três componentes comuns, sendo eles os sensores, os controladores e os atuadores.

Nestes sistemas é aplicado maioritariamente o controlo por retroação (ou realimentação), descrito na Figura 2. A Planta é o elemento cujo estado se pretende manter ou alterar. O Sensor é responsável pela leitura do estado da Planta e consequente transformação desse estado em sinal elétrico. O Controlador é o agente responsável pela determinação a partir da diferença entre o estado atual e Referência do estado a manter ou qual o estado para o qual deve ser alterado. Essa alteração é feita pelo Atuador mediante o estipulado pelo Controlador. Este sistema é também conhecido como sistema em malha fechada [4].

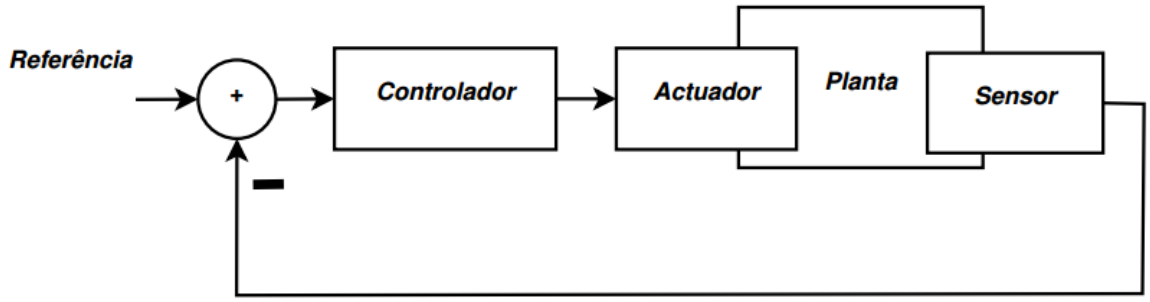


Figura 2: Componentes de um sistema de controlo por retroação [4].

Dos vários sensores passíveis de integração em sistemas de automação em edifícios os mais comuns são os simples sensores lógicos e sensores das variáveis de ambiente (temperatura, humidade e luminosidade).

A informação captada pela parte sensorial do sistema é geralmente interpretada por um dispositivo central pré-programado que tomará decisões baseadas nessa informação e aplicará ou não controlo sobre os atuadores.

Os atuadores são normalmente elementos de iluminação, AVAC (Aquecimento, Ventilação e Ar Condicionado) e controladores de acesso (Portas, Janelas, Persianas).

2.1.1. Gestão de Energia

Nos dias que correm, usamos cada vez mais energia no nosso dia-a-dia, quer essa energia seja na forma de combustíveis fósseis, eletricidade ou qualquer outro tipo de energia. Devido à maior parte dessa energia que consumimos ser proveniente de combustíveis fósseis que têm preços extremamente instáveis, devido aos múltiplos fatores considerados na sua determinação em que se destaca a especulação bolsista, existe cada vez mais a necessidade de controlar os consumos energéticos.

O controlo dos consumos energéticos é essencial porque é a única variável sobre a qual o consumidor tem controlo direto, isto é, não conseguimos controlar o preço da energia que gastamos, conseguimos sim controlar os consumos que efetuamos e daí poupar energeticamente e consequentemente poupar financeiramente [5].

No passado recente e no presente os consumos energéticos, nomeadamente os consumos de eletricidade, são contabilizados pelos distribuidores de energia e apenas algum tempo depois essa contagem é fornecida ao consumidor, geralmente sobre a forma de fatura. As eventuais medidas de poupança a aplicar ficam assim condicionadas porque para além

de não se saber exatamente de que parte da instalação consumidora os gastos vêm, o consumo já foi efetuado e não se sabe quando, apenas se sabe que foi num determinado período entre contagens.

Assim, de modo a gerir de forma eficiente os consumos de eletricidade é necessário que estes sejam monitorizados regularmente, em ciclos diários e semanais, e se possível que forneçam a indicação da zona ou equipamento específicos que efetuaram o consumo. Uma gestão eficiente da carga consumidora é importante quando se está perante tarifas energéticas repartidas no tempo.

Os sistemas de monitorização providenciam informação mais detalhada dos consumos, que pode ser utilizada para ajudar os consumidores a identificarem oportunidades de redução de consumos [5]. Para além disso, controla o estado de operacionalidade de dispositivos, podendo interagir sem intervenção do consumidor perante fatores de tempo ou ambientais.

Não só os consumos podem ser monitorizados, as fontes de energias renováveis podem também ser monitorizadas e consoante a produção de energia pode ser dada informação ao consumidor de que fonte está a vir a energia que está a consumir no momento.

2.1.2. Gestão de Segurança

A segurança é cada vez mais levada em conta, não só pela legislação que obriga a que certos requisitos sejam cumpridos mas também devido à crescente perceção por parte do público, em geral do aumento das ameaças à sua segurança e aos seus bens.

Geralmente quanto se aborda o tema da segurança em edifícios aponta-se para a problemática dos incêndios. No entanto para além dessa problemática existem outros riscos para a segurança, como o risco de inundação, contaminação do ar ou intrusão [6].

Os meios de segurança em edifícios dividem-se em duas classes, meios de segurança ativa e meios de segurança passiva. Os meios de segurança passiva são aqueles que estão inseridos na construção ou fazem parte da arquitetura dos edifícios, como por exemplo portas e janelas, os materiais utilizados na construção ou a disposição dos acessos do edifício [7].

Os meios de segurança ativa são todos aqueles que de uma maneira manual ou automática são acionados perante a deteção de falhas de segurança e têm como objetivo principal a deteção ou controlo de falhas de segurança aquando da sua ocorrência. Exemplos

de meios de segurança ativa são: os de centrais de desenfumagem, expressores ou alarmes de intrusão.

A monitorização da segurança é importante não só para a deteção de falhas de segurança na altura em que ocorrem, mas também para a sinalização prévia que poderá levar à eliminação de falhas antes que as mesmas ocorram.

Capítulo 3 - Estado da Arte

3.1. Tecnologia existente

Antes de dar início ao projeto do SISGE foram estudadas as principais opções tecnológicas existentes no mercado de modo a garantir que as escolhas efetuadas são as adequadas para este caso de estudo, tendo em conta os mais diversos aspetos.

Os sistemas de automação em edifícios utilizam várias tecnologias para levar a cabo a gestão da energia, segurança e controlo dos edifícios. A tecnologia presente nos sistemas atuais vai desde simples equipamentos analógicos já com décadas de utilização a equipamentos mais complexos (*e.g.* digitais) e de utilização específica [1].

Em geral, o controlo de todo o sistema é efetuado pela unidade central de controlo pré-programada para o efeito. O controlo é feito com base na informação captada pelos sensores ou instruções inseridas pelos utilizadores no sistema através de dispositivos ou aplicações que interagem com o mesmo. Outros sistemas utilizam pequenas unidades de controlo que fazem a gestão de subsistemas enquanto uma unidade central de controlo supervisiona todas as operações.

A comunicação entre a unidade central de controlo e os sensores ou atuadores pode ser efetuada através de redes *fieldBus* (barramento), por redes de pares de fio, fibra ótica, *power line communication* (PLC) ou através de redes sem fios. O ponto comum entre estas tecnologias é que qualquer uma delas forma uma rede de comunicação entre vários elementos do sistema [8].

3.1.1. Meios de transmissão

Atualmente são utilizadas várias tecnologias para a criação de redes para sistemas de automação. Estas redes podem ser dedicadas aos sistemas de automação ou partilhadas com redes já existentes, servindo assim essas redes como meios de comunicação para redes virtuais dos sistemas de automação.

Os sistemas de automação de edifícios utilizam geralmente duas redes, em que a rede principal liga os principais sistemas de controlo, geralmente através de redes *Ethernet*, *WiFi, USB*. A rede secundária interliga os equipamentos base de todo o sistema (sensores, controladores e atuadores). As redes secundárias são mais simples e fáceis de aceder devido

ao baixo nível dos sistemas neles contidos e utilizam na maioria dos casos as tecnologias descritas de seguida.

A utilização de tecnologias com ou sem fios em sistemas de automação em edifícios depende sempre dos requisitos da instalação. Como por exemplo das normalizações necessárias à aplicação, das frequências de operação (largura de banda necessária), da distância de comunicação, entre outros fatores. De seguida estão descritas as principais tecnologias de redes com e sem fios utilizadas no segmento tecnológico em estudo.

Redes sem fios

Bluetooth

A tecnologia *Bluetooth* foi originalmente implementada pela indústria das telecomunicações e apenas atingia velocidades de transferência até 1 Mbits/s. Nas versões mais recentes a velocidade pode ir até aos 24 Mbits/s [9].

Devido às melhorias que têm sido desenvolvidas para esta tecnologia ao nível das velocidades de transferência de dados e à redução dos consumos de energia, o *Bluetooth* deixou de ser uma tecnologia virada para os dispositivos de comunicação móveis e passou também a ser uma tecnologia utilizada em sistemas de entretenimento, saúde, segurança, etc.

Mesmo depois de grandes avanços alcançados por esta tecnologia, continua a ser limitada em relação ao seu custo mais elevado, distância de comunicação e consumo de energia quando comparada com outras tecnologias como o *ZigBee*, *Z-Wave* entre outras. Contudo, devido à sua vasta utilização em dispositivos móveis de comunicação e entretenimento, a sua utilização em sistemas de automação em edifícios pode ser essencial para a interação com estes sistemas móveis.

WiFi

A tecnologia *WiFi* é uma tecnologia de transmissão de dados de alta largura de banda baseada na norma IEEE 802.11. Na sua última versão comercializada, pode atingir velocidades até 1300 Mbits/s a 5 GHz e distâncias de comunicação sem antenas ou repetidores até cerca de 30 metros dentro de edifícios e cerca de 90 metros no exterior [10]. Em sistemas de automação para edifícios o *Wifi* é sobretudo utilizado como meio de transmissão para redes virtuais de automação devido à sua rápida velocidade de transferência.

A vantagem do *WiFi* em relação a outras tecnologias similares é a sua vasta implementação em dispositivos de consumo móveis o que facilita a sua integração nos sistemas de automação.

ZigBee

A tecnologia *ZigBee* é um meio de comunicação sem fios baseada na norma IEEE 802.15. Esta tecnologia apresenta caraterísticas de baixo consumo energético e facilidade de criação de redes. A sua aplicação em sistemas de automação em edifícios tem sido bastante elevada, particularmente em aplicações de controlo e monitorização.

A distância de comunicação entre nós de redes *ZigBee* pode variar entre os 10 e os 100 metros dependendo da sua implementação. Dado que, uma rede *ZigBee* pode ser implementada de modo a que qualquer dispositivo possa receber e reenviar informação até a mesma chegar ao seu destino, o que pode aumentar em muito a distância entre o dispositivo de origem e o dispositivo de destino da informação.

Uma rede *ZigBee* suporta até 64000 dispositivos o que é uma vantagem para aplicações em grandes edifícios.

Devido ao espectro de frequências utilizado no *ZigBee* variar de região para região do globo, a tecnologia *ZigBee* pode trabalhar sobre 3 frequências distintas que correspondem também às máximas frequências de transferência de dados, sendo as mesmas 869 MHz, 915 MHz e 2,4 GHz. A velocidade máxima de transferência de dados ronda dos 250 Kbits/s [11].

Tabela 1: Resumo das principais caraterísticas da tecnologia ZigBee por regiões [11].

	EU/Japão	EUA	Todo o Mundo
Banda	868MHz	915MHz	2,4GHz
Canais	1	10	16
Velocidade	20 Kbits/s	40 Kbits/s	250 Kbits/s

Z-Wave

A *Z-Wave* é uma norma de comunicação pertencente à *Zensys Ltd* que é também a entidade responsável pelo licenciamento aos seus utilizadores da norma *Z-Wave*. Foi desenvolvida com a intenção de garantir os requisitos dos sistemas de monitorização e controlo remotos em aplicações residenciais ou pequenos comércios [12].

Quando comparada com a norma IEEE 802.11 difere sobretudo na velocidade de transferência que apenas atinge velocidades até 100 kbits/s devido à sua pequena largura de banda [13]. A *Z-Wave* permite até um máximo de 232 nós por rede e tem uma distância máxima de funcionamento na ordem dos 30 metros. A banda de funcionamento depende de região para região do globo terrestre.

A sua utilização é adequada em aplicações onde a distância entre dispositivos e o conjunto de dados a serem enviados for reduzida, caso contrário esta tecnologia não será uma boa opção.

Resumo sobre os meios de transmissão sem fios

Tabela 2: Tabela de resumo para comparação das várias tecnologias de transmissão sem fios.

	Normalização	**Gama frequências**	**Distancia**	**Velocidade máxima**	**Nós p/rede**	**Consumo**
Bluetooth	IEEE 802.15.1	2,4-2,485GHz	100 m	24 Mbps	7	>2mW
WiFi	IEEE 802.15.1	2,4/5GHz	92 m	1300 Mbps	255	>20mW
ZigBee	IEEE 802.15.4	868M/2,4GHz	75 m	250 kbps	64000	1mW
Z-Wave	Zensys/ EN300220	868,4-869,85MHz	100 m	100 kbps	232	<1mW

Redes com fios

Ethernet

A *Ethernet* é uma tecnologia de comunicação baseada no envio de pacotes de informação utilizada em Redes de Área Local (LAN). É normalizada pela norma IEEE 802.3 e o seu funcionamento é baseado no método de difusão, ou seja, as transmissões são feitas por um meio físico comum [14].

O envio de informação através de *Ethernet* pode atingir velocidades até 10Gbits/s dependentemente do meio de comunicação utilizado (Pares de fios, fibra ótica, coaxial, etc). A sua utilização em sistemas de automação em edifícios é bastante comum devido às rápidas

velocidades de transmissão. Geralmente as redes *Ethernet* são utilizadas nos sistemas de automação como ligação entre as redes secundárias e o *interface* com o utilizador.

Devido à existência de cada vez mais dispositivos com capacidade de se ligaram a redes *Ethernet*, a integração desses equipamentos em sistemas de automação fica assim mais facilitada.

Internet Protocol e Transmission Control Protocol

Internet Protocol (IP) é o protocolo mais utilizado nas transmissões em redes de computadores. O IP define a dimensão dos pacotes, o cabeçalho da informação, o *hostname* entre outras informações para a transmissão de informação dentro da rede. O *Transmission Control Protocol* (TCP) define se a informação foi recebida ou não sobre o *Internet Protocol* [15].

A conjugação TCP/IP é utilizada como solução para transmissão de informação na internet. Quando aplicada a sistemas de automação de edifícios permite a ligação do sistema a redes externas, dotando assim os sistemas com um meio de comunicação fiável e seguro para a transmissão de informação entre o sistema e os clientes externos à rede.

PLC – Power Line Communication

No caso de aplicações mais económicas e simples para instalações de sistemas de automação em edifícios, a tecnologia PLC tem grandes vantagens em relação a outros meios de transmissão pois utiliza a rede elétrica já existente para transmitir informação, dispensando assim a necessidade de instalação de uma rede exclusiva para o sistema de automação.

A tecnologia PLC utiliza a rede criada para a instalação elétrica dos edifícios para transmitir informação por um domínio de frequências entre os 5kHz até aos 500kHz. A velocidade de transmissão da tecnologia PLC dependente dos protocolos utilizados [16].

As comunicações de banda larga por redes PLC foram normalizadas em 2010 pela norma IEEE 1901. Outras normalizações são também adotadas para as redes PLC em edifícios, como as normas *HomePlug* que possuem várias revisões da sua norma inicial do ano 2001 em que originalmente para uma frequência de funcionamento entre os 4,5MHz e os 21MHz era possível atingir velocidades até 8,2Mbps. A última versão é a *HomePlug* AV2 que pode atingir velocidades até 1Gbps [17].

FieldBus

A tecnologia de comunicação *fieldBus* é bastante utilizada em sistemas de automação em edifícios, abrangendo vários protocolos de comunicação. Esta tecnologia apareceu na década de 1980 quando surgiu a necessidade de substituir as ligações ponto-a-ponto entre os sensores ou atuadores e as unidades de controlo por ligações digitais únicas onde a informação é transmitida em série e multiplexada no tempo.

A transmissão da informação nos protocolos *fieldBus* é feita em pacotes com pequenas dimensões e enviados sequencialmente. Esta é a principal vantagem em relação a protocolos de envio de pacotes em paralelo devido principalmente à redução do número de condutores necessários.

Devido aos requisitos específicos dos vários setores da indústria, foram desenvolvidas adaptações baseadas na tecnologia *fieldBu*s, como por exemplo *Controller Area Network* (CAN), *EtherCat*, *ModBus*, *LonWorks* entre outros.

3.1.2. Protocolos de comunicação

Evolução dos protocolos

Os protocolos de comunicação foram definidos para regular a transferência de informação e garantir duas funções essenciais às redes, a primeira é que qualquer elemento da rede pode enviar ou receber informação, a segunda é que esses elementos podem sincronizar o envio entre si. Estas duas funções são vantajosas em relação a outras tecnologias pois permitem uma elevada quantidade de tráfego de informação sem aumentar o tempo de espera de cada elemento para envio da mesma, sendo esta vantagem essencial para aplicações de controlo.

Há mais de uma década protocolos como o *European Installation Bus* (EIB) ou *Lonworks* lideravam as tecnologias de comunicação nos sistemas de automação para edifícios. Atualmente, o protocolo *LonWorks* continua a ser uma tecnologia líder a par do protocolo KNX. Outro protocolo utilizado desde o aparecimento dos sistemas de automação em edifícios e que continua a ser utilizado hoje é o X-10 [18].

KNX

Figura 3: Logotipo KNX [19].

As organizações europeias *European Installation Bus* (EIB), *European Home Systems Association* (EHSA) e *Batibus Club International* (BCI) desenvolveram o protocolo KNX normalizado pelas normas técnicas para as redes de comunicação em sistemas de automação de edifícios EN 50090 e ISO/EIC 14543 [19].

O intuito do seu desenvolvimento foi o de criar um padrão europeu para sistemas deste tipo e com os objetivos de melhorar o desempenho das comunicações, permitir modos de funcionamento *Plug&Play* e incentivar as empresas de prestação de serviços nas áreas das energias e telecomunicações a envolverem-se no tema. Tecnicamente o objetivo deste padrão foi desenvolver um sistema de gestão de energia, controlo e gestão de segurança. Todos estes objetivos foram definidos tendo em conta o cumprimento dos requisitos e necessidade das instalações, quer estas sejam habitacionais ou de outro tipo.

O KNX herdou do EIB o envio de informação sequencial através de ligações por barramento, podendo utilizar arquiteturas centralizadas ou descentralizadas em que os vários dispositivos comunicam entre si permitindo que a qualquer momento sejam adicionados novos dispositivos à rede.

Todos os dispositivos construídos segundo o padrão KNX têm de obedecer a estritas regras de qualidade, sendo que têm de cumprir a norma ISO 9001. Os dispositivos KNX são totalmente compatíveis entre si o que torna esta tecnologia altamente flexível.

Configuração

As redes KNX podem ser configuradas recorrendo ao *software* de desenvolvimento *Engineering Tool Software* (ETS) que permite a personalização das funções de todos os dispositivos ligados à rede. A configuração dos dispositivos através do *software* ETS é chamada de *System mode* (S-mode). Por defeito, os dispositivos KNX permitem ainda uma configuração *Easy mode* (E-mode). Em modo de configuração simples (E-mode) os

dispositivos são programados com os padrões de fábrica cuja função é realizar uma determinada tarefa específica para o dispositivo em causa (figura 4).

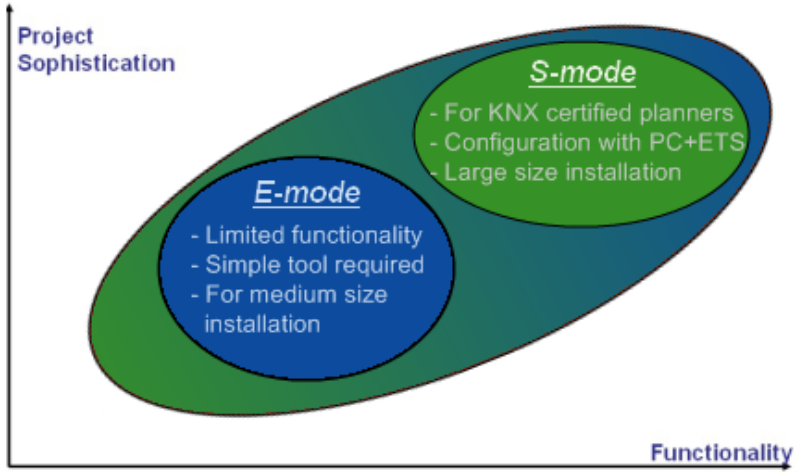


Figura 4: Relação Funcionalidade-Sofisticação do projeto dos modos de configuração KNX, modo simples *E-mode*, modo de sistema *S-Mode* [19].

Meios de transmissão

Vários meios de transmissão são atualmente utilizados no protocolo KNX, sendo que inicialmente apenas era utilizado os pares de fio. Com a evolução alargaram-se os meios de comunicação para a transmissão de sinais pelas instalações elétricas, por radiofrequência, por infravermelhos e por *Ethernet*. Estes meios foram identificados do seguinte modo:

- **KNX/TP0 e KNX/TP1**: A transmissão é feita através de um par de condutores com velocidade de transmissão até 4,8 kbps para TP0 e até 9,6 kbps para TP1. Os pares são também utilizados para fornecer a alimentação aos dispositivos com uma tensão de 24V CC e com um mínimo de 21V. O protocolo de comunicação *Carrier Sense Multiple Access with Collision Avoidance* (CSMA-CA) é utilizado para evitar a colisão de informação e melhorar o aproveitamento da largura de banda disponível. Este foi o primeiro meio de comunicação e é atualmente o mais utilizado em instalações KNX.
- **KNX/PL110 e KNX/PL132**: A transmissão é feita pelas instalações elétricas através de sinais com frequências na gama dos 110 KHz que permitem velocidades até 1,2 kbps para PL110 e frequências na gama dos 132 KHz que permitem velocidades até 2,4 kbps. A distância máxima de transmissão é de 600m, para distâncias superiores será necessário a utilização de repetidores. A modulação do

sinal utilizada é a *Spread Frequency Shift Keying* (S-FSK), semelhante à *Frequency Shift Keying* (FSK) com a diferença da separação entre as portadoras ser maior, o que permite a diminuição das interferências causadas pelos harmónicos. A modulação FSK altera a frequência da onda portadora consoante o sinal binário que se quer enviar.

- **KNX/RF**: A transmissão por radiofrequência permite distâncias em campo aberto até 300 metros, para distâncias superiores ou no interior de edifícios onde as estruturas provocam a diminuição da força do sinal será necessário a utilização de repetidores. Os sinais operam em frequências na gama dos 868 MHz e atingem velocidades até 16 kbps. A tecnologia *Listen Before Talk* (LBT) é utilizada para evitar a colisão de mensagens.
- **KNX/IR**: A transmissão por infravermelhos permite distâncias apenas em campo aberto até 12 metros e é utilizada geralmente apenas para a comunicação de controlos remotos que trabalham numa gama de frequências entre o 10 e os 70 KHz.
- **KNXnet/IP**: A transmissão por redes *Ethernet* com velocidades até 10 Mbit/s no padrão IEC 802.2, em que a sua utilização é feita sobretudo para interligar sistemas ou efetuar a ligação dos sistemas às centrais de controlo. Estas ligações são particularmente utilizadas para a transferência de informação através de protocolos IP entre instalações muito afastadas.

Envio da informação

Os conjuntos de informação que fluem entre dispositivos ligados à rede são chamados de telegramas e são constituídos por sequências binárias. A informação de cada telegrama emitido por um determinado dispositivo é convertida em binário e posteriormente em sinal analógico simétrico entre os pares de fio que constituem o barramento.

O envio de informação para o barramento só pode ser efetuado quando o barramento estiver livre, caso contrário o dispositivo que está a enviar informação necessita de aguardar que o barramento fique livre. Os dispositivos verificam constantemente a existência de outras mensagens no barramento com o intuito de evitar colisões de telegramas. Quando um dispositivo está a enviar um telegrama e deteta um telegrama com prioridade superior ao seu, vindo de outro dispositivo na rede, pára imediatamente o envio do seu telegrama e fica

a aguardar que o barramento fique novamente livre para que possa enviar de imediato o seu telegrama.

Endereçamento

A comunicação entre dispositivos é possível devido a todos os dispositivos KNX utilizarem a mesma linguagem e terem todos endereços físicos únicos para cada dispositivo. Os endereços são hierarquizados por Linhas e Zonas ou Áreas, atribuídos por instalação e têm o seguinte formato, como se mostra na Figura 4.

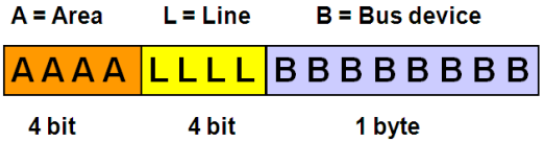


Figura 5: Modelo de endereçamento de dispositivos, A-bits de área, L-bits de linha e B-bits de barramento [20].

Outro modo de endereçamento existente é o endereçamento de grupo, podendo este ter dois (Figura 6) ou três níveis (Figura 7) definidos, ou níveis definidos consoante as necessidades da instalação (Figura 8).

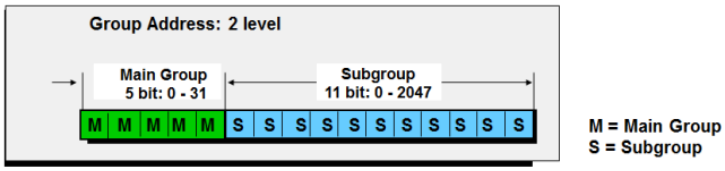


Figura 6: Modelo de endereçamento de grupos com dois níveis M-grupo principal e S-subgrupo [20].

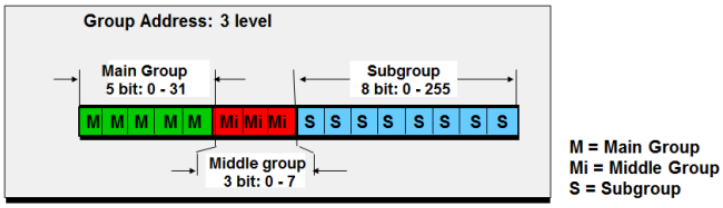


Figura 7: Modelo de endereçamento de grupos com três níveis, M-bits de grupo principal, Mi-bits de grupo intermédio e S- bits de subgrupo [20].

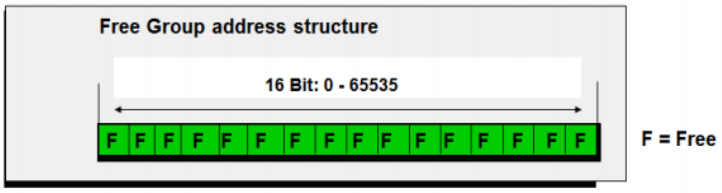


Figura 8: Modelo de endereçamento de grupos livre, F-bit de livre atribuição [20].

Um telegrama enviado de um dispositivo para outro é identificado com o endereço origem da informação e com o endereço de destino, sendo que o endereço de destino pode corresponder a um único dispositivo ou a um conjunto de dispositivos dependendo do tipo de endereçamento de grupo utilizado. Cada telegrama só pode ser enviado para um endereço de grupo, no entanto cada dispositivo pode ter vários endereços de grupo. Quando um telegrama é enviado todos os dispositivos recebem esse telegrama, no entanto apenas os dispositivos cujo endereço se encontra no telegrama tomam nota do mesmo, sendo este o princípio dos sistemas descentralizados.

Cada linha apenas suporta até um máximo de 64 dispositivos, sendo que para adicionar mais dispositivos ao sistema terá de se acrescentar uma segunda linha, o que levará a que a linha existente e a segunda linha tenham de ser ligadas a uma linha que serve apenas para ligar as duas ao sistema.

Vantagens e desvantagens

A vantagem principal deste protocolo é ser considerado por vários organismos internacionais como o protocolo padrão para as comunicações em sistemas de automação em edifícios [19]. Outras vantagens incluem os altos padrões de qualidade dos equipamentos KNX, e o elevado número de produtores de dispositivos KNX o que leva a uma grande e diversificada oferta e *software* único de configuração.

A desvantagem principal do protocolo KNX é o seu custo elevado em relação a outras tecnologias, devendo-se isto principalmente ao facto dos fabricantes terem de cumprir a norma de qualidade ISO 9001 e serem cobrados para que os seus produtos tenham reconhecimento em como cumprem os padrões do protocolo, o que aumenta o custo do produto final.

LonWorks

Figura 9: Logotipo LonWorks [21].

A tecnologia *LonWorks* foi desenvolvida pela *Echelon Corporation* no início da década de 1990 e tem vindo desde então a ser implementada em redes de sistemas de automação em edifícios. Foi desenvolvida com a intenção de respeitar os requisitos da maioria das aplicações de controlo mas a sua aplicação é feita sobretudo em edifícios não habitacionais onde o fator preço é menos importante do que os fatores fiabilidade e desempenho [22].

Um dos objetivos do desenvolvimento da tecnologia *LonWorks* era fornecer, ao contrário de outras tecnologias, uma tecnologia que qualquer produtor de dispositivos pudesse usar nos seus produtos sem ser cobrado por isso, o que permite reduzir os custos em relação a outras tecnologias e melhorar a sua adoção. Existiam no entanto várias contradições a este objetivo, pois qualquer dispositivo *LonWorks* tinha de ser baseado no microcontrolador *NeuronChip*, desenvolvido especificamente para o sistema *LonWorks* e que utiliza a linguagem de programação *LonTalk* desenvolvida também para o mesmo.

Apenas existiam até 1999 três fabricantes do microcontrolador *Neuron* o que diminuía bastante o fator concorrencial da produção destes dispositivos e implicava preços mais elevados. Em 1999 o protocolo foi aberto para outros processadores o que aproximou bastante esta tecnologia do seu objetivo inicial.

Os sistemas *LonWorks* utilizam uma arquitetura descentralizada que permite a partilha de informação direta entre os sensores e os atuadores ligados ao sistema, quer estes sejam físicos ou aplicações.

Meios de transmissão

Os meios de transmissão utilizados para *LonWorks* podem ser pares de fio, *power line communication* (PLC), radiofrequência, infravermelhos entre outros. O meio de transmissão mais utilizado nos sistemas *LonWorks* é o de pares de fio que utiliza um mecanismo anti-colisão - *Carrier Sense Multiple Access* (CSMA) - desenhado

especificamente para este protocolo. Os pares de fio podem atingir velocidades de transferência na ordem dos 78 kbits/s [22].

O microcontrolador *NeuronChip* é o centro desta tecnologia e contém a aplicação, pilha de endereçamentos e os métodos de acesso ao barramento de comunicação. Cada *NeuronChip* tem um identificador único chamado de *NeuronID* que permite o endereçamento de qualquer dispositivo nas redes *LonWorks*. O *NeuronID* tem uma dimensão de 48 bits e é definido durante a produção do microcontrolador.

Configuração

Existem várias ferramentas para configuração das redes *LonWorks*. Estas ferramentas são baseadas num método de desenvolvimento chamado de *LonWorks Network Services* (LNS) que utiliza uma base de dados em sistema operativo Windows para guardar todas as informações relativas à rede *LonWorks*.

Nas aplicações LNS é possível definir os dispositivos ligados à rede (instalar dispositivos), os seus níveis de hierarquia, regras da rede, assim como criar e gerir bases de dados referentes à rede.

A configuração dos dispositivos pode também ser feita através de programação direta do *NeuronChip* em linguagem de programação C baseada em ANSI C.

Envio da Informação

A comunicação é feita através de pacotes de informação os quais contêm a informação a enviar, a que dispositivo se destinam, o caminho para chegar a esse dispositivo, dados de controlo, informação da origem da mensagem e um código para deteção de erros.

O protocolo *LonWorks* utiliza um sistema de mensagens *peer-to-peer*, quer isto dizer que não existe um agente controlador das mensagens na rede e por isso algoritmos de deteção de conflitos têm de ser utilizados. Para um dispositivo responder a uma mensagem enviada por outro dispositivo é necessário assegurar que o dispositivo que enviou a mensagem está a aguardar uma resposta vinda do dispositivo que recebeu a mensagem. Para isso cada nó tem uma tabela de endereços com todos os dispositivos ligados na sua rede onde assinala quais os dispositivos com os quais partilha a mesma rede. A configuração destas opções é feita através da ferramenta de gestão da rede.

Endereçamento

As redes *LonWorks* são hierarquizadas e estruturadas por endereçamentos lógicos constituídas por domínios, sub-redes e dispositivos individuais chamados de nós [23].

Um domínio *LonWorks* é uma coleção lógica de dispositivos (nós) num ou mais canais físicos e é definido para que a comunicação possa ser apenas feita pelos dispositivos que estejam no domínio. Esta comunicação pode ser feita por partilha de informação para a rede, partilha de informação apenas para um dispositivo ou para um grupo de dispositivos.

Cada domínio pode ter um identificador de 0, 1, 3 ou 6 bytes de tamanho e é normalmente descrito em notação hexadecimal. O identificador de 0 bytes é apenas utilizado para propósitos de gestão da rede. Geralmente são utilizados identificadores de domínio com 1 byte de tamanho devido a este valor fazer parte da estrutura das mensagens do protocolo *LonWorks*. Ao utilizar um identificador de 6 bytes este tem mais 40bits do que o identificar de 1 byte o que se traduz num menor desempenho da rede. Os identificadores de domínio mais extensos devem ser utilizados de forma a garantir que um nó só não é ativado erradamente por interferência, na receção da mensagem [24].

Cada domínio *LonWorks* pode conter até 255 sub-redes cada uma com 64 nós e 127 grupos de dispositivos. As sub-redes são identificadas por valores numéricos de 1 a 255, já os grupos são identificados por de 1 a 127. As sub-redes e grupos de dispositivos são utilizados para comunicação em grupo, a diferença é que os grupos podem ao contrário de sub-redes conter nós em canais diferentes.

Existem cinco formas de endereçar mensagens definidas no protocolo *LonWork*s, através do domínio, sub-rede, grupo, código do dispositivo e *NeuronID*. Cada nó contém uma tabela com os endereços que podem ser utilizados pela aplicação de gestão na rede, para especificar um endereço destino para uma mensagem através de referência na lista. Este método é também conhecido como endereçamento implícito. Em alternativa o nó pode especificar o endereço completo de destino da mensagem, sendo este método conhecido como endereçamento explícito. Uma variedade de ferramentas é fornecida onde é possível programar os endereços e configurar os nós [25].

A estrutura das mensagens contém a informação sobre a versão do protocolo do barramento, o formato dos pacotes, o formato dos endereços e o tamanho do endereço. A dimensão do endereço em si é variável e depende do esquema de endereçamento utilizado.

Caso um domínio esteja definido, o endereço do domínio poderá também ser definido na mensagem. Os pacotes consistem em dados de transporte e da aplicação.

Vantagens e desvantagens

O protocolo *LonWorks* tem uma arquitetura ao nível do dispositivo, quer isto dizer que não necessita de outros intervenientes para além dos emissores e recetores de sinais. Este é um aspeto positivo mas também se pode tornar negativo devido à possibilidade de interação entre dispositivos errados.

Desde a sua invenção, a tecnologia *LonWorks* tem definido uma série de regras para que exista compatibilidade entre dispositivos *LonWorks* de produtores distintos. Apesar desta compatibilidade qualquer instalação *LonWorks* necessita, para além dos gastos normais com equipamento, de ser configurada por *software* que não é de livre utilização o que requer custos acrescidos ao valor final da instalação. A adição de novas funcionalidades ao protocolo apenas é efetuada pela empresa detentora do mesmo.

X-10

Figura 10:Logotipo X-10 [26].

O protocolo X-10 foi o primeiro protocolo de comunicação *Power Line Comunication* a surgir e foi desenvolvido na década de 1970 pela *Pico Electroncis*, com o objetivo de tornar mais fácil a implementação de aplicações de automação em edifícios. O nome X-10 deve-se a ser o décimo projeto da empresa. É um protocolo aberto, isto é, qualquer fabricante pode produzir dispositivos que usem o protocolo sem serem cobrados por isso.

A arquitetura é descentralizada, o que significa que os vários dispositivos ligados à rede podem comunicar entre si sem a necessidade de uma unidade central de controlo, o que torna a tecnologia mais flexível e imune a falhas totais dos sistemas. Inicialmente este

protocolo foi desenvolvido apenas para comunicações unidirecionais, no entanto surgiram dispositivos capazes de receber e enviar informação através deste protocolo.

Devido à sua tecnologia pouco complexa é uma tecnologia acessível, a sua instalação é económica e de fácil implementação comparando com outras tecnologias. Por utilizar uma comunicação PLC é possível instalar ou expandir sistemas que utilizem este protocolo sem a necessidade de adicionar novos meios de comunicação, o que possibilita também uma instalação a qualquer momento, não tendo a mesma que estar preparada de origem nos edifícios.

Protocolo

O protocolo X-10 utiliza um método de transmissão simples com uma estrutura de dados de 8 bits precedidos de um código de início de mensagem pré-definido. A transmissão da informação é feita através do envio de um sinal a 120 kHz durante 1 ms quando a onda sinusoidal em corrente alternada é zero, sendo o sinal enviado até um máximo de 50µs depois da passagem por zero. O valor lógico "1" é definido por um impulso seguido de uma ausência de impulso. O valor lógico "0" é definido pela ausência de um impulso seguido de imediato por um impulso. Todos os dispositivos X-10 têm deteção da passagem da corrente por zero (*zero-crossing*) para que os emissores e recetores de sinais estejam sincronizados. No caso da aplicação do protocolo X-10 sobre redes trifásicas o envio dos impulsos deverá ser feito três vezes por ciclo de relógio, com desfasamentos adequados a cada fase.

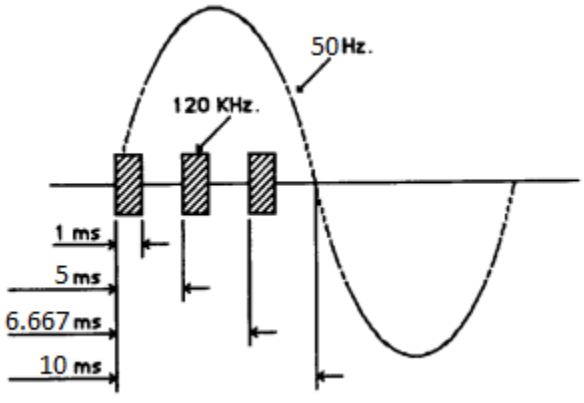


Figura 11:Envio de impulso a 120kHz num rede trifásica de 50Hz, impulsos desfasados consoante as diferentes fases [27].

Estrutura das mensagens

Qualquer dispositivo de uma rede X-10 recebe todas as informações enviadas sobre a forma de mensagens emitidas para a rede, logo qualquer dispositivo tem de ser capaz de endereçar cada mensagem para que seja apenas reconhecida pelo dispositivo a que se destina. Para isso o protocolo X-10 implementa uma estrutura de mensagem ordenada da seguinte forma:

- **Código de início**: O envio de mensagens no protocolo X-10 é iniciado com o código lógico “1110" de modo a ser sinalizado o início de uma mensagem. Ao contrário dos outros códigos constituintes da mensagem, para o código de início não são enviados bits complementares.
- **Código de casa**: 4 bits destinados à representação das letras de A a P.
- **Código do dispositivo ou função**: 4 bits com valores decimais entre 1 e 16 que representam o dispositivo ou a função a executar seguidos de 1 bit que sinaliza se os 4 bits anteriores são o código de dispositivo ou uma função. O envio deste último bit deve-se ao facto de serem enviadas duas mensagens, uma delas com a identificação do dispositivo e outra com a função a ser executada pelo mesmo.

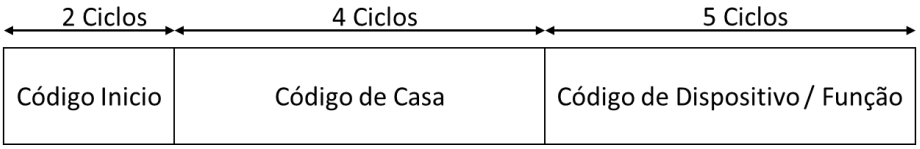


Figura 12: Mensagem X-10.

Os bits correspondentes ao Código de Casa e Código de Dispositivo/Função são enviados juntamente com o seu complementar, sendo que o envio de uma mensagem demora 11 ciclos a ser completado. Nas figuras seguintes (Figura 13 e Figura 14) estão resumidos os códigos possíveis de serem enviados e a sua estrutura.

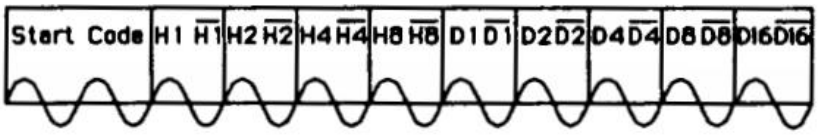


Figura 13: Sequência de envio dos vários bits constituintes de uma mensagem em X-10, envio alternado em meio ciclo de relógio [27].

House Codes						Key Codes					
	H1	H2	H4	H8		D1	D2	D4	D8	D16	
A	0	1	1	0	1	0	1	1	0	0	
B	1	1	1	0	2	1	1	1	0	0	
C	0	0	1	0	3	0	0	1	0	0	
D	1	0	1	0	4	1	0	1	0	0	
E	0	0	0	1	5	0	0	0	1	0	
F	1	0	0	1	6	1	0	0	1	0	
G	0	1	0	1	7	0	1	0	1	0	
H	1	1	0	1	8	1	1	0	1	0	
I	0	1	1	1	9	0	1	1	1	0	
J	1	1	1	1	10	1	1	1	1	0	
K	0	0	1	1	11	0	0	1	1	0	
L	1	0	1	1	12	1	0	1	1	0	
M	0	0	0	0	13	0	0	0	0	0	
N	1	0	0	0	14	1	0	0	0	0	
O	0	1	0	0	15	0	1	0	0	0	
P	1	1	0	0	16	1	1	0	0	0	
					All Units Off	0	0	0	0	1	
					All Lights On	0	0	0	1	1	
					On	0	0	1	0	1	
					Off	0	0	1	1	1	
					Dim	0	1	0	0	1	
					Bright	0	1	0	1	1	
					All *Lights* Off	0	1	1	0	1	
					Extended Code	0	1	1	1	1	
					Hail Request	1	0	0	0	1	①
					Hail Acknowledge	1	0	0	1	1	
					Pre-Set Dim	1	0	1	X	1	②
					Extended Data (analog)	1	1	0	0	1	③
					Status = on	1	1	0	1	1	
					Status = off	1	1	1	0	1	
					Status Request	1	1	1	1	1	

Figura 14:Lista de códigos e comandos X-10 e os seus respetivos bits.

Na figura anterior, os números 1, 2 e 3 têm a seguinte interpretação:

1 – Verifica a existência de outros transmissores X-10 na rede, caso existam um novo código de casa terá de ser atribuído.

2 – O bit D8 representa o bit mais significativo do nível, enquanto os bits H1, H2, H4 e H8 representam os 4 bits menos significativos.

3 – Este código de função é seguido de 1 byte que representa a informação geralmente proveniente de conversores Analógico-Digital. Não existem ciclos de intervalo entre o código de função e os 8 bits de dados. Os primeiros 8 bits de informação poderão corresponder ao número de bytes que vão ser enviados.

São enviados grupos de duas mensagens, uma para identificar o destino e outra para identificar a função, separadas por três ciclos de relógio entre si. Durante os três ciclos de relógio é envido o valor lógico “000000”. No caso de funções *dimming* são enviados grupos de três mensagens sem ciclos de relógio de intervalo.

Vantagens e Desvantagens

O X-10 é uma solução tecnologicamente simples e de baixo custo devido a utilizar a instalação elétrica dos edifícios como meio de comunicação. Em edifícios relativamente grandes o funcionamento poderá ser afetado devido à perda do sinal causada por interferências, o que levará à necessidade de incorporação de repetidores de sinal na instalação, aumentando assim o seu custo.

Os meios de comunicação são limitados à instalação elétrica onde a velocidade de envio de informação atinge os meros 20bits/s, limitando esta tecnologia a aplicações muito simples. Para além disso o X-10 não permite a utilização de dois dispositivos em simultâneo na mesma rede devido à inexistência de um controlador de tráfego na rede.

A existência de filtros capacitivos nas entradas de alimentação de dispositivos ligados à rede poderão eliminar os impulsos enviados por dispositivos X-10.

MQTT

O *Message Queue Telemetry Transport* (MQTT) é um protocolo de livre utilização, simples, leve e de fácil implementação para publicação/subscrição de mensagens [38]. Foi desenvolvido em 1999 e tem vindo desde então a ser utilizado nas mais diversas áreas da industria. Em 2014 o protocolo MQTT tornou-se um padrão internacional após deliberação da *Advancing Open Standards for the Information Society* (OASIS).

O protocolo MQTT serve de intermediário de mensagens e foi desenvolvido sobretudo para utilização em redes em que a largura de banda é pequena ou instável e em dispositivos com limitações ao nível do processador ou memória.

Os clientes, que podem ser dispositivos ou aplicações, ligam-se a um servidor conhecido como *broker* através de ligações TCP. Os clientes podem subscrever tópicos no servidor conseguindo assim ter acesso às mensagens que nesse tópico são publicadas por outros clientes. Os tópicos têm uma estrutura idêntica à utilizada para identificar sistemas de arquivos (*i.e.* casa/temperatura).

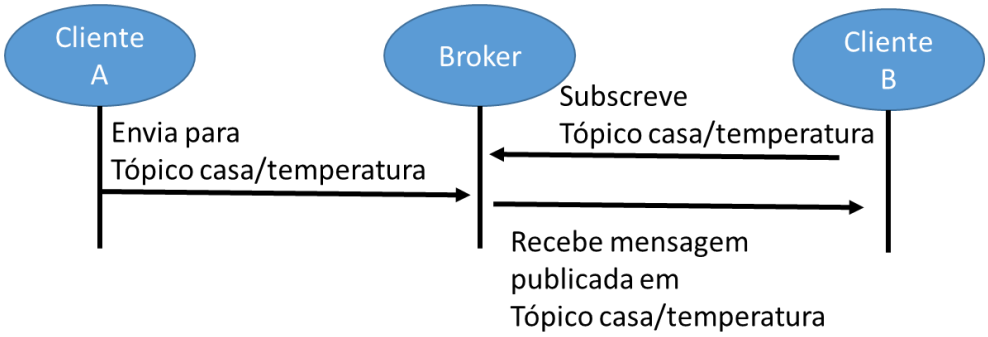


Figura 15:Funcionamento MQTT, envio de mensagem do Cliente A e receção no Cliente B.

As principais funcionalidades do *MQTT* são:

- Possibilidade de publicar e subscrever tópicos;
- Meio de transporte agnóstico ao conteúdo da mensagem;
- Ligações de rede feitas por *TCP*/*IP*;
- Três níveis de Qualidade de Serviço (*QoS*) de entrega disponíveis:
    1. A mensagem é enviada apenas uma vez, correspondente a um valor de *QoS* igual a 0. As mensagens são entregues caso as condições da rede assim o permitam, podendo ocorrer perdas de mensagem ou duplicações. Este é o nível mais baixo de garantia de entrega. Um exemplo da utilização deste nível é em sensores de temperatura, visto a sua informação não ser crítica;
    2. A mensagem é enviada pelo menos uma vez, correspondente a uma valor de *QoS* igual a 1. Todas as mensagens são entregues pelo menos uma vez, existindo a garantia de entrega, não ficando de parte a hipótese de entregas duplicadas;
    3. A mensagem é enviada exatamente uma vez, correspondendo a um valor de QoS igual a 2. Todas as mensagens são garantidamente entregues uma e uma só vez aos subescritores do canal.
- Estrutura de mensagens pequena e em que as trocas de informação são minimizadas para reduzir o trafego na rede;
- Mecanismo de notificação em caso de fim inesperado da ligação ao servidor por parte de um cliente.

O *MQTT* dispõe de um modo de identificação dos tópicos através de dois *wild-card*, "+" e "#". O "+" é utilizado para identificar todos os subtópicos num determinado nível

hierárquico. Já o "#" identifica todos os subtópicos no nível onde é colocado e todos os adjacentes.

As mensagens têm um cabeçalho definido, este está representado na Tabela 3 onde o byte 1 contém os campos com o tipo de mensagem e as *flags DUP*, nível *QoS* e *RETAIN* e o byte 2 contém a dimensão restante da mensagem. No DVD em anexo encontram-se os valores padrão dos bits do cabeçalho assim como o significado detalhado das *flag*.

Tabela 3: Cabeçalho MQTT [38].

bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
byte 1	Tipo de Mensagem				*DUP*	nível *QoS*		*RETAIN*
byte 2	Dimensão restante da mensagem							

O centro do protocolo *MQTT* é o intermediário de mensagens, o *broker*. Este servidor recebe as mensagens publicadas nos tópicos e reenvia-as para os clientes que subscreveram esses mesmos tópicos. Podem ser aplicados filtros no *broker* às mensagens/tópicos para limitar o acesso dos clientes.

Vantagens e Desvantagens

A baixa largura de banda necessária para o envio de mensagens é a principal vantagem deste protocolo. A boa utilização da largura de banda disponível tornam este protocolo eficiente, isto devido em grande parte à simplicidade do mesmo. A utilização de meios de transmissão considerados comuns, como Ethernet ou Wi-Fi, potenciam o futuro desde protocolo.

A desvantagem deste protocolo deve-se sobretudo à utilização de ligações TCP/IP, o que restringe a sua utilização a equipamentos dotados de tal ligações.

3.2. Internet Of Things

Recentemente surgiu um novo conceito ao qual se chama a Internet das Coisas (do inglês *Internet of Things* - IoT). Este conceito apesar de novo não introduz uma nova tecnologia no que diz respeito aos sistemas de automação e à comunicação entre sensores, atuadores e controladores. O que fez foi reintroduzir o conceito das redes descentralizadas a uma dimensão maior, através de tecnologias já existentes e a partir de uma identificação única para cada elemento (as coisas).

O objetivo principal da IoT é a criação de redes inteligentes onde todos os elementos comunicam entre si e tomam decisões perante a informação que partilham, através dos mais variados meios de comunicação, sendo que a troca de informação e as ações tomadas pelos elementos é feita sem a intervenção humana.

O conceito IoT está estruturado em três camadas, representadas na Figura 16. A primeira camada contém o núcleo da Internet comum, onde se encontram os meios de comunicação, os controladores de endereçamento e os servidores onde a informação é guardada, analisada e disponibilizada aos utilizadores. A segunda camada é a atual Internet, de onde fazem parte os nossos dispositivos de *interface* como os computadores pessoais ou os *Smartphones*. Esta camada faz a ligação entre o núcleo da Internet e a IoT. A terceira camada é a camada física da IoT, onde é feita a ligação de qualquer dispositivo (coisa), podendo este ser integrado como cliente, servidor ou ambos, servindo para os mais variados fins.

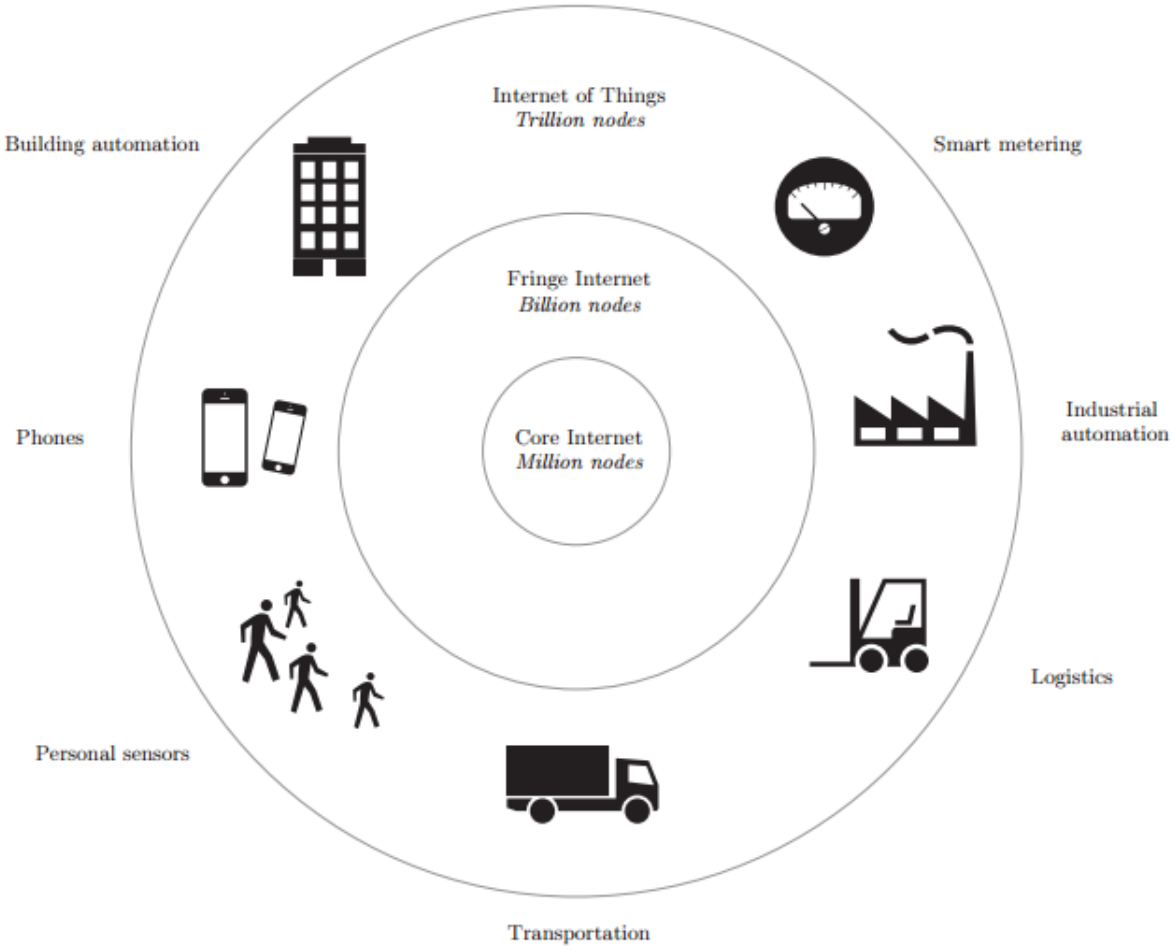


Figura 16: Visão Internet Of Things, previsão do número de nós em cada camada da IoT [28].

A IoT é algo em que temos vindo a trabalhar há algum tempo sem nos darmos conta de tal, pois há imenso tempo que adicionamos elementos periféricos aos nossos principais sistemas, como por exemplo impressoras de rede ou centralizamos *interfaces* como as Televisões Inteligentes. É igualmente aplicado nos mais diversos setores da sociedade, como por exemplo na indústria onde a montagem de componentes é controlada para que haja garantia que o equipamento final contém todos os componentes. Outras aplicações são experimentais como a gestão da produção de eletricidade numa determinada região através de sensores instalados o mais próximo do consumidor final que monitorizam o consumo e providenciam informação para que a produção possa ser ajustada.

A aplicação do conceito requer uma vasta componente tecnológica, tanto ao nível sensorial como ao nível da transmissão da informação. As redes IoT exclusivas têm como principal objetivo a minimização dos custos associados às mesmas por não ser necessário utilizar por exemplo uma rede de fibra ótica para enviar a informação de sensores de presença. Isto porque a maior parte da informação que se pretende recolher e partilhar é de pequenas dimensões e não requer meios de comunicação complexos, sendo que redes IoT exclusivas baixam significativamente os custos com a transmissão da informação.

Um caminho longo ainda terá de ser percorrido até que este conceito se torne minimamente aceitável pelo público em geral. Um dos problemas espectáveis será a adoção de protocolos e tecnologias como padrão para a *Internet of Things*. Isto porque, caso não exista uma padronização do conceito incorreremos no mesmo problema que temos na atualidade, em que na maioria dos casos apesar de ser possível não é direta a integração de sistemas que utilizem protocolos ou tecnologias diferentes, sendo a uma escala muito maior na *Internet of Things* devido ao elevado número de dispositivos.

3.3. Regulamentação

A norma ISO 50001, adotada em Portugal como NP EN ISO 50001, especifica quais os requisitos necessários para estabelecer, implementar, manter e melhorar Sistemas de Gestão de Energia. Estes requisitos foram fundamentados de modo a que as organizações que adotem esta norma possam seguir um processo contínuo de otimização do desempenho energético.

Esta norma é baseada noutras normas de gestão organizacionais como a norma de Gestão da Qualidade - ISO 9001, Gestão Ambiental - ISO 14001, Segurança Alimentar –

ISO 22000 ou Segurança da Informação ISO 27001. O processo de melhoramento utilizado pela norma ISO 50001 é o processo ou ciclo *Plan-Do-Check-Act* (PDCA) (Figura 17). Este processo de melhoria visa a identificação dos problemas para elaboração de planos de ação, execução dos planos de ação, controlo da execução, eficácia dos planos, atuação sobre eventuais desvios aos planos de ação e normalização.

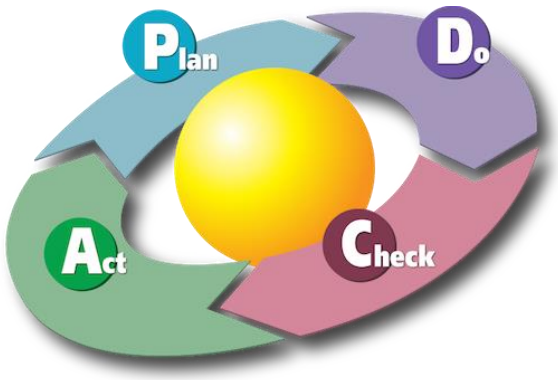


Figura 17:Ciclo de gestão PDCA utilizado na aplicação de normas [29].

A implementação desta norma poderá trazer uma redução de custos e dos consumos energéticos da organização, sendo que a sua implementação apenas fará sentido caso a organização consiga tirar partido da mesma.

Não existindo limites para o tipo e dimensões da organização que deseje seguir esta norma, a certificação da mesma obriga à organização ter definidos os objetivos e planos de ação que garantem a implementação de todo o processo de melhoria contínua de acordo com a norma.

As políticas de consumo energético são da responsabilidade da organização e são de livre arbítrio, não existindo políticas de consumo energético definidas na norma.

3.4. Produtos existentes no mercado

Algumas das soluções disponíveis foram abordadas com o intuito de conhecer as tecnologias utilizadas no mercado Nacional no que diz respeito a sistemas de controlo e monitorização.

Domatica

A empresa portuguesa "Domatica" disponibiliza soluções de monitorização e controlo para instalações de quaisquer dimensões e tipologias e ainda soluções de aquisição

de informação sensorial e controlo à distância, através de meios de transmissão controlados por dispositivos que obedecem a protocolos *Modbus* ou KNX.

As soluções de monitorização e controlo têm como dispositivo central o M2M *Gateway* que armazena a informação vinda diretamente dos sensores ligados ao mesmo e comunica via GPRS ou *Ethernet* através de *software*. Este dispositivo permite a ligação direta nas suas entradas digitais ou analógicas de sensores de estado, contadores de impulsos ou medidores analógicos, podendo ainda ligar-se via *Modbus* ou KNX a outros dispositivos que interligam a mais sensores ou atuadores, sendo estes últimos chamados de I/O *Controller's*.

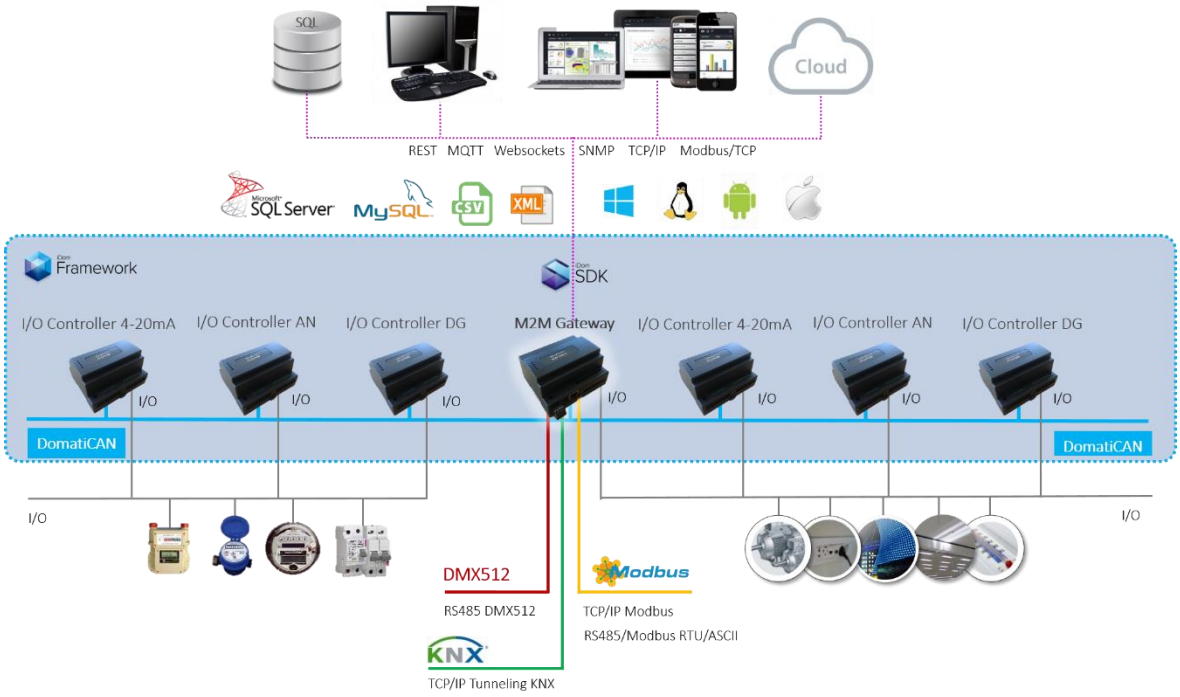


Figura 18: Arquitetura genérica de solução Domatica [30].

Os sensores e atuadores que ligam ao M2M *Gateway* ou aos I/O *Controller* não necessitam de ligações com valores de tensão ou correntes padronizadas visto os dispositivos coletores de informação serem configuráveis através da ferramenta *Driver Builder Tool* onde são configurados os *drivers* de acoplamento dos referidos periféricos. Para além disso, a ligação de dispositivos compatíveis com protocolos *Modbus* ou KNX é direta.

A extração dos dados armazenados no M2M *Gateway* pode ser feita através da extração em ficheiro CSV com o *software Log2Data*, através da ferramenta *Log2Report* que periodicamente produz relatórios sobre a informação que se encontra nos dispositivos ou

ainda através de ligações diretas por protocolos DMX512, *Modbus* ou KNX, serviços *Web* com interfaces *RESTfull* ou *WebSocket*s. A informação recolhida diretamente pode ser depois interpretada por *software* de produção de relatórios.

Figura 19:M2M Gateway Domatica [30].

O M2M *Gateway* é disponibilizado em modelos com várias combinações de portas de entrada e saídas analógicas ou digitais (6 portas de entrada e 6 portas de saída), com ou sem PWM. Contém interfaces de ligação por Ethernet, GSM/GPRS, RS232, RS485 e *DomaticaCAN Bus*. Consegue armazenar informação vinda dos periféricos a ele ligados até 150000 registos. Os preços variam entre versões para valores desde 250€ aos 370€, sendo o *software* de livre utilização.

Figura 20:I/O Controller Domatica [31].

Os I/O *Controller* dispõem de recursos próprios o que significa que quando adicionados a uma rede acrescentam recursos à mesma em vez de consumirem os existentes. A ligação entre I/O *Controller* é feita apenas através do protocolo *DomaticaCAN Bus*, podendo em alternativa funcionar sozinho como elemento central de uma rede. São disponibilizados em modelos com várias combinações de portas de entrada e saídas

analógicas ou digitais (16 portas de entrada e 16 portas de saída), com ou sem PWM. Os preços variam entre os 250€ e os 730€ para as várias versões, dependo do tipo de portas do dispositivo.

Capítulo 4 - Projeto SISGE

O Sistema Integrado de Segurança e Gestão de Energia (SISGE) foi desenvolvido com o intuito de estudar quais as vantagens da utilização de um sistema deste tipo. Foi dada mais relevância à parte do sistema relacionada com monitorização de energia, por ser aquela em que existem menos estudos desenvolvidos por parte do VITA.IPT.

O sistema desenvolvido é capaz de medir os consumos com uma precisão aceitável, nomeadamente o valor da tensão e da corrente, a potência ativa, o fator de potência e a energia consumida. A partir do valor da energia são apresentados os custos diários, semanais, mensais e anuais tendo em conta o serviço de energia contratado pelo consumidor.

Em termos de segurança, um sensor de deteção de presença foi incorporado no sistema, sendo este capaz de detetar a presença humana que origina a ativação de alarmes.

Tendo em conta desenvolvimentos futuros deste sistema, foram simulados alguns cenários de produção de energia e deteção de falhas de segurança.

O sistema apresentado é composto por uma unidade central de controlo e duas unidades periféricas (Figura 21).

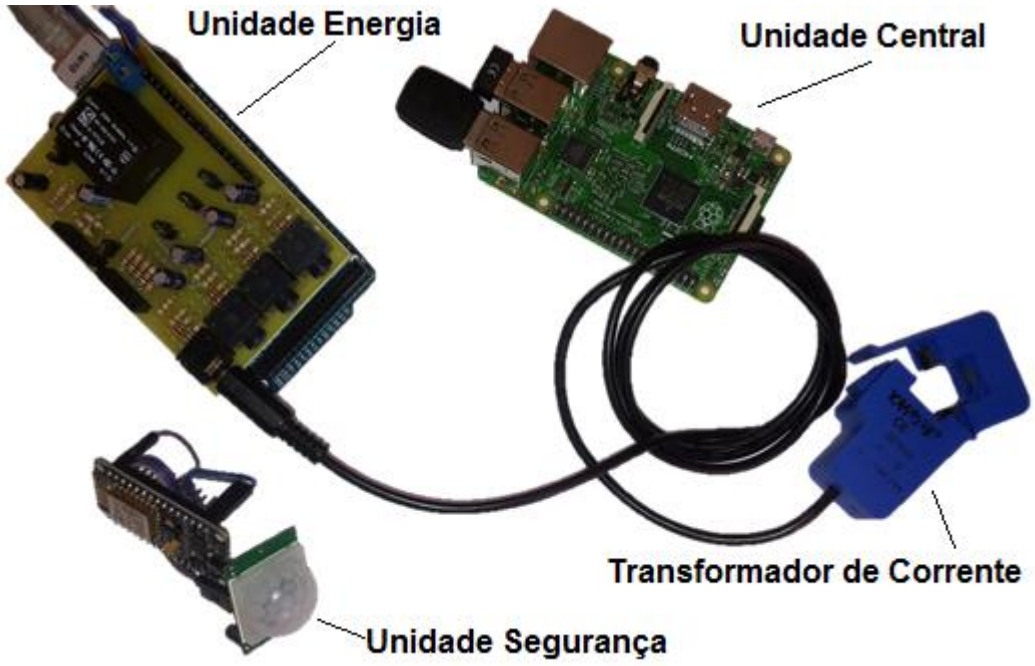


Figura 21: Protótipo SISGE desenvolvido.

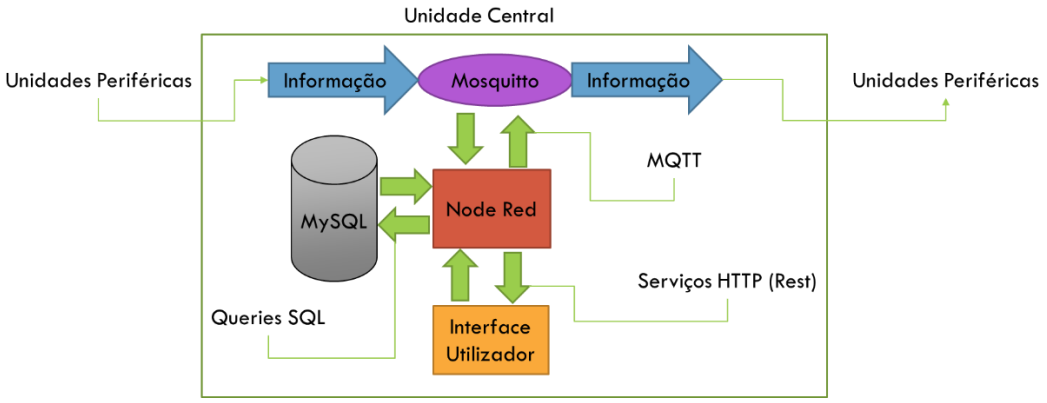


Figura 22: Diagrama do fluxo de informação do SISGE.

A informação vinda das unidades periféricas é guardada numa base de dados que é consultada pela aplicação SISGE. Esta é de acesso *web* e permite a visualização do consumo em tempo quase real, assim como o estado de sensores lógicos. A página principal da aplicação SISGE está representada na Figura 23.

Figura 23: Página inicial da aplicação SISGE.

4.1. Estudos preliminares

O projeto SISGE foi desenvolvido em diversas etapas. Inicialmente fez-se a análise das variáveis que seriam necessárias obter, assim como quais as tecnologias que melhor se ajustariam às necessidades do sistema. As etapas mais relevantes foram a definição da estrutura do sistema, a escolha dos equipamentos principais, o dimensionamento dos componentes de aquisição de dados, a programação do sistema e o desenvolvimento da interface gráfica.

O desenvolvimento deste projeto foi feito a pensar no caso de instalações de habitação, podendo ser replicado facilmente a instalações de serviços e comércio. Os testes de validação da aplicação foram efetuados numa habitação T2 com uma potência contratada de 3,45 kVA e com consumos normais para uma instalação desta tipologia. A habitação não tem nenhum sistema de produção de energia através de fontes renováveis, contudo foi contemplada uma possível implementação de um sistema destes.

As escolhas efetuadas relativas ao *hardware* tiveram em conta a análise efetuada ao Estado da Arte e a necessidade de manter o custo do sistema o mais baixo possível.

4.1.1. Estrutura do Sistema

O sistema deverá ser capaz de adquirir informação proveniente de sensores, manter um registo dessa informação e apresentá-la numa interface gráfica. O intuito de juntar a gestão da segurança e a gestão da energia é o de criar um sistema único, centralizando a interação do utilizador num só ponto do sistema, através da interface gráfica.

Pretende-se adquirir os dados dos consumos energéticos efetuados na habitação, e monitorizando assim em tempo real o valor eficaz da tensão e corrente, a potência ativa, a potência reativa, fator de potência e a energia consumida. A parte do sistema relacionada com a segurança deverá interpretar informação recebida de um ou mais sensores e, se necessário, efetuar as ações mais adequadas autonomamente, *i.e.* sem a consulta do utilizador. A informação adquirida é apresentada numa interface gráfica, a ser utilizada dentro e fora da rede do sistema. No fundo, este sistema corresponde um sistema de domótica.

De acordo com os pressupostos apresentados nos capítulos anteriores, e analisadas as vantagens e desvantagens das várias opções tecnológicas, optou-se por um sistema com uma arquitetura centralizada. Uma unidade central levará a cabo a gestão da informação e suporte à interface de interação com o utilizador. Unidades periféricas de captura de dados enviarão informação proveniente dos sensores para a unidade central. A unidade central enviará ordens de comando para quaisquer atuadores que estejam ligados às unidades periféricas. Não foi escolhida uma arquitetura descentralizada porque esta pressupunha um maior investimento em *hardware*. Por outro lado, uma arquitetura centralizada torna-se também de desenvolvimento menos complexo e ajusta-se perfeitamente à intensão de criar um sistema único.

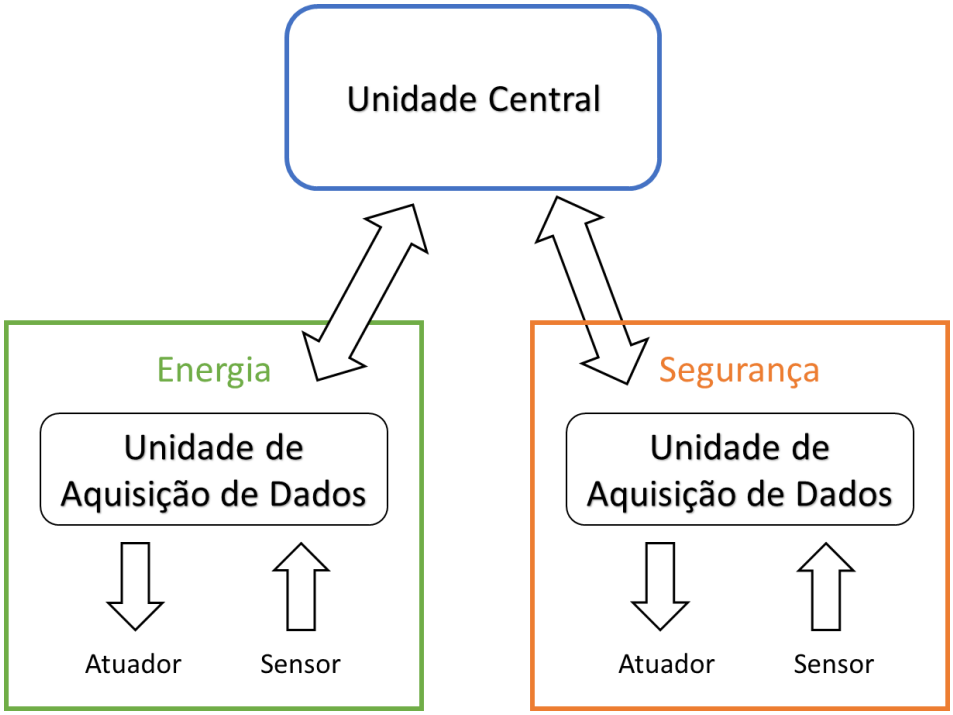


Figura 24: Diagrama de blocos geral do sistema.

4.1.2. Principais componentes

Unidade Central

A unidade central do sistema é constituída por um computador. A escolha deste dispositivo, para funcionar como cérebro do sistema, deve-se à sua flexibilidade de adaptação às mais variadas aplicações. O processamento de informação é mais rápido, devido à sua maior capacidade de processamento, podendo assim processar informação a uma velocidade que permite a sua visualização em tempo real.

O armazenamento temporário (memória volátil) e permanente de informação é também melhor pelo facto da capacidade de armazenamento ser maior, quando comparado com outros tipos de equipamento, como por exemplo microcontroladores. Isto leva-nos à possibilidade de guardar mais, quantidade e com maior detalhe, informação vinda dos sensores do sistema global.

O custo de aquisição deste equipamento é a sua maior desvantagem, sendo que o consumo energético deste sistema também deverá ser tido em conta. Contudo, o computador é um equipamento amplamente difundido em que existe um computador na maioria dos lares.

Unidades periféricas

Para as unidades de aquisição de dados definiu-se que seriam comandadas por microcontroladores, devido ao seu custo reduzido e à sua flexibilidade de ajuste perante as tarefas a executar.

Os microcontroladores são microprocessadores orientados para operações de controlo, ou seja, interação entre sensores e atuadores, e estão presentes em grande parte dos equipamentos eletrónicos que utilizamos no dia-a-dia. Os microcontroladores são desenvolvidos para executarem aplicações específicas, ao contrário dos computadores pessoais que são preparados para correrem um variado conjunto de aplicações.

Os microcontroladores são dispositivos autónomos constituídos por um processador, memória não volátil (*Flash*), memória Volátil (*RAM*) e portas de entrada e saída de informação programáveis. Outros periféricos são incluídos juntamente com os anteriormente nomeados, tais como temporizadores, conversores analógicos-digitais, geradores de impulso (relógio), entre outros. A junção dos vários periféricos num só circuito integrado reduz o número de componentes do sistema, dimensão do produto, consumo e custo global do sistema.

A opção por microcontroladores é vantajosa uma vez que, geralmente, cada família de microcontroladores utiliza um conjunto de instruções de programação específicas que ajudam a melhorar o desempenho do sistema devido a estas já estarem otimizadas.

Leitura de grandezas elétricas

Para efetuar uma monitorização dos consumos energéticos é necessário calcular o valor da potência consumida através dos valores eficazes da tensão e corrente medidos em cada instante. A leitura do valor da corrente é efetuada por um transformador de corrente de núcleo separado (*split core*). A escolha deste tipo de sensor em oposição a outros, tais como circuitos integrados ou transformadores de corrente com núcleo sólido, deveu-se à não interferência com a instalação elétrica da habitação, caso contrário o fornecimento de energia teria de ser interrompido para que o sensor pudesse ser instalado. O aspeto negativo deste tipo de sensor quando comparado com sensores de circuito integrado é a sua classe de precisão inferior.

O esquema elétrico do sensor de corrente está representado na Figura 25, onde T é o transformador de corrente, *Ip* é a corrente no primário, *Rb* é a resistência de *burden* que produz uma tensão U, proporcional à corrente Is induzida no secundário.

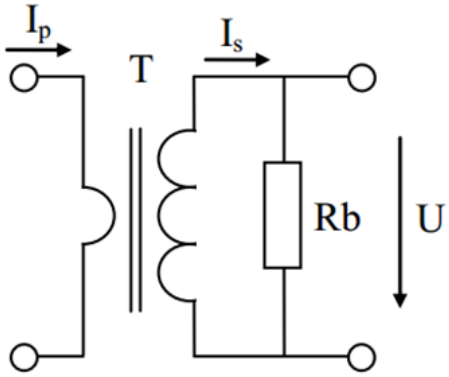


Figura 25: Esquema elétrico do sensor de corrente.

O princípio de funcionamento do transformador de corrente é conhecido como efeito de *Hall*. Este efeito refere-se à tensão produzida quando um condutor é colocado sob influência de um campo magnético, e disposto de forma perpendicular à corrente e campo magnético [32]. A tensão produzida varia consoante a dimensão do campo magnético criado pela corrente que passa pelo condutor. O campo magnético gerado pela corrente aumenta proporcionalmente à mesma.

A amostragem da tensão da rede é feita através do enrolamento secundário de um transformador. Este transformador reduzirá o valor da tensão que ronda os 230V AC para valores que possibilitam a sua medição através de um microcontrolador.

Comunicações

Os meios de comunicação destinam-se à troca de informações entre as unidades periféricas e a unidade central.

Das várias tecnologias disponíveis, considerou-se que a utilização da rede de comunicações de pares de cobre, já montada na habitação, seria a mais adequada. Este meio de comunicação pode atingir velocidades de comunicação elevadas, dependendo do protocolo utilizado. A desvantagem deste meio deve-se sobretudo à pouca proteção contra interferências externas, dependendo este fator da existência ou não de blindagem nos cabos.

Gestão da informação

A informação produzida pelos sensores e enviada para a unidade central pelas unidades periféricas será guardada numa base de dados. O registo em base de dados da informação visa a análise imediata e futura da informação, assim como a correta distribuição e ordenação. A vantagem de guardar a informação numa base de dados, em relação a outros suportes como ficheiros *CSV* ou texto, deve-se sobretudo à capacidade de proteção contra erros na formatação da informação e permitir a disponibilização da informação a vários intervenientes.

A apresentação da informação ao utilizador será feita através de uma aplicação *Web*. Ao utilizar este meio, o acesso à informação poderá ser feita a partir de qualquer dispositivo que tenha acesso à rede local, através do endereço de IP interno ou fora da rede através de IP externo. A vantagem de utilizar uma aplicação *Web* é sobretudo não necessitar de instalar qualquer *software* no dispositivo, podendo ser qualquer dispositivo que tenha um *browser* de *Internet*.

4.2. Tecnologias adotadas

O diagrama representado na Figura 26 apresenta de forma simplificada os principais componentes do Projeto SISGE. Este diagrama teve origem nas tecnologias adotadas que neste ponto serão descritas.

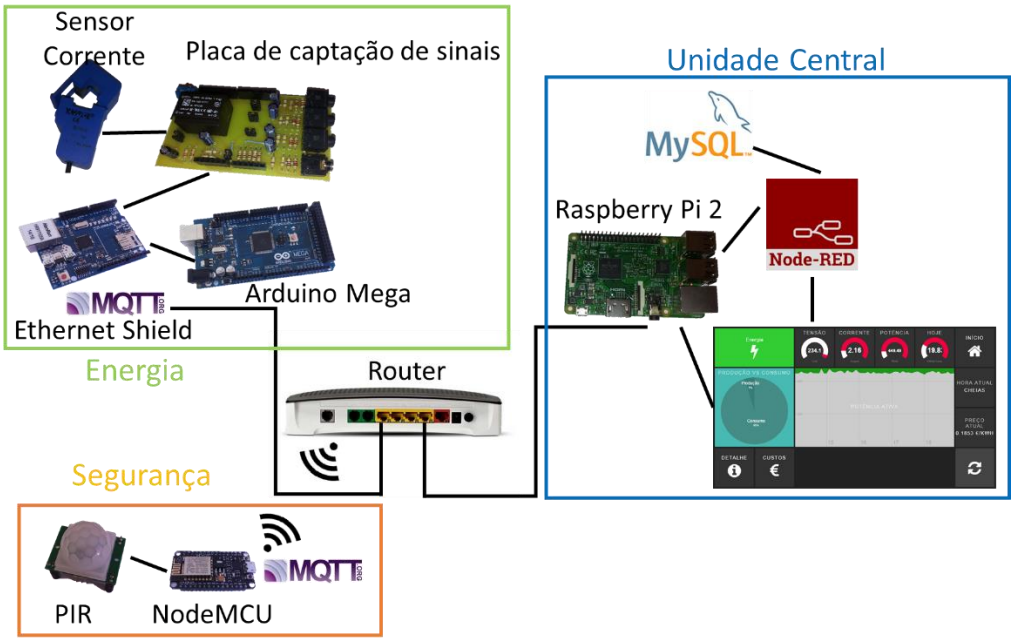


Figura 26: Diagrama geral descritivo dos dispositivos utilizados.

4.2.1. Unidade central

Tendo em conta a necessidade de proceder à manipulação, armazenamento e disponibilização de informação vinda das unidades periféricas e ainda de controlar essas mesmas unidades, foi escolhido para dispositivo de unidade central do sistema um computador *Raspberry Pi 2*. Esta escolha foi baseada no reduzido custo deste equipamento, quando comparado com um computador pessoal normal, e à possibilidade de interação futura deste dispositivo diretamente com sensores e atuadores, através das portas *General Purpose Input/Output* (GPIO) que possui.

O computador *Raspberry Pi* foi desenvolvido pela *Raspberry Pi Foundation* com o objetivo de promover a aprendizagem das tecnologias de informação e comunicação em escolas e países de terceiro mundo [33].

Figura 27:Computador Raspberry Pi 2.

A unidade central de processamento do *Pi 2* é baseada num processador *ARM Cortex-A7* de 32bits, tendo quatro núcleos com relógio a 900MHz. Esta unidade de processamento está integrada no interior de um *system on a chip (soc) Broadcom BCM2836* juntamente com um processador gráfico *Broadcom VideoCore IV*, este com uma frequência de relógio de 250MHz. Contém 1 Gigabyte de memória SDRAM, 4 portas USB, 40 portas GPIO e uma porta Ethernet. O bloco de portas GPIO está caraterizado no anexo A. O armazenamento não volátil é feito num cartão de memória *microSD* colocado no *slot* existente para o efeito. Neste caso foi utilizado um cartão de 16 Gigabyte de memória da marca Samsung com uma velocidade mínima de leitura e escrita de 10Mbit/s.

Possui também quatro portas *USB* versão 2.0, uma porta *HDMI*, uma porta *Display Serial Interfa*ce (DSI), uma porta *Camera Serial Interface* (CSI) e uma porta combinada áudio/vídeo composto.

O *Raspberry Pi 2* utiliza uma distribuição *Linux* de nome *Raspbian* especificamente compilada para este dispositivo, sendo a mesma baseada na conhecida distribuição *Linux* de nome *Debian*. Existindo outras distribuições, esta foi a escolhida devido à sua estabilidade e instalada diretamente no cartão de memória da unidade. A versão instalada é baseada no *Debian* versão 7, sendo que à data era a última versão disponível.

A ligação à rede de acesso local (LAN) é feita através de uma *pen wireless*, da norma *IEEE 802.11 n*, podendo atingir velocidades até 300MBits/s, ligada a uma das quatro portas USB.

4.2.2. Unidades periféricas

As unidades periféricas escolhidas utilizam microcontroladores com arquiteturas e capacidades distintas entre si. As duas unidades escolhidas têm em comum o facto de serem placas de desenvolvimento. A escolha destes dispositivos deveu-se ao facto de facilitarem a construção do sistema global, dado terem já integrados todos os componentes essenciais ao funcionamento dos microcontroladores.

Foi escolhida, de entre outras opções, a placa de desenvolvimento *Arduino Mega 2560*, para servir de unidade de aquisição dos dados dos consumos energéticos, devido ao seu reduzido custo em relação à sua capacidade de realizar o processamento de sinais e à elevada quantidade de portas programáveis que dispõe, entre outros periféricos. A escolha desta placa teve também em conta a possível expansão futura do sistema.

O *Arduino Mega* (Figura 28) foi desenvolvido com o intuito de disponibilizar uma placa de desenvolvimento de baixo custo para estudantes, foi por isso definido que o projeto do mesmo é de livre utilização. Está equipado com um microcontrolador *Atmel ATmega2560* da família *AVR*.

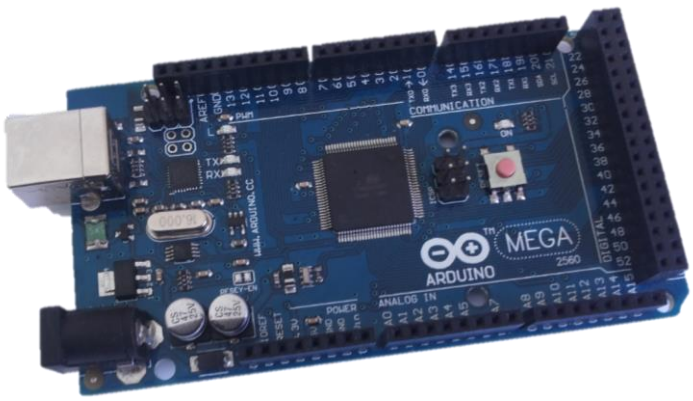


Figura 28: Microcontrolador *Arduino Mega 2560*.

Este microcontrolador possui uma velocidade de processador de 16MHz gerada a partir do gerador de sinal de relógio externo integrado na placa, uma memória EEPROM de 4 KB, uma memória volátil SRAM de 8 KB e uma memória *flash* de 256 KB [34].
Contém 54 entradas/saídas digitais programáveis, das quais 15 têm "Modelação por largura de pulso" (*PWM*) e 16 contêm um conversor analógico-digital. Estas portas operam a tensões entre os 0 e os 5V e possuem uma resistência de *pull-up* interna de 20 kΩ a 50 kΩ. Para além destas portas tem disponíveis 4 portas série (RX/TX), uma delas (pinos 0 e 1) ligada ao conversor *ATmega16U2* que faz a conversão do sinal USB para TTL.
A placa contém um regulador de tensão que funciona para tensões entre os 6 e os 20V para uma saída a 5V. Possui também portas para comunicação SPI e I2C (TWI).

O código enviado para o microcontrolador é desenvolvido, compilado e descarregado para o microcontrolador no *software* disponibilizado pela *Arduino,* que utiliza uma linguagem de programação C e C++. Este inclui uma série de livrarias que contêm funções de vários tipos que facilitam a programação do microcontrolador. As funções *setup()* e *loop()* têm de ser definidas no programa enviado para o microcontrolador para que este funcione, sendo que a função *setup()* é apenas corrida uma vez após o microcontrolador ser ligado enquanto a função *loop()* é corrida ininterruptamente.

Outras caraterísticas encontram-se em anexo no manual do microcontrolador.

A outra unidade periférica escolhida foi a placa de desenvolvimento *NodeMCU*. Esta placa tem como base o microcontrolador *Espressif ESP8266* que tem integrado um módulo Wi-Fi normalizado pela norma IEEE 802.11 b/g/n. Este microcontrolador está equipado com um processador de *Tensilica Xtensa LX106* de 32 bits regulado a 80MHz, uma memória RAM de 64 KBits para instruções, uma memória não volátil de 96KBits, e uma memória *flash* de 4MBits. Possui ainda 16 portas *GPIO*, portos SPI e I2C. Apenas contém um conversor analógico-digital.

Figura 29: Placa de desenvolvimento *NodeMCU* versão 2.

A programação deste microcontrolador pode ser efetuada no *software* disponibilizado pela *Arduino*, após a instalação das bibliotecas que contêm a informação dos registos internos do microcontrolador. O programa original desta placa de desenvolvimento é feita em linguagem *Lua Scripting*.

A placa inclui um regulador de tensão para 3,3V e um conversor de USB para TTL. Mais informação relativa a esta placa de desenvolvimento está disponível no anexo B.

4.2.3. Medição dos consumos

Tensão

O transformador utilizado para amostragem da tensão da rede foi o transformador para placa de circuito impresso da marca *Hahn* modelo *BV202157* de 230V AC para 9V AC, com uma potência aparente máxima de 0,5VA e uma corrente máxima no secundário de 0,55mA. Este transformador possui isolamento galvânico que protege a unidade de aquisição de dados contra sobretensões e ruído elétrico provenientes da rede e contra curtos circuitos. O valor da tensão em vazio é de 14,2V, devendo-se isto à tensão variar consoante a carga aplicada no secundário.

Figura 30: Transformador para circuito impresso *Hahn BV 202 0157* [35].

A tensão admitida nas portas de entrada do *Arduino* tem valores padrão entre os 0 e os 5V. Contudo o valor máximo pode ser ajustado através de uma referência interna ou externa ao microcontrolador por meio de programação. De modo a obter detalhe nas leituras efetuadas, foi necessário condicionar do sinal enviado para o conversor analógico-digital (*ADC*), neste caso programado para tensões entre os 0V e os 2,56V. Este condicionamento

foi feito através de um divisor de tensão resistivo, em que foi levado em conta a corrente máxima no secundário e a tensão máxima do conversor *ADC*.

Como se queria medir o valor eficaz da tensão, decidiu-se que em vez de se retificar a onda sinusoidal vinda do secundário do transformador se somaria uma tensão DC de *offset* tendo em conta o intervalo de tensão 0 - 2,56V, isto para subir o valor da tensão para valores apenas positivos. Esta tensão DC provém diretamente da saída de tensão do *Arduino* e tem o valor de 3,3 V, podendo a mesma conduzir uma corrente máxima de 50mA.

Na Figura 31 estão representadas as formas de onda aproximadas anteriormente referidas, em que está representado a cor azul a forma de onda da tensão no secundário do transformador, a vermelho a forma de onda da tensão no secundário após aplicação de um divisor de tensão, a laranja a tensão de corrente contínua proveniente do *Arduino* e a roxo a soma da tensão DC com a tensão AC dividida.

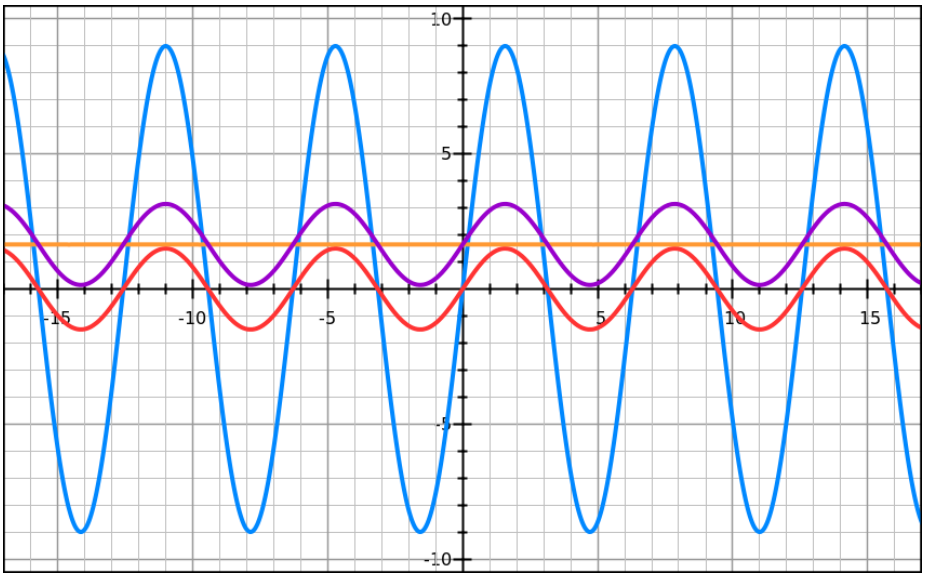


Figura 31: Sintetização do condicionamento do sinal de tensão.

Procedeu-se então ao dimensionamento e escolha dos componentes a utilizar, tendo em conta que se queria obter uma amostra da tensão vinda do transformador com um valor de aproximadamente 1V e que a soma a este valor da tensão de *offset* não poderia exceder os 2,56V. Para efeitos de dimensionamento, foi atribuído o valor de 10KΩ à resistência R1.

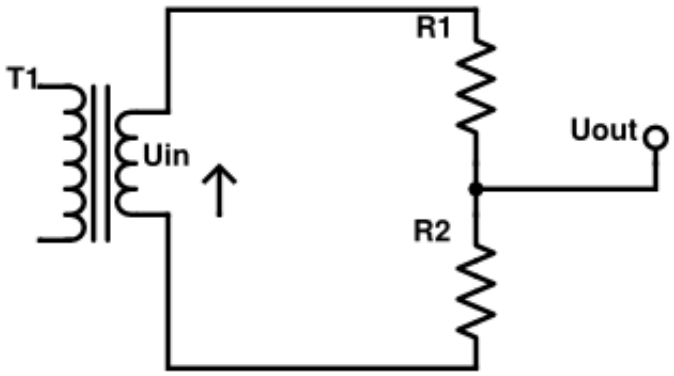


Figura 32:Divisor de tensão, em que Uin é a tensão de entrada e Uout é a tensão de saída.

$$Uout \cong 1 = \frac{R1}{R1 + R2} \times Uin \rightarrow 1 = \frac{10K\Omega}{10K\Omega + R2} \times 14{,}2 \rightarrow \ R2 \cong 130K\Omega \qquad (1)$$

Ajustou-se o valor da resistência R2 para o valor existente de 120KΩ. Para estes valores de resistência fica garantida que a carga aplicada ao transformador não fará ultrapassar o valor da corrente máxima que o mesmo pode fornecer. Sendo o valor da tensão de amostragem próximo de 1V decidiu-se reduzir o valor da tensão de *offset* para metade. Com a tensão à saída do *Arduino* 3,3V foi aplicado um divisor de tensão constituído por duas resistências de 470KΩ para descer o valor da tensão para metade e garantir um baixo consumo. O valor destas resistências foi atribuído tendo em conta que se desejam aplicar vários pontos de *offset*, pelo que é necessário manter a corrente baixa devido à saída de corrente do *Arduino* apenas conseguir fornecer até 50mA. O valor da tensão de *offset* é aproximadamente 1,667V, somando este ao valor da tensão secundário do transformador assegura-se que a tensão é sempre positiva.

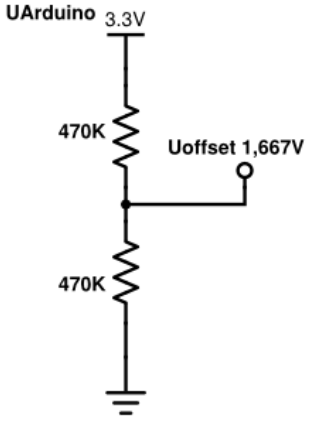


Figura 33:Divisor de tensão para a obtenção da tensão DC de *offset*.

Para limitar a corrente na porta de entrada do microcontrolador, colocou-se uma resistência com o valor de 1KΩ, ficando o circuito como o representado na Figura 34. Para filtrar o ruído gerado pela componente de corrente contínua da tensão de *offset*, foi adicionado um condensador de 10µF em paralelo com a resistência que liga o nó da tensão de *offset* à massa.

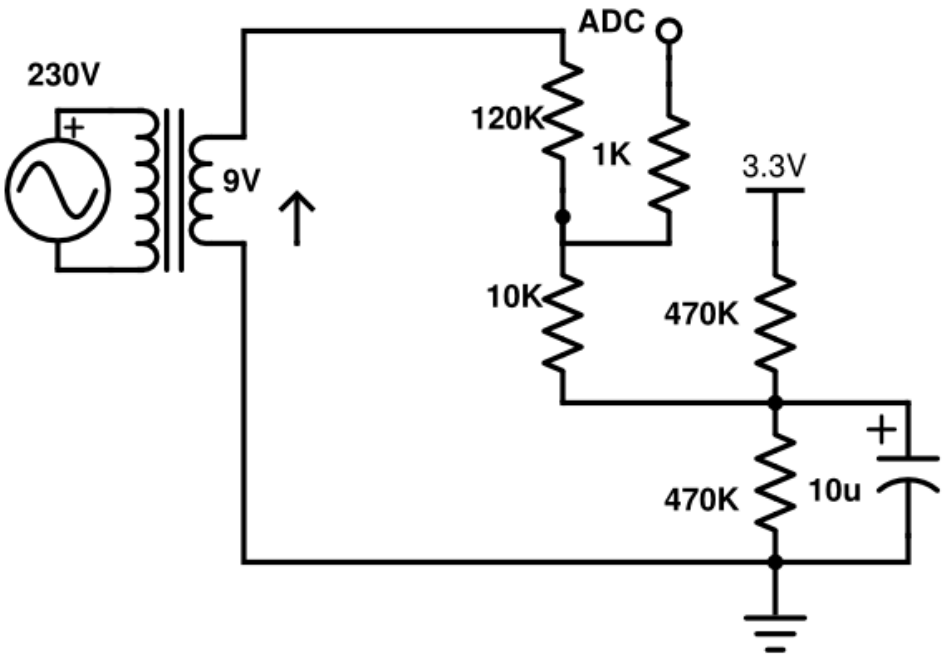


Figura 34: Circuito de amostragem da tensão, ligado a uma entrada com conversor ADC do *Arduino*.

Corrente

O transformador de corrente de núcleo separado *SCT-013-030* da *YHDC* foi o sensor utilizado para fazer a leitura da corrente. A escolha deste transformador teve em consideração o valor da corrente máxima que é possível medir com o mesmo. Este sensor consegue medir uma corrente máxima de 30A. Visto a habitação ter uma potência contratada de 3,45kVA (monofásica), ou seja, uma corrente máxima de 15A, este sensor consegue assegurar a medição da corrente. A escolha deste tipo de transformadores para instalações com potências superiores poderia recair num modelo com caraterísticas mais adequadas à corrente máxima da nova instalação.

Figura 35:Transformador de corrente *YHDC SCT-013-030*.

O sensor possui um circuito interno com uma resistência de *burden* de 62Ω que assegura a produção de uma tensão proporcional à corrente. Esta tensão só é proporcional enquanto não existe saturação da bobine, caso contrário o valor deixa de ser proporcional. Para impedir que a saturação ocorra para valores passiveis de serem medidos na instalação em estudo, o valor da resistência deve ser ajustado. A expressão utilizada para dimensionar o valor da resistência de *burden* tem em conta a corrente gerada no secundário do transformador.

$$Is = \frac{N2}{N1} \times Ipmax \tag{2}$$

$$Is = \frac{1}{1800} \times 30A = 0{,}0167mA \tag{3}$$

$$U = Rbarden \times I \rightarrow Rbarden = \frac{1V}{0{,}0167} \approx 60\Omega \tag{4}$$

O rácio de transformação deste sensor é de 1800:1 (*N2:N1*) e para os 30A máximos no primário do transformado é produzida uma tensão com o valor de 1V no secundário, o que equivale a uma corrente de 0,0167mA (equação 3). Logo a resistência de *burden* que assegura a proporcionalidade da tensão produzida em relação à corrente medida tem o valor de 60Ω. O valor real mais próximo é 60Ω, contudo este valor não é comum, pelo que foram utilizadas duas resistências de 120Ω em paralelo.

Foi aplicada uma tensão de *offset* igual à aplicada ao transformador de amostragem da tensão. Foi igualmente adicionada uma resistência de 1KΩ para proteção da porta de entrada do microcontrolador e um condensador para filtragem de possível ruido causado pela componente de corrente contínua. O circuito resultante está representado na Figura 36.

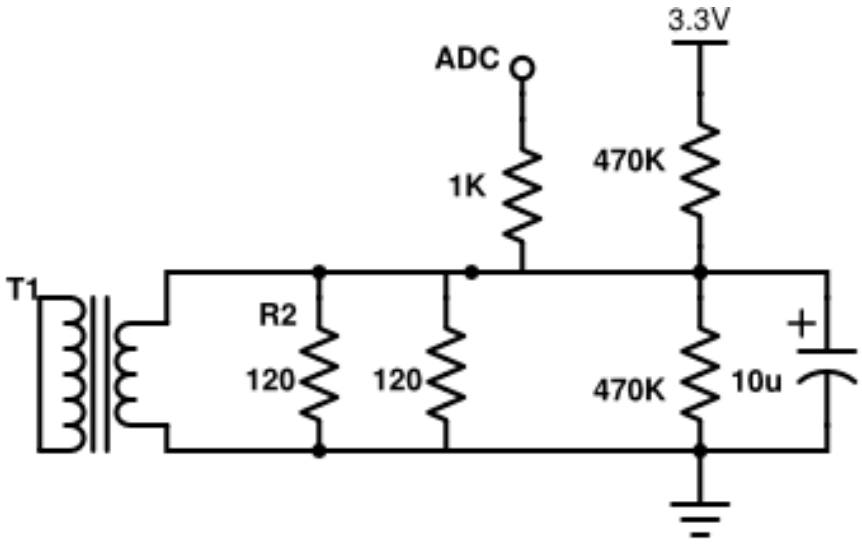


Figura 36: Circuito de medição de corrente, em que T1 representa o transformador de corrente.

Placa de aquisição de dados

Os circuitos de amostragem da tensão e corrente foram adicionados ao esquema da placa de captura de sinal. Esta placa foi projetada para poder ser utilizada como um *Shield* no *Arduino*, visto apenas se querer projetar um sistema para provar o conceito e não projetar um produto final. No entanto, a placa desenvolvida prevê uma utilização em futuros desenvolvimentos.

Desenhou-se então, através do *software Eagle*, o esquema da placa de captura de sinal (Figura 37) ao qual foi adicionado o circuito para amostragem da tensão e quatro circuitos de amostragem da corrente.

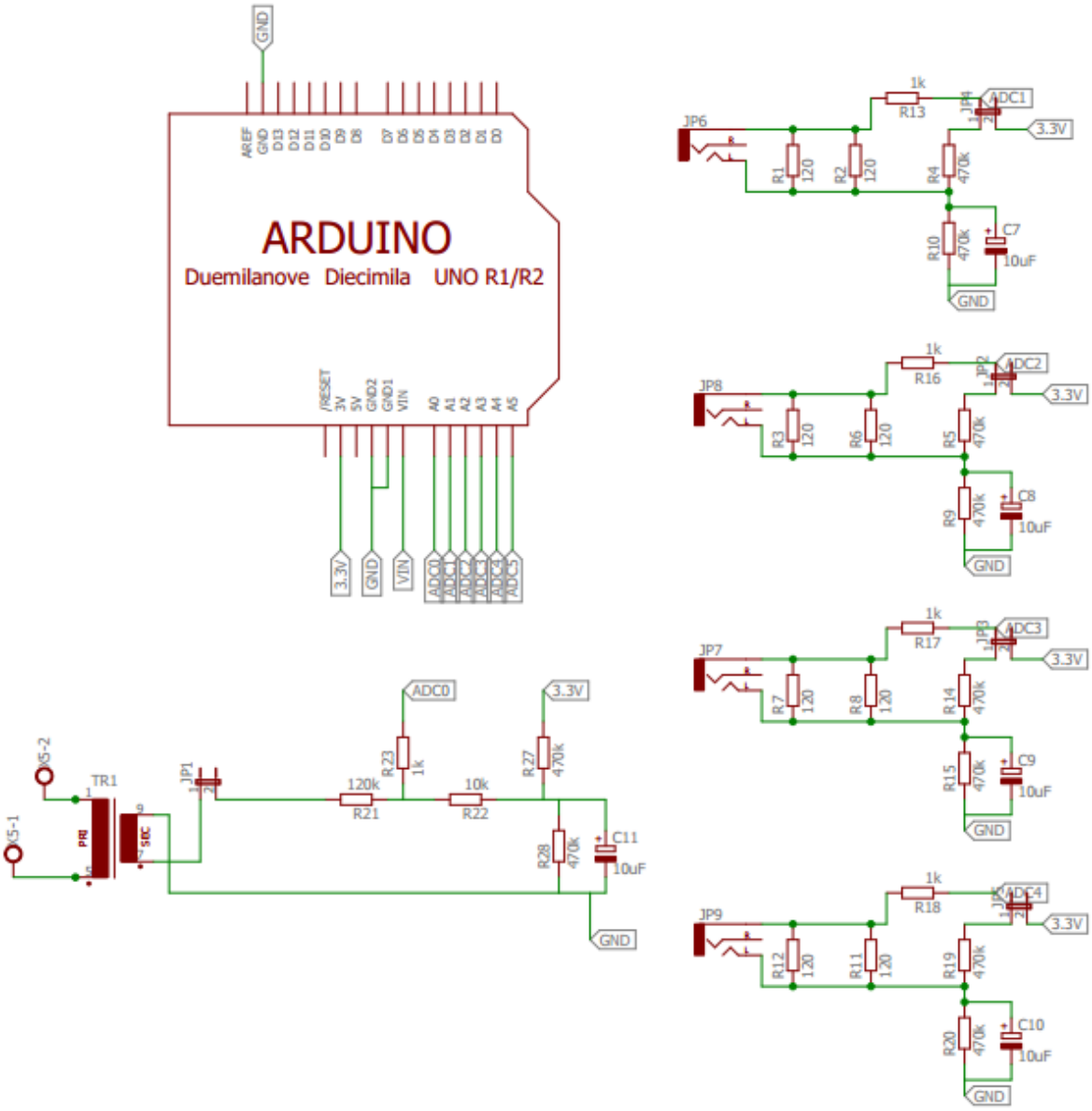


Figura 37: Esquema da placa de captura de sinal para medição das grandezas elétricas.

Durante o desenho do esquema, foram adicionadas quatro portas *stereo* de 2,5mm para ligação de transformadores de corrente. Foram inserido *Jumpers* nos circuitos de alimentação DC (tensão de *offset*) para permitir que estes sejam desligados. Um *Shield* anteriormente desenvolvido no IPT serviu de modelo para o desenho da placa de captura. O resultado do desenho em placa de circuito impresso está representado na Figura 38. O resultado final está representado na Figura 39.

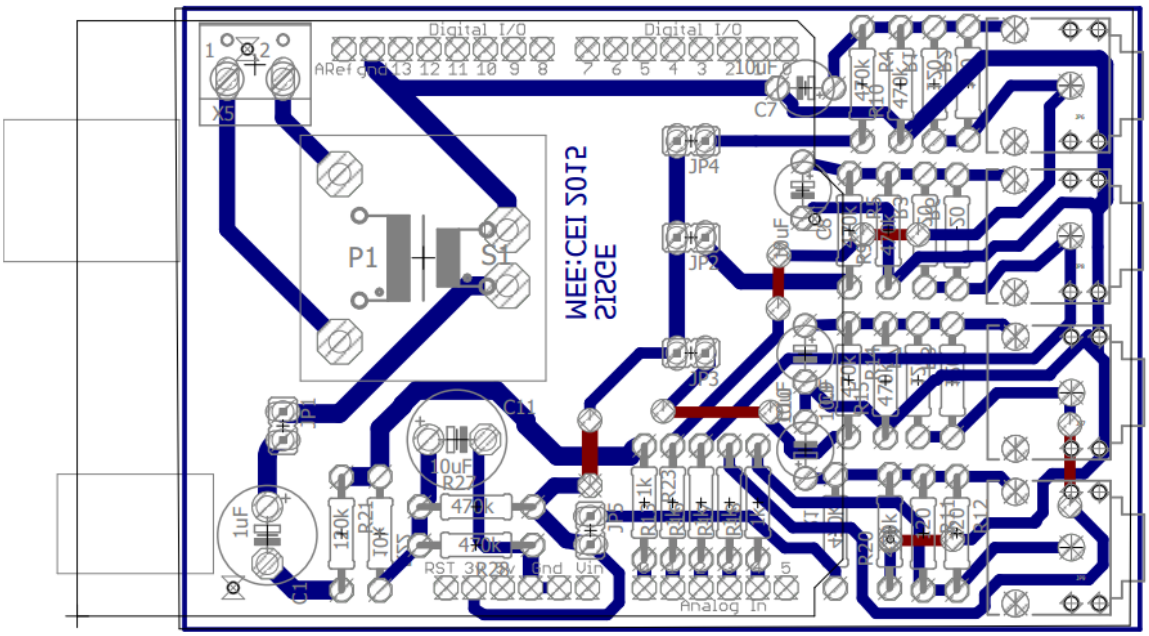


Figura 38: Desenho da placa de circuito impresso do *Shield* para medição de energia do SISGE.

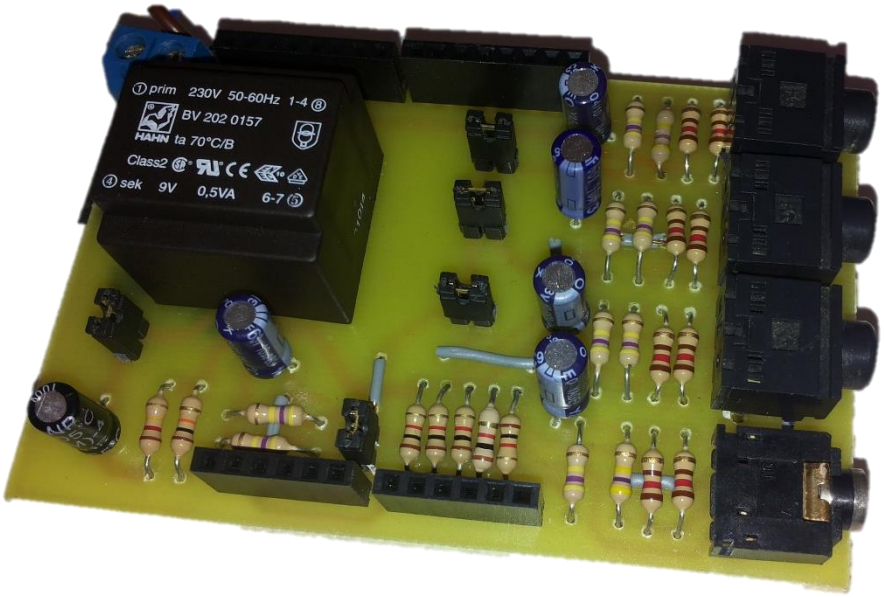


Figura 39: *Shield* para medição de energia do SISGE.

A combinação dos *Shield* de *Ethernet* e medição de energia deixam disponíveis para utilização todas as portas da placa de desenvolvimento *Arduino Mega*, com a exceção das portas *A0-A4*, *D4*, *D10* e porta *SPI*.

4.2.4. Segurança e controlo

Com o intuito de interligar o sistema de gestão de energia com um possível sistema de segurança foi utilizado a placa de desenvolvimento *NodeMCU* para servir de unidade periférica de aquisição de dados para a informação sobre o estado de sensores relativos à segurança. A esta placa foi ligado um sensor para deteção de movimento através de alterações na radiação de infravermelhos dos objetos, isto é, usando um sensor de infravermelhos passivo (*PIR*).

A escolha deste sensor deveu-se essencialmente ao seu baixo custo, baixa manutenção e baixo consumo energético. Este tipo de sensores é bastante utilizado devido à sua fiabilidade.

Os sensores piroelétricos são constituídos por duas partes iguais sensíveis a Infravermelhos [36]. Como representada na Figura 41, a deteção de movimento é feita através da diferença entre as duas partes do sensor. Esta diferença causada pela alteração na quantidade de radiação infravermelha captada por cada uma das partes. A alteração dos níveis de radiação é causada pelo aumento de temperatura devido à presença de pessoas ou animais no raio de alcance do sensor. O aumento de temperatura gera uma diferença positiva, enquanto a diminuição gera uma diferença negativa.

A lente dos sensores *PIR*, geralmente de plástico branco, serve para focar os feixes de radiação infravermelha no sensor piroelétrico, melhorando a deteção da diferença na radiação de infravermelhos ao aumentar o campo de visão do sensor. Esta lente serve também para proteção mecânica do sensor piroelétrico.

O sensor *PIR* utilizado foi o *HC-SR501* que contém um sensor piroelétrico LHI 778 integrado numa placa com circuito de conversão de sinal (ADC).

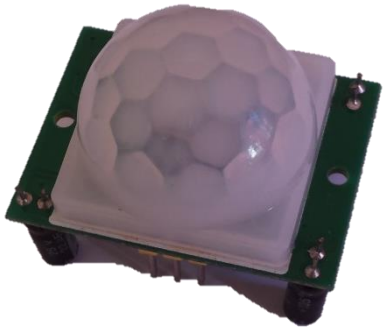


Figura 40: Sensor piroelétrico *HC-SR501*.

Este circuito pode ser alimentado por uma tensão entre os 3,3V e os 5V. A saída de informação é lógica, valor lógico alto para deteção de diferença de temperatura e baixo para ausência de diferença. O circuito integrado *BIS0001* é responsável pelo controlo do tempo a que a saída fica ligada, aquando da deteção de diferença na temperatura. O *datasheet* deste sensor encontra-se no DVD em anexo.

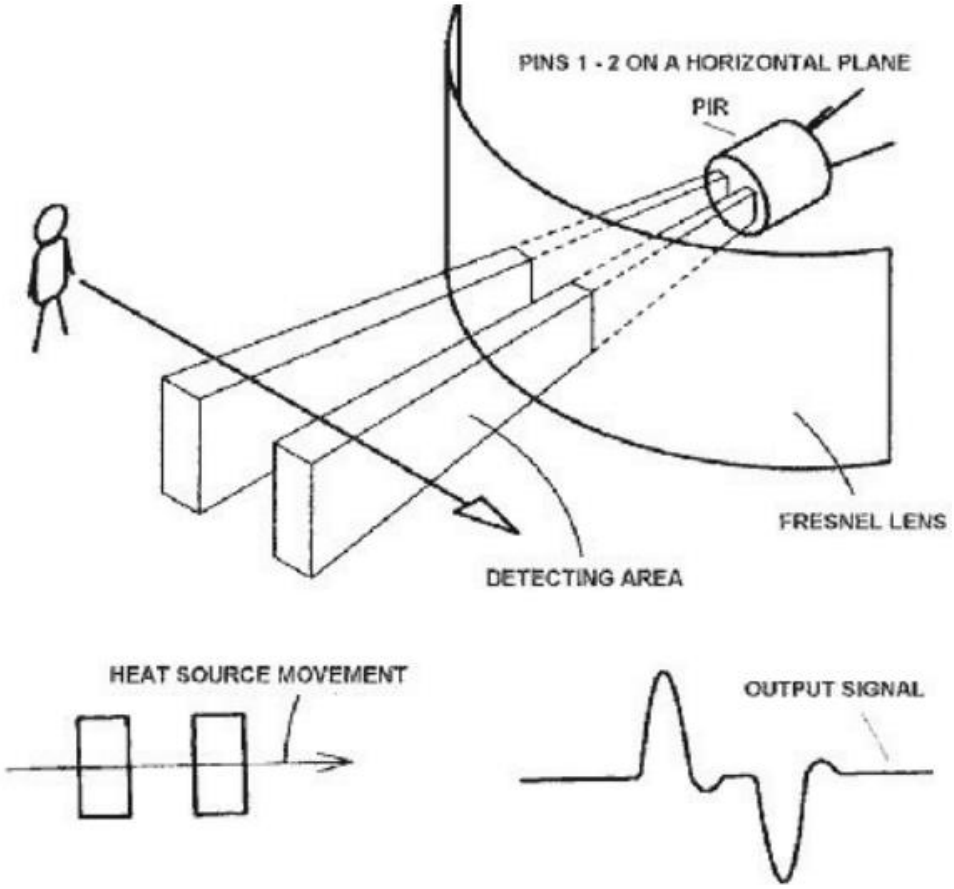


Figura 41: Funcionamento de um sensor PIR [36].

O esquema de ligação do sensor *PIR* à placa de desenvolvimento *NodeMCU* é bastante simples. Optou-se por não fabricar nenhuma placa, devido à simplicidade do circuito.

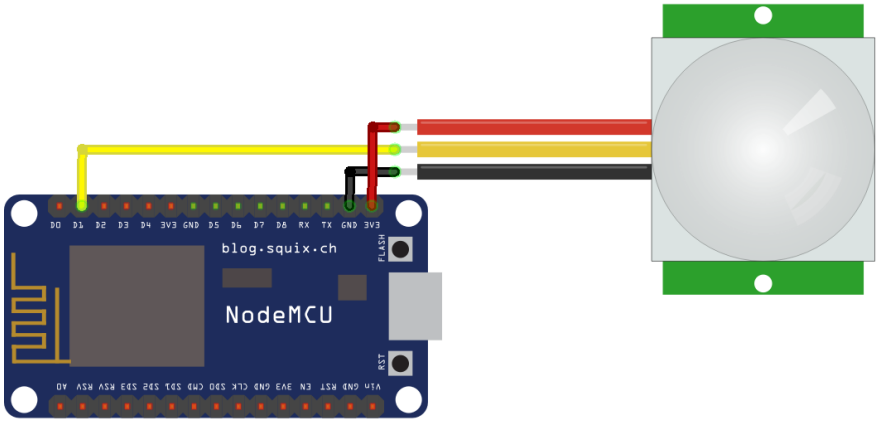


Figura 42: Esquema de ligação entre o sensor *PIR HC-SR501* e a placa de desenvolvimento *NodeMCU*.

A deteção de movimentos é feita através da deteção do acionamento do sensor *PIR* e a desativação é feita quando existe uma transação negativa, o código utilizado para detetar a ativação do sensor é o seguinte [37]:

```
void pirSensor(){
  if(digitalRead(pirPin) == 1){//Deteta a ativação do sensor
        if(lockLow){  //Alarme ativado
   lockLow = false;
   delay(50);
   }
   takeLowTime = true;
        }
        if(digitalRead(pirPin) == 0){     //Deteta a desativação do sensor
        if(takeLowTime){
                lowIn = millis();          //guarda o tempo da transição de "1" para "0"
                takeLowTime = false;
  }
        //Confirma a desativação do sensor
        if(!lockLow && millis() - lowIn > pause){
    lockLow = true;
    delay(50);
    }
 }
```

4.2.5. Meios de comunicação

Ethernet

Os condutores de pares de cobre instalados na habitação estão ligados a uma rede *Ethernet* gerida por um *Router*. Decidiu-se que esta tecnologia de transmissão seria uma solução adequada para a troca de informação entre as centrais periféricas e a unidade central de controlo. A facilidade de acesso a este tipo de rede, as possibilidades de fornecer acesso externo ao sistema e possibilitar a ligação de unidades periféricas através do ponto de acesso *Wi-Fi* do *Router* foram os principais fatores que levaram a esta escolha. Decidiu-se a não utilização de outros protocolos pois estes trariam um custo acrescido ao projeto.

O *Router* utilizado foi um *Technicolor TG784n v3*, contém quatro entradas *RJ45* para ligação de pares de cobre que podem atingir velocidades de transmissão até 100Mbits/s. Possui também um ponto de ligação *Wi-Fi* normalizado para IEEE 802.*11* b/g/n.

A ligação da placa *Arduino Mega* à rede *Ethernet* efetua-se através do acrescento de um *Shield* com uma placa de *Ethernet*. Este *Shield* contém um controlador de *Ethernet WIZnet W5001* (*datasheet* no DVD em anexo) que contém a pilha de endereços dos dispositivos ligados ao barramento. A ligação ao barramento é feita através de uma porta *RJ45*, podendo esta integrar um transformador de linha e permite a alimentação através do barramento, também conhecida como *Power over Ethernet* (*PoE*), sendo que para isto é necessário a adição de um módulo *PoE*. A placa possui um controlador de reinício que reinicia o circuito integrado *W51000* após a placa ser ligada para assegurar o seu correto funcionamento.

Figura 43: Shield Ethernet *Arduino*.

Esta placa contém um leitor de cartões de formato *microSD* que pode ser utilizado para a partilha de informação para a rede. Contudo, este leitor não será utilizado nesta fase do projeto.

A comunicação é feita por meio de protocolo de comunicação *SPI* através da porta *ICSP* do *Arduino Mega*. Este protocolo de comunicação série é usado em microcontroladores para comunicarem com periféricos a pequenas distâncias. Utiliza uma gestão *mestre/escravo*, geralmente o mestre é o microcontrolador e os escravos são os restantes elementos periféricos. A transmissão de informação é feita através de três linhas com sentidos únicos. A MISO (*Master in Slave out*) de envio de dados para o mestre, a MOSI (*Master out Slave in*) de envio de dados para o escravo e SCK que transmite um impulso de relógio que serve de meio de sincronização dos dados. A linha SS (*Slave Select*) ativa o escravo com o qual o mestre quer comunicar.

A comunicação é partilhada entre o circuito integrado *W51000* e o leitor de cartões, sendo a escolha feita através da ativação das portas 10 e 4 respetivamente. Como partilham a mesma porta *SPI*, a comunicação apenas pode ser feita à vez.

A placa dispõe de uma série de *LED* que indicam vários estados, como descrito na Tabela 4.

Tabela 4: Resumo dos indicadores luminosos do *Shield Ethernet*.

Nome	Funcionamento
PWR	Indica que a placa está ligada
LINK	Indica a existência de ligação à rede, pisca quando existe transmissão de informação
FULLD	Indica se a ligação Ethernet está a ser feita com suporte full Duplex (comunicação bidirecional simultânea)
100M	Indica que a ligação Ethernet está a ser feita a 100 Mbits/s ao invés de 10 Mbits/s
RX	Pisca quando informação é recebida
TX	Pisca quando informação é enviada
COLL	Pisca quando são detetadas colisões na rede

Todas as portas da placa de desenvolvimento *Arduino Mega* estão acessíveis no *Shield*, com exceção das portas 4, 8 e *SICSP*. A atribuição de um endereço de *IP* ao *Shiled Ethernet* é configurada através da programação do microcontrolador. Para configurar o

Shield Ethernet no *Arduino* foi utilizada a biblioteca *Ethernet.h,* disponível no *IDE Arduino* e de livre utilização. Esta biblioteca é responsável pelo controlo do circuito integrado *W51000*, em que foram utilizadas as seguintes funções:

```
byte mac[]    = {  0xDE, 0xEE, 0xBA, 0xFE, 0xFE, 0xED };
IPAddress ip(192, 168, 1, 11);
EthernetClient ethClient;
Ethernet.begin(mac, ip);
```

Em que, "mac" representa o endereço físico que se deseja atribuir ao dispositivo a registar na rede *Ethernet*. A função "IPAddress" define o endereço de *IP* que mais tarde será atribuído ao dispositivo. Um novo cliente de Ethernet é criado através da função "EthernetClient". Este cliente será chamado sempre que se queira enviar ou receber informação para a rede. O registo do dispositivo na rede é feito pela função "Ethernet", que regista o endereço "mac" e o seu respetivo endereço de *IP* na rede.

A ligação de *Wi-Fi* integrada no microcontrolador *ESP8266* permite a ligação direta do microcontrolador ao *Router*.

A biblioteca *ESP8266WiFi.h* foi utilizada para registar a placa *NodeMCU* na rede *Wi-Fi*. Foi utilizado o seguinte conjunto de funções presentes nesta biblioteca:

```
WiFiClient espClient;
WiFi.begin(ssid, password);
WiFi.status()
```

A função "WiFiClient" cria um novo cliente de *Wi-Fi* que será chamado sempre que se queira enviar ou receber informação para a rede. É utilizada a função "WiFi.begin()" para registar o dispositivo na rede identificada pelo seu *SSID*. Para verificar o estado da ligação é utilizada a função "WiFi.status()".

A unidade central de controlo do sistema, o computador *Raspberry Pi 2*, liga-se à rede Ethernet através da porta *RJ45* disponível no mesmo. A atribuição de um endereço de IP é automática e gerida pelo *Router*.

MQTT

O protocolo de mensagens escolhido para as comunicações entre as unidades periféricas e a unidade central foi o *Message Queue Telemetry Transport* (*MQTT*).

O *broker* escolhido para instalação na unidade central de controlo foi o *Mosquitto*. Este é um *broker* de livre utilização que implementa o protocolo *MQTT*.

Para instalar o servidor de *MQTT Mosquitto* no *Raspberry Pi 2* foram executados através de uma ligação *SSH* os seguintes comandos:

```
wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key
sudo apt-key add mosquitto-repo.gpg.key
cd /etc/apt/sources.list.d/
sudo wget http://repo.mosquitto.org/debian/mosquitto-wheezy.list
sudo apt-get update && sudo apt-get install mosquitto
```

Este conjunto de comandos adicionam a chave de verificação de veracidade e o reportório *Mosquitto* à lista de fontes de pacotes de instalação. De seguida é instalado o *broker Mosquitto*. Por padrão a porta do servidor é a 1883, esta manteve-se inalterada.

Para testar o correto funcionamento do *broker* foi efetuado um teste de subscrição e publicação para um tópico através da linha de comandos. Numa linha de comandos foi subscrito o tópico “sensors/temperature” com nível *QoS* igual a 1 ao utilizar o comando mosquitto_sub:

```
mosquitto_sub -t sensors/temperature -q 1
```

Numa segunda linha de comando foi enviada uma mensagem através do comando mosquitto_pub, com o valor “32” para o canal “sensors/temperature” e com nível *QoS* igual a 1:

```
mosquitto_pub -t sensors/temperature -m 32 -q 1
```

Resultando na receção por parte do subscritor de uma mensagem com o valor “32”.

Devido ao *Router Ethernet* restringir o acesso à rede a partir de fontes externas, optou-se por não configurar nenhum método de segurança adicional no *broker*. No entanto, este dispõe do comando *mosquitto_passwd* que permite a gestão de utilizadores e palavras-passe para autentificação dos clientes do servidor. Mais informação sobre o *broker Mosquitto* encontra-se no DVD em anexo.

A biblioteca *PubSubClient.h* foi utilizada para registar as placas *Arduino* e *NodeMCU* no servidor *Mosquitto*. As funções de publicação/subscrição de tópicos funcionam em conjunto com as funções das bibliotecas de ligação às redes *Ethernet* e *Wi-Fi*, é através dos seus clientes que as mensagens *MQTT* são enviadas. As seguintes funções da biblioteca *PubSubClient.h* foram utilizadas:

```
PubSubClient client(ethClient);
client.connect("Nome_Cliente")
client.subscribe("subscrever/tópico");
client.publish("publicar/tópico", mensagem);
client.setServer(server, 1883);
client.setCallback(callback);
client.connected()
```

Em que, a função "PubSubClient" cria um cliente *MQTT* através de uma ligação *Ethernet* ou *Wi-Fi*. A função "connect()" regista o cliente no *broker*. A informação sobre o endereço do *broker* é feita através da função "setServer()". A verificação do estado da ligação ao *broker* é feita pela função connected(). A subscrição de tópicos é feita recorrendo à função "subscribe()", enquanto a publicação de mensagens para tópicos é feita recorrendo à função "publish()". Quando uma mensagem é enviada para um tópico subscrito por um destes clientes o método "callback" é chamado para que a mensagem possa ser recebida. Este método é definido no cliente local pela função "setCallback".

4.2.6. Leitura das grandezas elétricas

A leitura da tensão da rede e da corrente fornecida é necessária para o cálculo aproximado do valor da potência ativa, potência aparente e fator de potência. Estes valores são obtidos através de técnicas de processamento de sinais, a uma frequência de amostragem de 4 kHz.

Para a leitura dos valores da tensão e corrente eficazes é necessário efetuar a remoção da componente DC introduzida no sinal para o tornar positivo. Após ser removida a componente DC, a medição dos sinais da tensão e da corrente é feita através de um ciclo de amostragem que permite o cálculo dos seus valores eficazes. Este ciclo depende do número de amostras efetuadas.

A remoção do *offset* DC introduzido nos sinais de tensão é feita através de um filtro digital passa alto. Este é um filtro de resposta a impulso infinito (*IIR*), a função de transferência do filtro é dada pela fórmula da equação 5 [39].

$$y[n] = 0{,}996 \times y[n-1] + 0{,}996 \times x[n] - 0{,}996 \times x[n-1] \quad (5)$$

Para uma entrada a degrau este filtro estabiliza ao atingir pelo menos 1% do valor de *offset* DC presente no sinal, o que ocorre em cerca de 1200 amostras [39]. Isto significa que depois de ligado o sistema, o filtro demora algum tempo até a componente DC do sinal ser removida.

O código utilizado na programação do microcontrolador *Arduino Mega* correspondente a este filtro é o seguinte:

```
//guarda ultimas amostras
lastU=sampleU;
lastI=sampleI;
//Recolhe amostras
sampleU = analogRead(inU);
sampleI = analogRead(inI0);
//guarda ultimos valores filtrados
lastFilterU = filterU;
lastFilterI = filterI;
//Aplicação do filtro digital.
filterU = 0.996 * (lastFilterU+sampleU-lastU);
filterI = 0.996 * (lastFilterI+sampleI-lastI);
```

A potência ativa de um dispositivo é responsável por produzir trabalho e é medida em Watt. Matematicamente define-se pela integração da tensão *u(t)* vezes a corrente *i(t)* de acordo com a fórmula da equação 6 [39].

$$P \equiv \frac{1}{T}\int_0^T u(t) \times i(t)dt \equiv U \times I \times cos\varphi \quad [\text{W}] \quad (6)$$

A tensão *U* e corrente *I* dizem respeito aos valores eficazes, também conhecidos como valores *Root Mean Square* (RMS). A potência pode ser calculada a partir da multiplicação desses valores eficazes pelo fator de potência (cosseno do ângulo fi). O ângulo fi diz respeito ao ângulo de desfasagem entre a onda da tensão e da corrente. Contudo, esta fórmula apenas é válida para ondas com formas sinusoidais ideais.

As formas de onda sinusoidais da rede (tensão e corrente) sofrem regularmente de distorção causada pela soma de harmónicos de alta frequência, devido a isto o fator de potência deixa de ser dado pelo cosseno de fi.

Visto que a medição da tensão e da corrente eficazes é realizada em simultâneo de forma discreta, a potência ativa é dada pela equação 7.

$$P = \frac{1}{N}\int_{n=0}^{N-1} U(n) \times I(n) \quad [\mathrm{W}] \tag{7}$$

Os resultados obtidos através da equação 7 seriam bastante aproximados aos valores obtidos através dos valores eficazes da tensão e corrente. A potência é assim calculada num ciclo de amostragem com N amostras, resultando do cálculo o valor médio obtido pelas N amostras. Este cálculo é válido para curvas sinusoidais puras ou com forma de onda pouco distorcida.

Tendo como base os pressupostos anteriores, o cálculo da tensão e corrente eficazes é dado pelas equações 8 e 9.

$$U = \sqrt{\frac{\sum_{n=0}^{N-1} U^2(n)}{N}} \quad [\mathrm{V}] \tag{8}$$

$$I = \sqrt{\frac{\sum_{n=0}^{N-1} I^2(n)}{N}} \quad [\mathrm{A}] \tag{9}$$

O código utilizado para calcular a tensão, corrente e a potência ativa é o seguinte:

```
//Calibração de fase
calibU = lastFilterU + calibFase * (filterU - lastFilterU);
//Tensão RMS
sqrtU= calibU * calibU;
sumV += sqrtU;
//Corrente RMS
sqrtI = filterI * filterI;
sumI += sqrtI;
//Potência instantanea
instP = abs(calibU * filterI);
sumP += instP;
//Cálculo dos valores aproximados
Vrms = VCAL*sqrt(sumV / samples);
Irms = ICAL*sqrt(sumI / samples);
activPower = VCAL*ICAL*sumP / samples;
```

Os valores de tensão e corrente eficazes reais são obtidos quando multiplicados os valores eficazes das amostras por um fator de calibração. Esse fator é obtido a partir da divisão do número de espiras do transformador pela resistência de carga (*Rburden*) vezes a resolução do ADC e vezes um fator de calibração apurado a partir de testes efetuados. Devido à resolução do ADC presente no microcontrolador, a corrente mínima possível de ser medida pelo sistema ronda os 75mA.

A potência aparente *S* corresponde normalmente à soma vetorial da potência ativa e reativa. Pode também ser dada pela equação 10, que expressa a multiplicação da tensão pela corrente eficazes.

$$S = \sqrt{\frac{\sum_{n=0}^{N-1} U^2(n)}{N}} \times \sqrt{\frac{\sum_{n=0}^{N-1} I^2(n)}{N}} = U \times I \quad \text{[VA]} \tag{10}$$

A potência aparente é calculada recorrendo ao seguinte código:

```
apparentPower = Vrms * Irms;
```

Calculado o valor da potência ativa, é possível chegarmos ao valor da energia consumida (num dado tempo) através da integração para num dado intervalo de tempo (equação 11).

$$E = \int_0^T p(t)\, dt = P \times T \qquad \text{[Wh]} \tag{11}$$

A energia é calculada através do seguinte código:

```
//Calcula o tempo deste a ultima medição
endTime = startTime;
startTime = millis();
timeLastMeasure = startTime - endTime;
//Calcula a energia consumida no intervalo e adiciona à soma dos kWh
 energyTotal = energyTotal + ((activPower/1000.0) * 1.0/3600.0 *
(timeLastMeasure/1000.0));
```

Devido à presença de distorções nas formas de onda da tensão e corrente, o fator de potência é calculado através da divisão da potência ativa pela potência aparente (equação 12).

$$FP = \frac{P}{S} \tag{12}$$

O Fator de potência é calculado pelo seguinte código:

```
if(apparentPower!= activPower){
        powerFactor = activPower / apparentPower;
}else{
        powerFactor=1.0;
}
```

O cálculo da frequência é feito através da deteção da passagem dos valores por zero, e é obtida pelo seguinte código:

```
//Frequência
if (n==0) LastZero = micros();
        //Verifica se está a passar por zero
        if (lastFilterU < 0 && filterU >= 0 && n>1)
        {
        //Periodo da onda da tensão
        periodU = micros() - LastZero;
        //Filtra leituras erradas
        if (periodU > (sinPeriod-filterWidth) && periodU<(sinPeriod+filterWidth))
        {
        periodUSum += periodU;
        periodUCount++;
        }
        LastZero = micros();
}
//Frequência
freq = (1000000.0 * periodUCount) / periodUSum;
periodUSum=0;
 periodUCount=0;
```

4.2.7. Manipulação e armazenamento da informação

De modo a manipular a informação enviada pelas unidades periféricas para o *broker MQTT*, a aplicação *Node-Red* foi utilizada para ligar as várias dimensões do sistema (unidades periféricas, base de dados e aplicação).

O *Node-Red* é uma aplicação de livre utilização para criação e implementação de aplicações. Foi criada com o objetivo principal de facilitar a interação entre os vários elementos da *Internet of Things*, ao facilitar a ligação entre dispositivos físicos e aplicações.

Esta aplicação é acedida através de página *web e* contém nós (*nodes*) que podem, de uma maneira simples, ser ligados através de um método de programação visual *Drag and Drop*. Estão organizados por fluxos. Cada nó dispõe de diferentes funções, desde o simples nó de *debug* que serve para testar o fluxo, até nós para comunicação direta com as GPIO do *Raspberry Pi*.

A escolha desta aplicação deve-se sobretudo à simplicidade que apresenta para realizar tarefas complexas através de lógica simples. Devido à sua programação em modo visual, os fluxos de nós criados tornam-se simples de interpretar.

Foi inicialmente concebido para trabalhar exclusivamente com o protocolo *MQTT*. No entanto, devido à sua capacidade de interação com os vários elementos ligados ao sistema (hardware/software), foram desenvolvidas novas funcionalidades.

O *Node-Red* corre sobre a aplicação base *Node.js*. Esta utiliza uma linguagem de programação *Javascript* que corre sobre uma "Máquina Virtual". Para instalar o *Node-Red* foram executados os seguintes comandos através de uma ligação *SSH*:

```
curl -sL https://deb.nodesource.com/setup_0.12 | sudo bash -
sudo apt-get install -y build-essential python-dev python-rpi.gpio nodejs
sudo npm install -g --unsafe-perm  node-red
sudo npm install -g node-red-node-mysql
```

Este conjunto de comandos descarrega e instala a aplicação *Node-Red*, assim como as suas dependências. Para iniciar o *Node-Red* o seguinte comando foi executado:

```
node-red-pi --max-old-space-size=128
```

Ao aceder à pagina web do *Node-Red*, através do endereço local http://raspberrypi:1880/, é apresentada uma página como demonstrado na Figura 44.

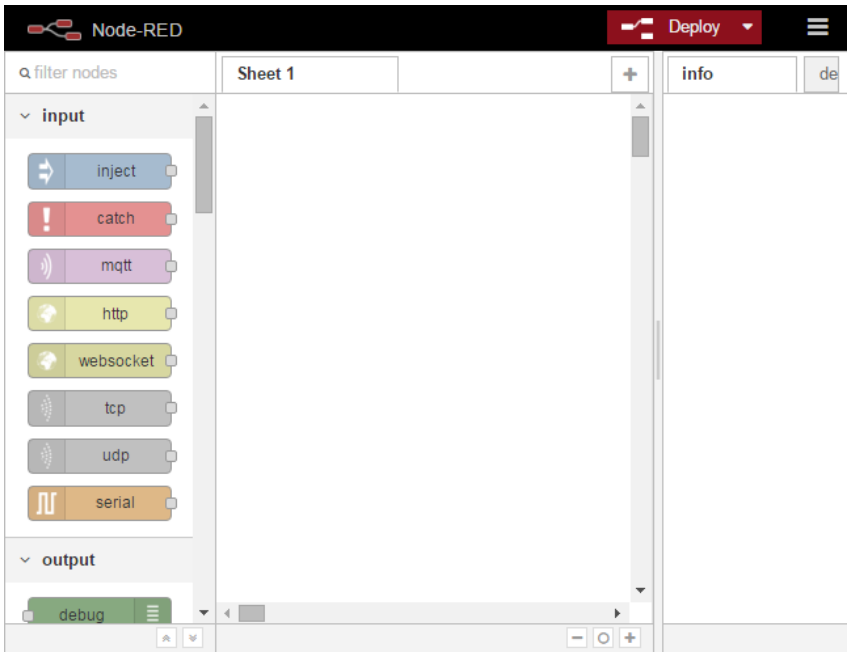


Figura 44:Página inicial do *Node-Red* com lista de nós à esquerda, área de desenho do fluxo no centro e painel de informação à direita.

Nos vários fluxos desenvolvidos foram utilizados os seguintes nós:

- http in: recebe pedidos *REST* do tipo *POST* ou *GET* enviados para o endereço definido (*e.g.* http://raspberrypi:1880/services/rest/getTemperature recebe pedido para o envio de informação da temperatura)
- mqtt in: subscreve um determinado tópico de modo a receber as mensagens para ele enviadas, por outras palavras, cria um cliente *MQTT*.
- inject: injeta manualmente ou através de temporizador uma mensagem no fluxo.
- twitter in: Recebe mensagens enviadas para as mensagens privadas do *twitter*.
- mysql: Cria uma ligação a uma base de dados *mysql* e executa comandos na mesma. Recebe a resposta aos comandos executados.
- function: Permite que a mensagem que circula no fluxo seja alterada consoante as necessidades.
- http out: Envia respostas aos pedidos efetuados por *REST* através do nó http in.
- mqtt out: Efetua publicações nos tópicos determinados.
- tweeter out: Envia mensagens para a rede social *Twitter*.
- email: Envia mensagens de correio eletrónico para o endereço determinado.
- debug: apresenta no painel *debug* da página do *Node-Red* a mensagem que é enviada do fluxo para este nó.

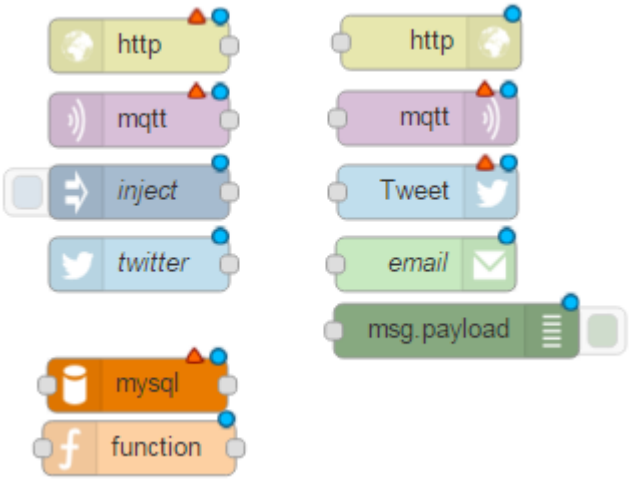


Figura 45:Nós *Node-Red* Utilizados.

Depois de criados os fluxos, estes são colocados em funcionamento após o *deploy* ser concluído. O *deploy* é feito através do botão "Deploy" presente na página do *Node-Red*.

Floram desenvolvidos três fluxos no *Node-Red*, um fluxo para receção de dados vindos das unidades periféricas, um fluxo para envio de informação para a interface do utilizador e um fluxo para alteração do estado de sensores e atuadores.

O fluxo relativo à inserção da informação vinda das unidades periféricas está representado na Figura 46, em que, o nó sisge/energy é do tipo *mqtt in* e é um cliente subscritor do tópico para onde são enviadas as mensagens que contêm os dados dos consumos. O nó sisge/security é o cliente *mqtt* subscritor do tópico para onde é enviada uma mensagem em caso de deteção de movimento. O nó Create Inser Query é do tipo *function* e constrói, com base na informação recebida no nó *mqtt* in, o código para inserção e atualização de tabelas na base de dados. O código é executado no nó do tipo *mysql* que se liga à base de dados. O nó *Prepare Email* constrói uma mensagem a partir da informação recebida no nó *mqtt in*. Essa mensagem é depois enviada para um endereço de correio eletrónico. O nó *Save to DB* cria o código *SQL* para inserção do estado da deteção de movimentos na base de dados.

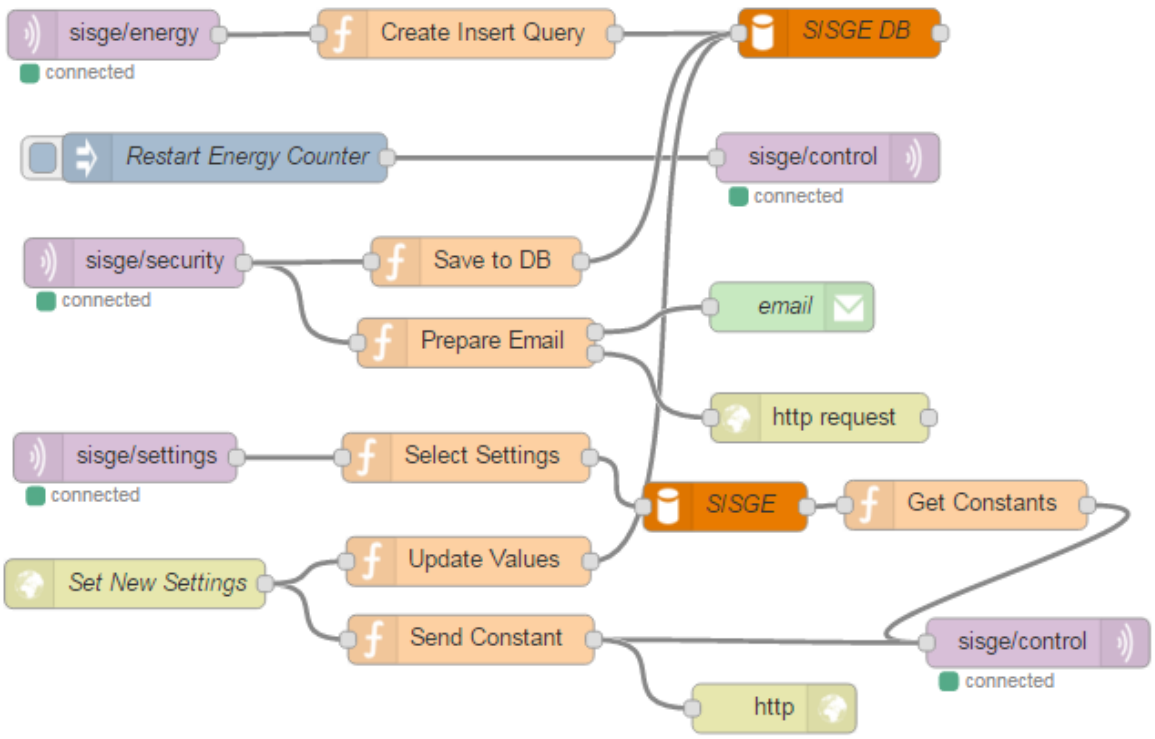


Figura 46:Fluxo de receção dos consumos, mensagens de alarme e pedido de definições.

O fluxo relativo ao envio de informação para a interface do utilizador está representado na figura 47. Os nós com nome começado por *get* são do tipo *http in* e implementam entradas de serviços de protocolo *REST*. Consoante o serviço invocado, é criada uma *query SQL* nos nós do tipo *function* que é posteriormente executada na base de dados. A resposta obtida da base de dados é convertida em linguagem *Json* nos nós *toArry* e enviada ao requerente do serviço REST pelo nó do tipo *http* out de nome *http*.

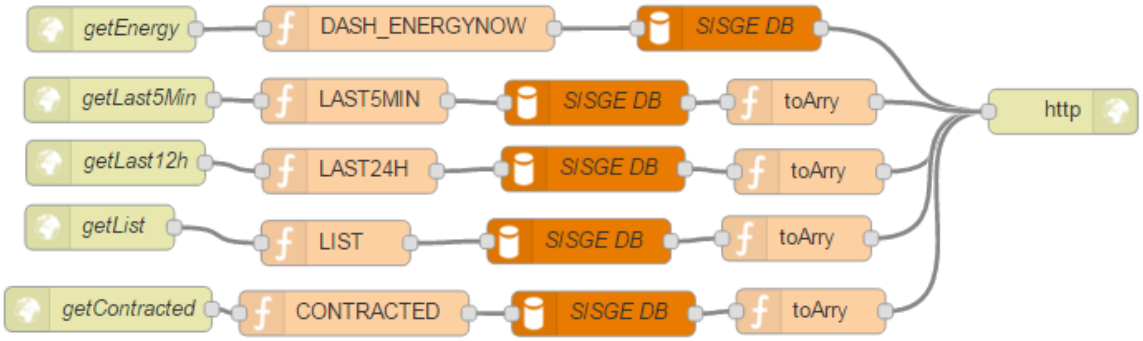


Figura 47: Fluxo de envio de informação para a interface do utilizador.

O fluxo de alteração em base de dados do estado de atuadores e a comunicação desse estado à interface do utilizador, está representado na figura 48. Os nós *getSensorStatus* e *sendCommand* são do tipo *http in* e recebem respetivamente os pedidos de envio e alteração do estado dos atuadores ligados às unidades periféricas. No caso de pedidos de envido é gerada uma *query SQL* no nó *Select Status* que é executada em base de dados e posteriormente transformada em linguagem *Json* e enviada para o requerente no nó *Http Response*. No caso dos pedidos de alteração, estes são tratados pelos nós *Create* Statement e enviados para as unidades periféricas através do tópico sisge/control, alterados em base de dados e comunicados à interface do utilizador pelo nó *Update Widget*.

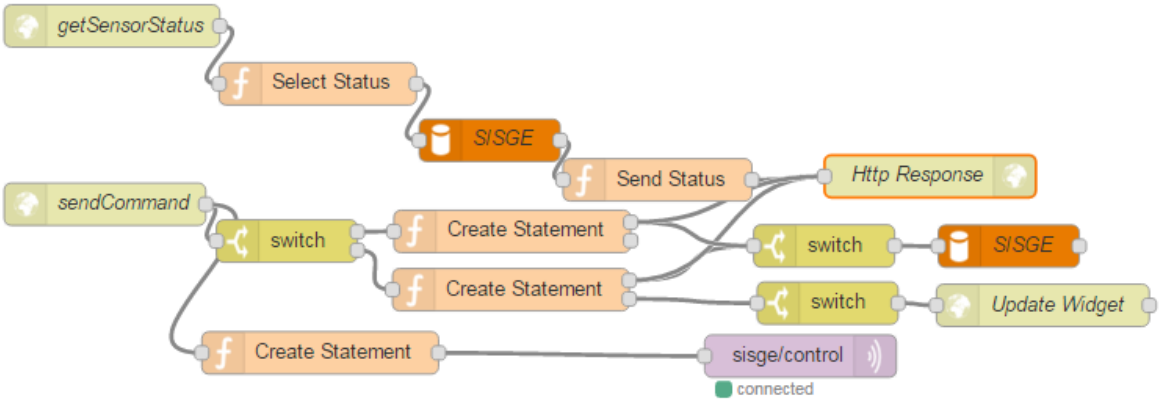


Figura 48: Fluxo de alteração do estado de atuadores e sensores e envio do estado para a interface do utilizador.

A base de dados escolhida para este projeto foi a *MySQL*. Esta escolha deveu-se à capacidade que este tipo de base de dados tem e às funções nativas de interação com protocolos *Web*. Foram criadas várias tabelas onde são guardadas as informações vindas dos sensores. Essas informações são enriquecidas com dados que localizam a informação no tempo. As tabelas criadas estão representadas na Figura 49.

A tabela "CYCLE" contém a informação relativa aos ciclos horários de faturação para Portugal Continental, retirados da página da Entidade Reguladora dos Serviços Energéticos [40]. Estes ciclos têm em conta as diferenças devidas ao período de horário de verão. Os preços relativos aos serviços de prestação de energia são guardados na tabela "PRICES" e têm em conta as várias potências, ciclos e tarifas.

A Tabela "ENERGYNOW" guarda a última informação dos consumos enviada pela unidade de aquisição de dados. Toda a informação dos consumos é guardada na tabela "ENERGY_HISTORY". Na tabela "STATUS" são guardados valores correspondentes ao estado dos sensores e atuadores. As configurações da aplicação são guardadas na tabela "OPTIONS".

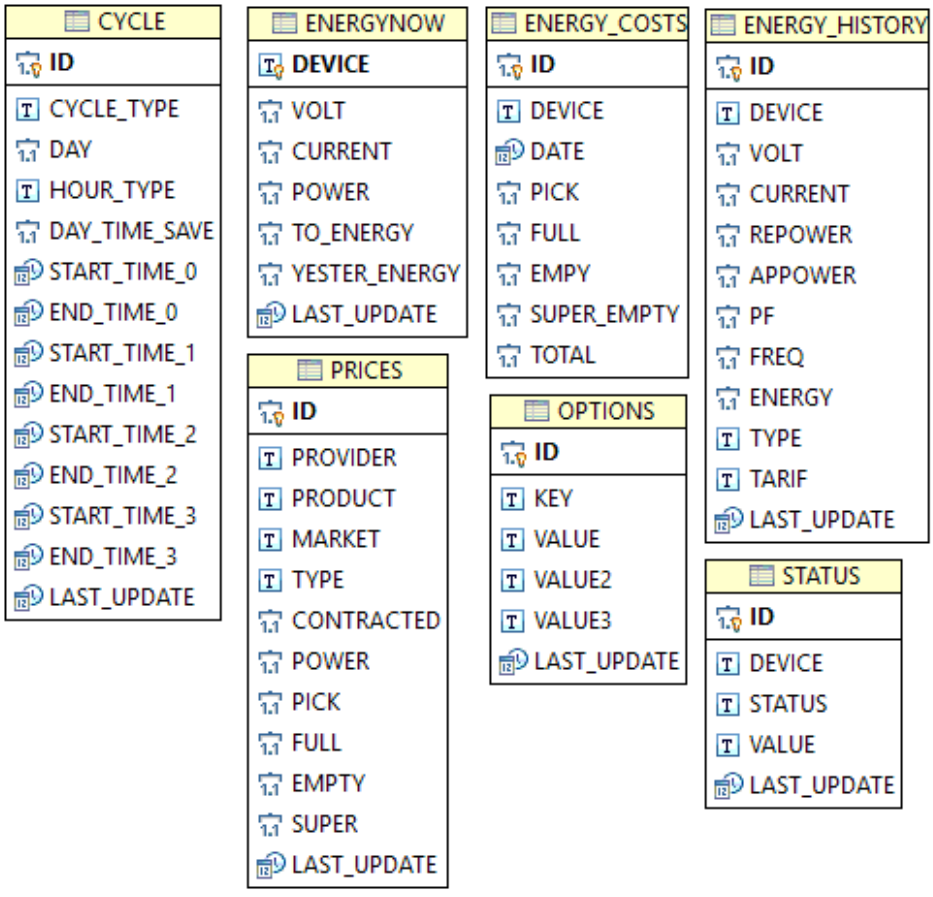


Figura 49:Tabelas criadas para armazenamento de histórico e apoio à aplicação SISGE.

Foram criadas diversas vistas para a base de dados, as mesmas podem ser consultadas no DVD e anexo a este documento. As vistas "LIVE_DASH", "LAST5MIN", "LAST24H" e "LIST" executam a seleção de informação dispersa em várias tabelas e reúnem-na numa só vista. Foram ainda criados eventos e *triggers* de base de dados para criação dos registos diários de consumo, atualização dos custos diários de consumo e atualização dos principais indicadores utilizados na interface do utilizador.

4.2.8. Interface do utilizador

Para a interface do utilizador foi utiliza uma aplicação cuja fonte é de utilização livre, denominada *Dashing*. Esta aplicação foi escolhida devido às subaplicações disponíveis, para além de ser de utilização livre, tendo ainda a seu favor o aspeto visual. Várias outras aplicações foram testadas, como *OpenHab, Domoticz ou HomeGenie.* No entanto optou-se por não utilizar nenhuma dessas e criar uma nova baseada numa estrutura já existente, visto querer-se mostrar informação de consumos e respetivos custos, algo para o qual nenhuma aplicação estava preparada.

Aos elementos constituintes de cada componente das páginas do *Dashing*, dá-se o nome de *widget*. Cada *widget* corresponde a um determinado tipo de interface *e.g.* gráfico, medidor ou caixa de texto.

O *Dashing* corre sobre a aplicação *Sinatra.* Esta é baseada em linguagem de programação *Ruby*. A programação desta aplicação é feita em várias linguagens, específicas para páginas *web*, como *Javascript*, *Ruby* ou *html*. A programação das páginas foi efetuada em tês locais principais:

- ficheiro *home.erb* que contem todo o código *html* que invoca os *widgets* de todas as páginas. É neste ficheiro que se decidem quais e como são dispostos os *widgets*, assim como a sua dimensão.
- *Scripts* em linguagem *Ruby*. Estes *scripts* executam serviços ao iniciar da aplicação ou em rotinas programadas.
- Configurações dos *widgtes* que contém três ficheiros:
    - Script *coffeescript* que executa invocações *REST* e define valores dos *widgets* ,entre outras possibilidades;
    - Ficheiro *html* que contém a estrutura do *widget*;

- Ficheiro *scss* que contém a programação *javascript* de configuração dos estilos do *widget*.

A aplicação *Dashing* executa sobre um serviço *Ruby* que já se encontrava instalado no *Raspberry Pi*.

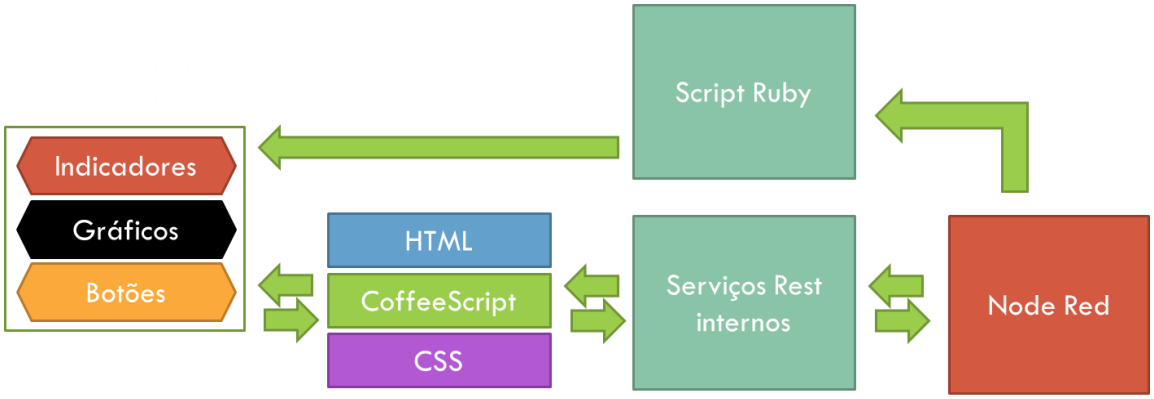


Figura 50: Diagrama dos vários componentes da interface utilizador.

A instalação do *Dashing* no *Raspberry Pi* foi feita ao ser executado o seguinte comando:

```
sudo gem install dashing
```

A criação de um novo projeto *Dashing* foi de seguida efetuada:

```
sudo dashing new sisge
cd sisge
bundle
```

A aplicação é iniciada ao ser executado o seguinte comando:

```
dashing start -d
```

Depois de iniciar, a interface gráfica fica disponível no endereço local http://raspberrypi:3030/ ou no endereço externo http://sisge.zapto.org. O acesso à interface requer a autentificação seguinte:

Utilizador: “admin”

Palavra-passe: “admin”

Os valores apresentados nas figuras seguintes são reais com exceção dos valores da energia produzida que são simulados.

A página principal (*home*) da interface (Figura 51) contém os botões de acesso às páginas Energia, Segurança, Controlo e Sobre. Estão também presentes nesta página as identificações do IPT e do VITA.IPT e um relógio.

Figura 51:Página principal da interface.

Em todas as subpáginas existe um botão de retorno à página inicial (Figura 53) e outro que reinicia a página (*refresh*) (Figura 54). Algumas das subpáginas contém também um botão para voltar à subpágina anterior (Figura 55).

A interação entre a interface gráfica e os equipamentos é feita nas subpáginas Energia, Segurança e Controlo. A página Sobre contém informação relativa ao projeto (Figura 52).

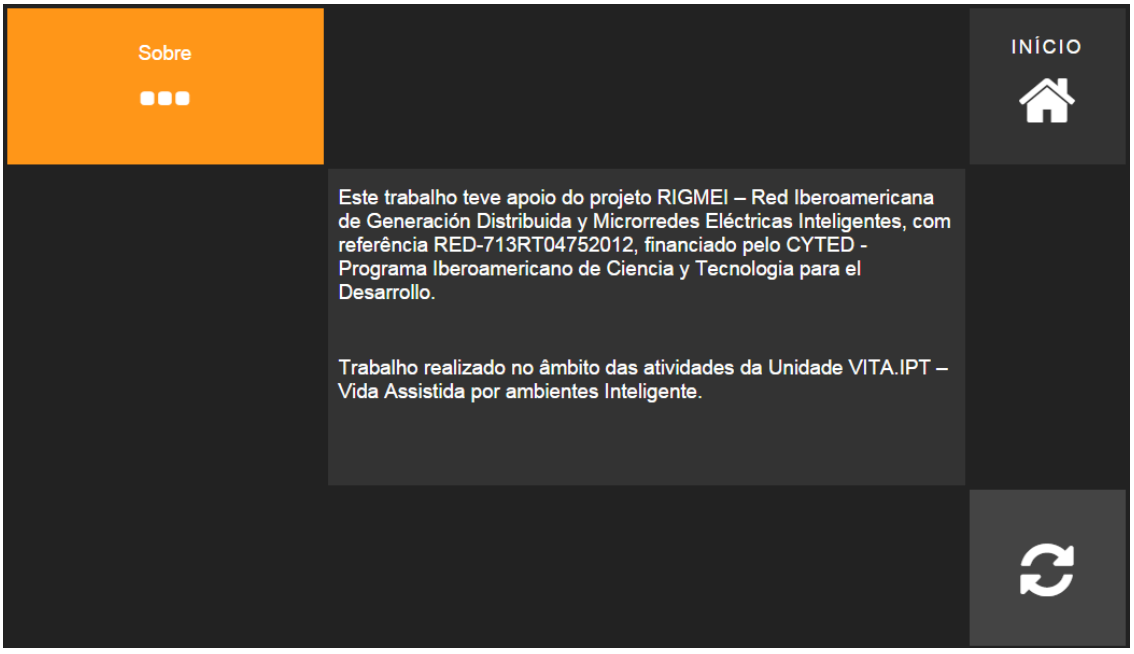


Figura 52:Página “Sobre”, contém descrição sobre os apoios ao SISGE.

Figura 53: Botão de retorno à página inicial.

Figura 54: Botão de *refresh*.

Figura 55: Botão voltar.

A página "Energia" apresenta em tempo real os valores da tensão, corrente, potência ativa, energia consumida no dia, tipo de hora de faturação e preço atual da energia. Contém também dois gráficos em que um deles compara os consumos com a produção efetuada e o outro apresenta a potência ativa medida nos últimos 5 minutos.

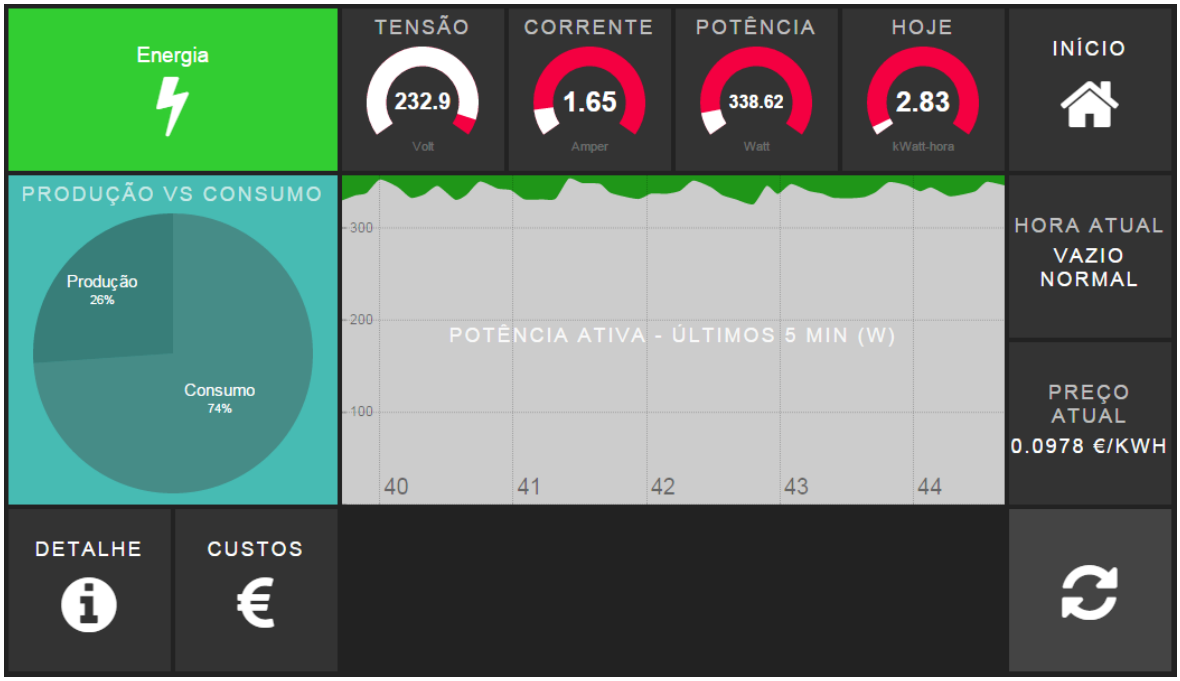


Figura 56:Página "Energia", contém os principais indicadores do consumo.

Os *widget* presentes na página "Energia" têm as seguintes caraterísticas:

- Tensão: valor da tensão em número e em indicador gráfico com escala entre os 200V e os 255 V.
- Corrente: valor da corrente em número e em indicador gráfico com escala entre os 0A e os 15A. Corresponde ao geral da habitação.
- Potência: valor da potência ativa em número e em indicador gráfico com escala entre os 0W e os 3500W. Corresponde ao geral da habitação.
- Hora Atual: atual tipo de hora de faturação. Este valor é baseado no ciclo horário definido na base de dados.
- Preço Atual: atual preço da energia de faturação. Este valor é baseado no ciclo horário definido na base de dados.
- Produção Vs Consumo: percentagem comparativa da produção (renováveis) e do consumo diário.
- Potência Ativa – Últimos 5 min: potência ao longo dos últimos cinco minutos, com um intervalo entre amostras de 5 segundos.

Na página "Energia" estão disponíveis mais dois botões correspondentes às subpáginas "Detalhe" e "Custos".

Na página “Detalhe” (Figura 57) é possível visualizar quais os últimos valores da potência e energia enviados da unidade periférica. Esta página contém um gráfico que representa a potência ativa nas últimas doze horas.

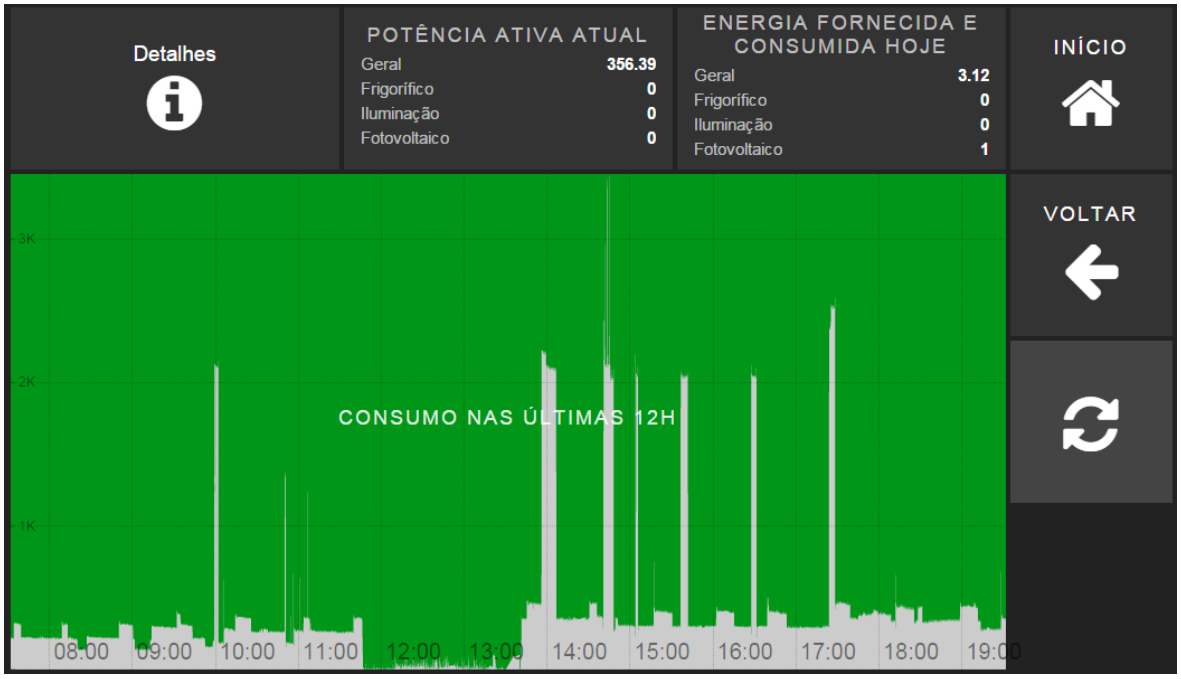


Figura 57: Página “Detalhe”, pertencente à página “Energia”.

A página “Custos” (Figura 58) contém os *widgets* com as seguintes caraterísticas:

- Serviço: serviço de energia contratualizado.
- Hoje: custo energético do dia, tendo por base os consumos efetuados e o serviço contratado.
- Esta Semana: custo energético dos últimos 7 dias, tendo por base os consumos efetuados e o serviço contratado.
- Este Mês: custo energético do presente mês, tendo por base os consumos efetuados e o serviço contratado. Por base incluí o custo com a potência contratada para o mês todo.
- Este Ano: custo energético do presente ano, tendo por base os consumos efetuados e o serviço contratado. Por base incluí o custo com a potência contratada para o ano todo.
- Tarifa: tipo de repartição de tarifa contratado.
- Ciclo: ciclo horário contratado.
- Preço da Energia: preço da energia contratado.

- Potência Contratada: valor da potência contratada.
- Preço Potência: preço diário da potência contratada.

O cálculo dos custos energéticos é feito através de uma vista na base de dados.

Figura 58: Página "Custos", pertencente à página "Energia".

A página "Segurança" (Figura 59) contém a informação relativa ao estado dos sensores presente na habitação, sendo que o seu valor é simulado, com exceção do valor do sensor de presença.

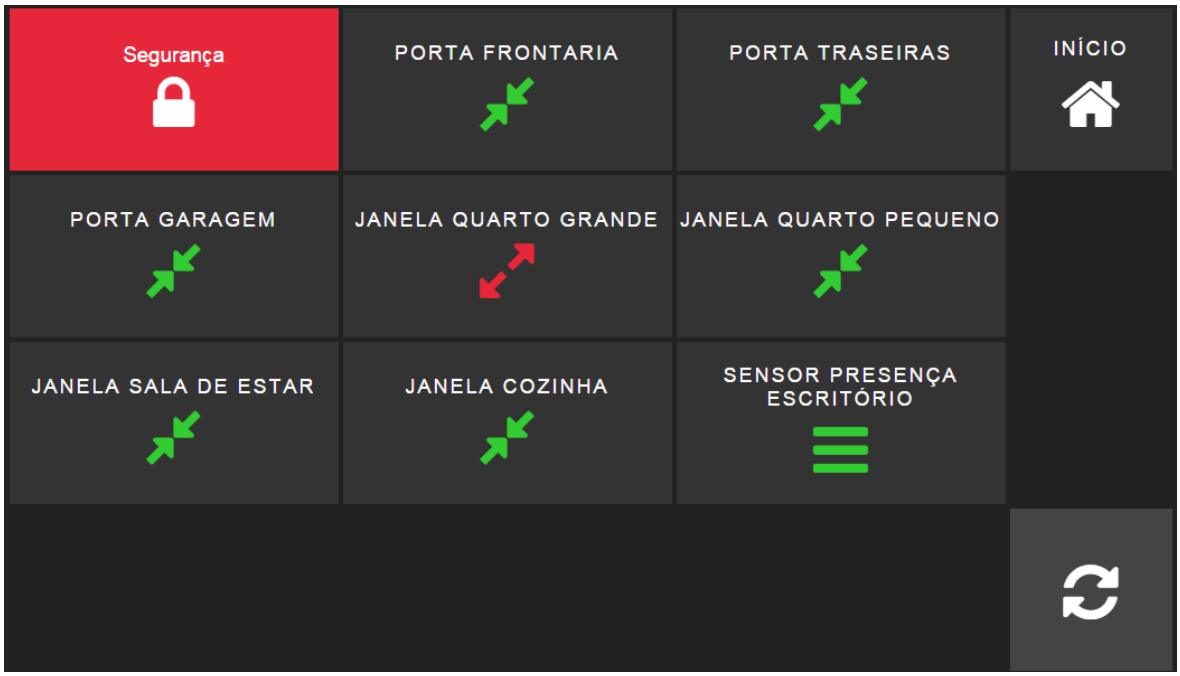


Figura 59: Página "Segurança", sensores com indicação de estado aberto ou fechado.

Na página “Controlo” (Figura 60) é possível controlar o estado da iluminação em algumas áreas da habitação. Foi simulada a existência de iluminação com e sem controlo *PWM* que estaria ligada a um cliente *MQTT*. Os *widgets* presentes na página controlo têm as seguintes caraterísticas:

- Ligar Tudo: liga todos os dispositivos de iluminação.
- Desligar Tudo: desliga todos os dispositivos de iluminação.
- Iluminação Cozinha: liga e desliga a iluminação da Cozinha.
- Iluminação Sala: liga e desliga a iluminação da Sala de estar.
- Modo Cinema: Define a iluminação com controlo PWM para um determinado valor.
- Modo Normal: Define a iluminação com controlo PWM para o seu valor máximo.
- Regulação Intensidade: Define a iluminação com controlo PWM para um valor percentual selecionado.

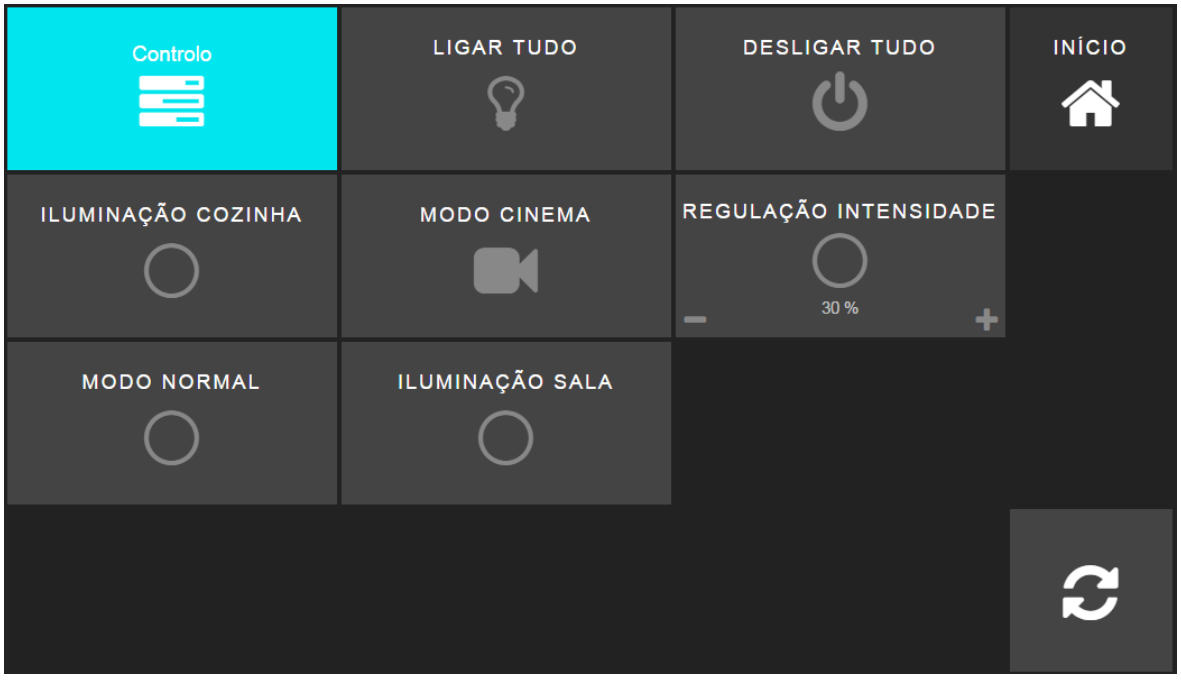


Figura 60:Página “Controlo”.

4.3. Simulações e resultados

A calibração do sistema de captura dos consumos foi feita através de ensaios. Para averiguar qual seria o valor das constantes a aplicar para calcular o valor medido através de amostragem dos sinais condicionados da tensão e corrente.

Para a estipulação do valor constante utilizado no cálculo da tensão, foi utilizado um multímetro digital para comparar o valor da tensão da rede com o valor da tensão medido pelo microcontrolador. Apurou-se que a constante a utilizar para a tensão tem o valor de 0,9.

Foi utilizada uma lâmpada de 100 Watt de potência para ajudar o valor da corrente. Obteve-se o valor constante de 0,29.

Devido ao *Arduino* não ter capacidade de ler o sinal da tensão e corrente em simultâneo, foi necessário adicionar uma compensação que movimenta a onda da tensão no tempo para compensar a diferença temporal das leituras através da soma de uma constante. Caso contrário, ocorreria um erro de leitura relativamente aos valores da potência devido aos sinais da corrente e tensão serem obtidos em momentos distintos, atrasados no tempo.

Após serem feitos os ensaios de calibração o sistema foi ligado no quadro geral da habitação, medindo assim o consumo total da mesma. Foram obtidas leituras ao longo de vários dias, originando uma imensa quantidade de informação. Analisada essa informação obtiveram-se os seguintes valores médios, máximos e mínimos (Tabela 5).

Tabela 5: Resumo das medições efetuadas durante 24 horas

Valor	Tensão (V)	Corrente (A)	FP
Médio	232.64	1.44	0.85
Mínimo	230.04	0.58	0.71
Máximo	237.06	10.23	0.97

Capítulo 5 - Conclusão

5.1. Conclusões

A revisão feita ao estado da arte permitiu identificar várias tecnologias utilizadas em sistemas de automação em edifícios. As suas principais funcionalidades foram enumeradas, assim como as suas vantagens e desvantagens. Esta revisão ajudou na identificação de tecnologias a utilizar neste projeto.

Nem todos os objetivos traçados foram cumpridos. No entanto, os objetivos que ficaram por concretizar foram tidos em conta de modo igual aos objetivos cumpridos, isto é, a sua adição ao SISGE será de fácil introdução tendo em conta que o trabalho base se encontra realizado.

Em certo aspeto o SISGE insere-se na ideologia *IoT* devido à interação que proporciona entre as várias camadas do sistema. As comunicações *MQTT* são de rápido desenvolvimento e conseguem facilmente superar os requisitos de comunicação dos sensores utilizados.

A capacidade do SISGE para medir os consumos numa instalação, permite ao utilizador retirar ilações da relação direta entre o estado de certo equipamento e o consumo, isto em tempo real. A apresentação do preço atual permite ao utilizador decidir se, na altura em que está a efetuar a análise, deve ou não efetuar determinado gasto energético. A apresentação do consumo ao longo das últimas doze horas permite a identificação das fontes de consumo de fundo, ou seja, que estão sempre a consumir, na maior parte do tempo são apenas aparelhos em *standby*.

Os valores medidos são aproximados, mas devido à simplicidade das técnicas utilizadas os resultados são satisfatórios. Este será um dos principais pontos a ter em conta para futuros desenvolvimentos. A medição dos valores da corrente é limitada, isto devido à resolução do *ADC* do *Arduino Mega* não permitir que valores mais baixos sejam detetados.

A simples deteção de presença é suficiente para a deteção de intrusões. O sensor *PIR* utilizado é de baixa qualidade, no entanto os resultados obtidos são bastante satisfatórios, com o sensor a detetar corretamente a existência de movimento durante os ensaios efetuados.

Ao longo da implementação do projeto surgiram diversos contratempos que obrigaram à reformulação de alguns aspetos do projeto. Foram sentidas dificuldades na

implementação da aplicação, devido ao desconhecimento das linguagens de programação orientadas para páginas *web*. Contudo, estas dificuldades foram superadas e a plataforma cumpre a maior parte dos requisitos que inicialmente foram propostos.

O desenvolvimento deste projeto permitiu a aquisição e aplicação de conhecimentos nas mais diversas áreas. Por vezes, foram tomadas decisões que mais tarde se vieram a revelar menos corretas, o que deixa em aberto a necessidade de desenvolvimentos futuros.

Conclui-se que o SISGE tem de facto utilidade prática e é neste momento um projeto com uma flexibilidade expansiva enorme, podendo o mesmo servir de base para desenvolvimentos futuros.

5.2. Futuros desenvolvimentos

No decorrer da realização deste trabalho foram identificados vários aspetos que merecem ser investigados futuramente, nomeadamente:

- Análise dos consumos em instalações trifásicas;
- Adoção de ligações sem fios para todas as unidades periféricas;
- Utilizar uma unidade central mais capacitada ou escolher aplicações mais simples de modo a aumentar o desempenho da unidade;
- Desenvolver uma placa de aquisição dos dados do consumo com microcontrolador integrado na mesma.
- Adicionar sensores às unidades periféricas, *e.g.* temperatura, humidade, fumo entre outros;
- Adicionar um elemento para armazenamento de informação nas unidades periféricas;
- Desenvolver código para o *Arduino* para utilizar as quatro entras para sensores de corrente.

Durante a análise de resultados deu-se conta de algumas falhas que poderiam ter sido evitadas ao serem utilizados outros métodos para captar os sinais de amostra.

A utilização de resistências com valores elevados, apesar de terem baixos consumos de potência, inserem ruído no sinal que se deseja medir. Isto é particularmente desvantajoso quando se estão a medir diferenças mínimas de tensão. Uma das soluções para este problema é utilizar um retificador precisão.

O cérebro da unidade central é o *software Node-Red*. As tarefas para as quais foi programado tornam-se demasiado "pesadas" para as capacidades da unidade *Raspberry Pi*. Uma das soluções para este problema passa pela utilização de outro computador mais potente para servir unidade central. A aplicação carece de uma ligação direta à base de dados, o que permitiria resolver parte do problema das tarefas do *Node-Red*.

Capítulo 6 - Referências Bibliográficas

[1] Springer, Handbook of Automation, Springer.

[2] DGEG/MAOTE, PORDATA, "Consumo de energia eléctrica: total e por tipo de consumo," Junho 2015. [Online]. Available: http://www.pordata.pt/Portugal/Consumo+de+energia+el%C3%A9ctrica+total+e+por+tipo+de+consumo-1124.

[3] S. Wang, Intelligent Buildings and Building Automation, Paperback, 2010.

[4] P. Coelho, *Documentos de apoio à unidade curricular de Controlo de Sistemas e Domótica,* Tomar, 2013.

[5] I. O. f. Standardization, *Norma ISO 50001 Gestão Energia: Desenvolvimento de sistemas de gestão para efeciencia energética*, 2011.

[6] Associação Portuguesa de Segurança Eletrónica e de Proteção de Incêndio, "Sector da Segurança em Portugal," [Online]. Available: http://www.apsei.org.pt/. [Acedido em Abril 2015].

[7] G. G. Teixeira, "Sistemas de Automação e Manutenção de Edifícios - Concepção dos Sistemas de Detecção e Proteção contra Incêndios de Uma Unidade Industrial," Instituto Superior de Engenharia de Lisboa, Lisboa, 2013.

[8] N. Hanna, "Introduction to Fieldus Systems," [Online]. Available: http://people.cs.pitt.edu/~mhanna/Master/Introduction.pdf. [Acedido em Agosto 2015].

[9] "Bluetooth," [Online]. Available: https://en.wikipedia.org/wiki/Bluetooth. [Acedido em Agosto 2015].

[10] Diffen, "Bluetooth vs. Wi-Fi," [Online]. Available: http://www.diffen.com/difference/Bluetooth_vs_Wifi. [Acedido em Agosto 2015].

[11] Ó. D. Javier Garcia, *ZibBee: IEEE 802.15.4,* Tampere: Tampere University of Technology, 2007.

[12] Z-Wave LTD, "Z-Wave," [Online]. Available: http://www.z-wave.com/what_is_z-wave. [Acedido em Setembro 2015].

[13] "Z-Wave," [Online]. Available: https://en.wikipedia.org/wiki/Z-Wave. [Acedido em Setembro 2015].

[14] "Ethernet definition," [Online]. Available: http://searchnetworking.techtarget.com/definition/Ethernet. [Acedido em Setembro 2015].

[15] "TCP/IP," [Online]. Available: https://pt.wikipedia.org/wiki/TCP/IP. [Acedido em Setembro 2015].

[16] X. Carcelle, Power Line Communications in Practice, Artech House, 2009.

[17] Homeplug Alliance, [Online]. Available: http://www.homeplug.or/. [Acedido em Setembro 2015].

[18] G. Almeida, *A Domótica e a Casa do Futuro,* Aveiro, 2006.

[19] KNX Association, [Online]. Available: http://www.knx.org/. [Acedido em Agosto 2015].

[20] KNX Association, "KNX System Specifications - Architecture," 29 Novembro 2013. [Online]. Available: http://www.knx.org/media/docs/downloads/KNX-Standard/Architecture.pdf. [Acedido em Outubro 2015].

[21] Galaxias, "Technology Glance," [Online]. Available: http://www.galaxias.in/technology/.

[22] "LonWorks," [Online]. Available: https://en.wikipedia.org/wiki/LonWorks.

[23] Echelon, "Introduction to the LonWorks Platform Rev2," 2009. [Online]. Available: http://downloads.echelon.com/support/documentation/manuals/general/078-0183-01B_Intro_to_LonWorks_Rev_2.pdf. [Acedido em Outubro 2015].

[24] Enerlon, "LonTalk Adressing Part1," [Online]. Available: http://www.enerlon.com/JobAids/LonTalk%20Addressing.doc. [Acedido em 2015 Outubro].

[25] Echelon, "LonWorks," [Online]. Available: www.echelon.com/technology/lonworks/.

[26] "X10 Home Gadgets," [Online]. Available: http://www.x10.com/. [Acedido em Outubro 2015].

[27] Powerhouse, *X-10 Technical Note, Power Line Interface and Two-way Power Line Interface.*

[28] Bernardo, "M2M (Machine-To-Machine)," [Online]. Available: http://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2014_2/bernardo/pages/introducao.html.

[29] " Taking the First Step with PDCA," 2009. [Online]. Available: http://www.bulsuk.com/2009/02/taking-first-step-with-pdca.html.

[30] Domática, "Domática M2M Gateway," [Online]. Available: http://www.domaticasolutions.com/products/domatica-m2m-gateway/.

[31] Domatica, "I/O Controller," [Online]. Available: http://www.domaticasolutions.com/products/io-controller/.

[32] Honeywell, "HALL EFFECT SENSING AND APPLICATION," [Online]. Available: http://sensing.honeywell.com/index.php?ci_id=47847. [Acedido em 2015 Setembro].

[33] BBC, "Can a £15 computer solve the programming gap?," [Online]. Available: http://news.bbc.co.uk/2/hi/programmes/click_online/9504208.stm.

[34] *Arduino*, "*Arduino* Mega 2560," [Online]. Available: https://www.*Arduino*.cc/en/Main/*Arduino*BoardMega2560.

[35] Hahn, "BV 202 0157," [Online]. Available: http://www.hahn-trafo.com/deutsch/daten/2020157.htm.

[36] Adafruit, "PIR Sensors," [Online]. Available: https://learn.adafruit.com/downloads/pdf/pir-passive-infrared-proximity-motion-sensor.pdf. [Acedido em Outubro 2015].

[37] *Arduino*, "PIRsense code," [Online]. Available: http://playground.*Arduino*.cc/Code/PIRsense.

[38] IBM, "MQTT Protocol Specification," [Online]. Available: http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html. [Acedido em Novembro 2015].

[39] Atmel, "Single phase power energy meter with tamper detection ap notes," AVR, 2013.

[40] ERSE, "Periodos Horarios," [Online]. Available: http://www.erse.pt/pt/electricidade/tarifaseprecos/periodoshorarios.

[41] I. E. Agency, "International Energy Agency, Secure Sustainable Together," [Online]. Available: http://www.iea.org/. [Acedido em Março 2015].

[42] "KNX (standard)," [Online]. Available: https://en.wikipedia.org/wiki/KNX_(standard).

[43] Element14, "Raspberry Pi 2 Model B GPIO 40 Pin Block Pinout," [Online]. Available: http://www.element14.com/community/docs/DOC-73950/l/raspberry-pi-2-model-b-gpio-40-pin-block-pinout.

[44] J. Paiva, Redes de Energia Eléctrica - Uma Análise Sistémica, Ist Press, 2005.

[45] http://www.knx.org/.

[46] YHDC, "YHDC SCT-013-030," [Online]. Available: http://www.yhdc.com.

[47] GIT - stonehippo, "Setting up a Raspberry Pi as a dashboard server with Dashing," [Online]. Available: https://gist.github.com/stonehippo/5896381.

[48] Farnell, "Raspberry Pi 2," [Online]. Available: http://pt.farnell.com/productimages/standard/en_GB/2461029-40.jpg.

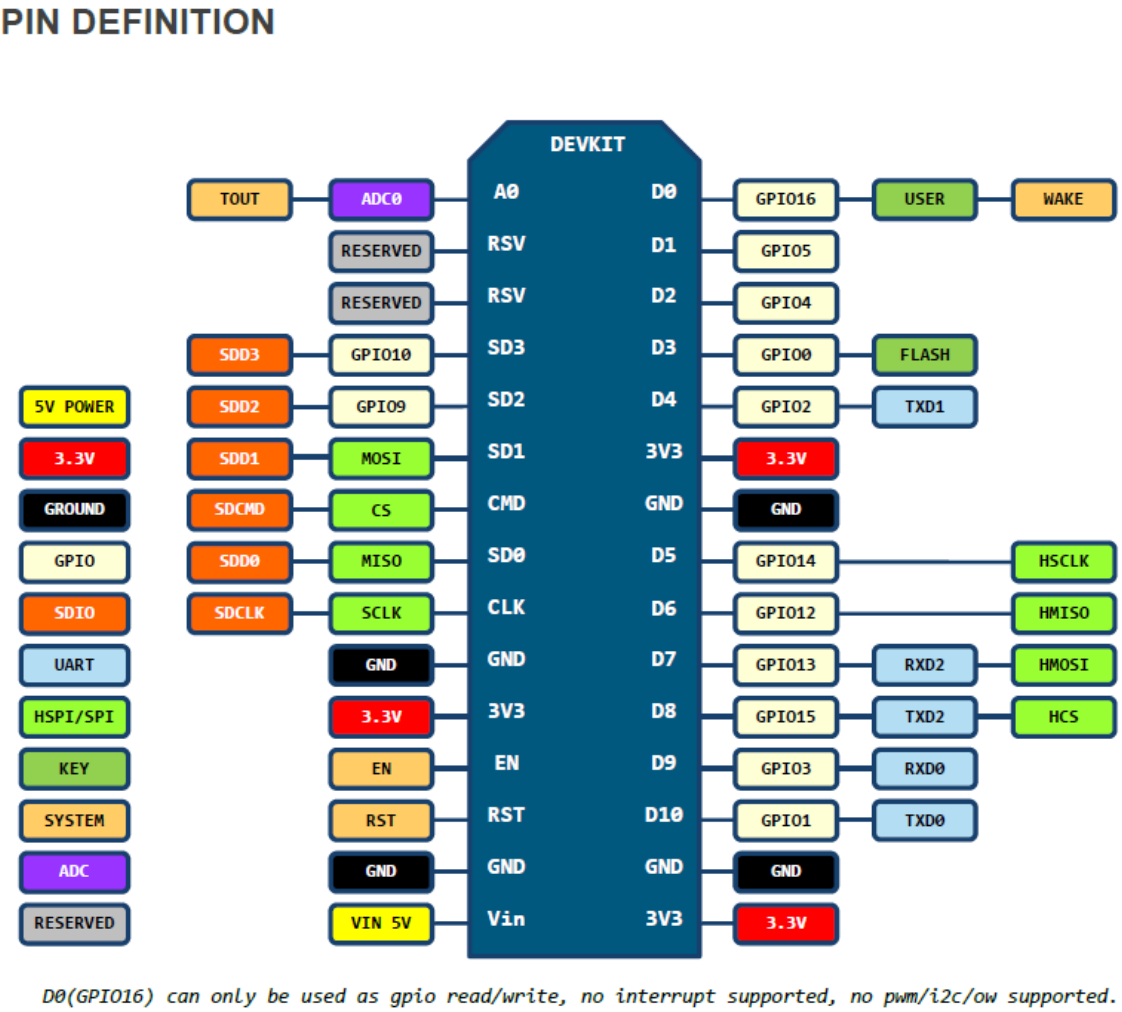
Capítulo 7 - Anexos

Anexo A – Portas Raspberry Pi2

Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I²C)		DC Power 5v	04
05	GPIO03 (SCL1 , I²C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)		(I²C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Autor: *Element14*, retirada a página de venda do equipamento.

Anexo B – Portas *NodeMCU*.



Autor: *Zeroday nodemcu*, retirada da página *web* https://github.com/nodemcu.

Anexo C – Código de programação do *Arduino Mega*

Partes deste código são baseadas nos exemplos fornecidos com as bibliotecas utilizadas.

```
//Inclusão de bibliotecas
#include <Ethernet.h>
#include <PubSubClient.h>
#include <SPI.h>

// Definições de rede
byte mac[]    = {  0xDE, 0xEE, 0xBA, 0xFE, 0xFE, 0xED };
IPAddress ip(192, 168, 1, 11);
IPAddress server(192, 168, 1, 68);

//Variaveis
int samples = 1500; //número de amostras
int inU = A0;//Portas de entrada tensão e corrente
int inI0 = A4;
double UCAL = 0.9;
double ICAL = 0.29;
double calibFase = 0.1;
int lastU,lastI,sampleU,sampleI;
double lastFilterU, lastFilterI, filterU, filterI = 0;
double filterTemp;
double calibU;
double sqrtI,sqrtU,instP,sumI,sumV,sumP;
double activPower,apparentPower,powerFactor,Vrms,Irms;
unsigned long postingInterval = 6000;  //Intervalo de envios
unsigned long lastRequest = 0;
long now = 0;
int restart = 0;
unsigned long endTime, startTime, timeLastMeasure;
unsigned long startUpTime;
double energyTotal  =0.0;
unsigned long LastZero;
unsigned long periodU;
unsigned long periodUSum;
unsigned long periodUCount;
float freq;
```

```
unsigned long sinPeriod = 20000;
unsigned long filterWidth = 2000;
//recebe as mensagens vindas do topico mqtt subscrito
//atualiza as constantes de calibração, tempo e estado
void callback(char* topic, byte* payload, unsigned int length) {
   Serial.print("Message arrived [");
   Serial.print(topic);
   Serial.print("] ");
   char msgBuffer[100];
    for (int i=0;i<length;i++) {
      msgBuffer[i] = payload[i];
    Serial.print((char)payload[i]);
   }
   String msg = String(msgBuffer);
   Serial.println();
   if(msg.startsWith("sendTime")){
    postingInterval = msg.substring(8).toFloat();
    Serial.println(postingInterval);
   }else if(msg.startsWith("VConstant")){
    UCAL = msg.substring(9).toFloat();
   }else if(msg.startsWith("IConstant")){
    ICAL = msg.substring(9).toFloat();
   }else if(msg.startsWith("setToday")){
    energyTotal  = msg.substring(8).toFloat();
   }else if(msg.equals("newDay")){
    energyTotal  = 0;
   }else if(msg.startsWith("device")){
    digitalWrite(msg.substring(6,7).toInt()+5,msg.substring(7,8).toInt());
   }
}

EthernetClient ethClient;
PubSubClient client(ethClient);
//reinicia o programa
void software_Reset()
    {
    asm volatile ("  jmp 0");
    }
```

```
 void reconnect() {
  while (!client.connected()) {
   Serial.print("Attempting MQTT connection...");
   if (client.connect("arduinoClient")) {
    Serial.println("connected");
    client.subscribe("sisge/control");
    String getSettings = "getSettings";
    char getSettingsArray[getSettings.length()];
    getSettings.toCharArray(getSettingsArray, getSettings.length()+1);
    client.publish("sisge/settings", getSettingsArray);
    restart = 0;
   } else {
    Serial.print("failed, rc=");
    Serial.print(client.state());
    Serial.println(" try again in 5 seconds");
    restart++;
    if(restart>=12){
     software_Reset();
    }
    delay(1000);
   }
  }
 }
int Energy()
{
 for (int n=0; n<samples; n++)
 {
  //guarda ultimas amostras
  lastU=sampleU;
  lastI=sampleI;
  //Recolhe amostras
  sampleU = analogRead(inU);
  sampleI = analogRead(inI0);
  //guarda ultimos valores filtrados
  lastFilterU = filterU;
  lastFilterI = filterI;
  //Aplicação do filtro digital.
  filterU = 0.996 * (lastFilterU+sampleU-lastU);
```

```
    filterI = 0.996 * (lastFilterI+sampleI-lastI);
    //Calibração de fase
    calibU = lastFilterU + calibFase * (filterU - lastFilterU);
    //Tensão RMS
    sqrtU= calibU * calibU;
    sumV += sqrtU;
    //Corrente RMS
    sqrtI = filterI * filterI;
    sumI += sqrtI;
            //Potência instantanea
    instP = abs(calibU * filterI);
    sumP += instP;
    //Frequência
    if (n==0) LastZero = micros();
    //Verifica se está a passar por zero
    if (lastFilterU < 0 && filterU >= 0 && n>1)
    {
      //Periodo da onda da tensão
      periodU = micros() - LastZero;
      //Filtra leituras erradas
      if (periodU > (sinPeriod-filterWidth) && periodU<(sinPeriod+filterWidth))
      {
        periodUSum += periodU;
        periodUCount++;
      }
      LastZero = micros();
    }
  }
  //Cálculo dos valores aproximados
  Vrms = UCAL*sqrt(sumV / samples);
  Irms = ICAL*sqrt(sumI / samples);
  //Potência
  activPower = UCAL*ICAL*sumP / samples;
  apparentPower = Vrms * Irms;
  if(apparentPower>activPower){
  powerFactor = activPower / apparentPower;
  }else{
    powerFactor=1;
```

```
 }
 //Frequência
 freq = (1000000.0 * periodUCount) / periodUSum;

 periodUSum=0;
 periodUCount=0;
 //Calcula o tempo desde a ultima medição
 endTime = startTime;
 startTime = millis();
 timeLastMeasure = startTime - endTime;
 //Calcula a energia consumida no intervalo e adiciona à soma dos kWh
 energyTotal  = energyTotal  + ((activPower/1000.0) * 1.0/3600.0 * (timeLastMeasure/1000.0));
 Serial.print("RP=");
 Serial.print(activPower);
 Serial.print(" AP=");
 Serial.print(apparentPower);
 Serial.print(" PF=");
 Serial.print(powerFactor);
 Serial.print(" V=");
 Serial.print(Vrms);
 Serial.print(" I=");
 Serial.print(Irms);
 Serial.print(" KW=");
 Serial.print(energyTotal );
 Serial.print(" F=");
 Serial.println(freq);
          //envia mensagem
if (now - lastRequest >= postingInterval) {
 if(freq){freq=50;}
 String                                          energyStr                                          =
          "Ard001;Monitor1;"+String(Vrms)+";"+(String)Irms+";"+(String)activPower+";"+(String)appare
          ntPower+";"+(String)powerFactor+";"+(String)energyTotal +";"+(String)freq;
 int  energyStrLength = energyStr.length();
 char energyArray[energyStrLength];
 energyStr.toCharArray(energyArray, energyStrLength);
 client.publish("sisge/energy", energyArray);
 Serial.println("Message sent");
    lastRequest = now;
```

```
  }

  sumV = 0;
  sumI = 0;
  sumP = 0;
}
void setup()
{
        //inicia os resgitos relativos à porta Serie de debug,
        //cliente mqtt, ethernet, e define quais os pinos de saída para os atuadores
  Serial.begin(9600);
  //Define a referência do ADC para 2,56V
  analogReference(INTERNAL2V56);
  client.setServer(server, 1883);
  client.setCallback(callback);
  Ethernet.begin(mac, ip);
  startTime = millis();
  startUpTime=startTime;
  lastRequest = millis();
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  digitalWrite(6,0);
  digitalWrite(7,0);
  digitalWrite(8,0);
  digitalWrite(9,0);
}
void loop()
{
  now = millis();
  if (!client.connected()) {
    reconnect();
 }
  client.loop();
  Energy();
  delay(100);
}
```

Anexo D – Código de programação do *NodeMCU*

Partes deste código são baseadas nos exemplos fornecidos com as bibliotecas utilizadas. Biblioteca e procedimentos de instalação para utilização deste código no *IDE Arduino* em:

https://github.com/esp8266/*Arduino*-esp8266fs-plugin

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

// Config Rede
const char* ssid = "_NOME_DA_REDE_";
const char* password = "_PALAVRA_PASS_DA_REDE_";
const char* mqtt_server = "192.168.1.68";
//Tempo em segundos para o sensor se calibrar ao ambiente
int calibrationTime = 15;
//tempo em que o sensor passa a "0"
long unsigned int lowIn;
//Temmpo em que se considera que não haverá mais movimento
long unsigned int pause = 5000;
//Variaveis de estado
boolean lockLow = true;
boolean takeLowTime;
//Porta à qual está ligado o sensor PIR
int pirPin = D1;
// Inicia o cliente WiFi e MQTT
WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;
int active = 1;
void setup() {
  //inicia a resistência de pullup interna
  pinMode(pirPin, INPUT);
  Serial.begin(9600); //inicia a ligação série
```

```
  setup_wifi(); //efetua a ligação à rede wireless
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
    Serial.print("calibrating sensor ");
    for(int i = 0; i < calibrationTime; i++){
      Serial.print(".");
      delay(1000);
      }
    Serial.println(" done");
    Serial.println("SENSOR ACTIVE");
    delay(50);
}

void setup_wifi() {

  delay(10);
  // Efetua a ligação à rede wireless
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);
  //espera até que a ligação esteja concluida
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
```

```
  char msgBuffer[100];
  for (int i = 0; i < length; i++) {
     msgBuffer[i] = payload[i];
    Serial.print((char)payload[i]);
  }
String msg = String(msgBuffer);
    Serial.println();
    if(msg.startsWith("active")){
            //define estado de envio
      active = msg.substring(6,7).toInt();
      Serial.println(active);
    }

}

void reconnect() {
  // Loop até que a ligação ao broker seja efetuada
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // tentativa de ligação
    if (client.connect("ESP8266Client")) {
      Serial.println("connected");
      // Assimm que ligado publica mensagem
      client.publish("sisge/security", "PIRSensor-ON");
      // subscreve o topico de controlo
      client.subscribe("sisge/control");
      String getSettings = "getSettings";
      char getSettingsArray[getSettings.length()];
      getSettings.toCharArray(getSettingsArray, getSettings.length()+1);
      client.publish("sisge/settings", getSettingsArray);
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // espera 10 seundos até voltar a tentar
      delay(10000);
    }
  }
```

```
}

    void pirSensor(){
     if(digitalRead(pirPin) == 1){
     digitalWrite(BUILTIN_LED, LOW);//Liga o LED interno caso haja ativação do PIR
     if(lockLow){
       lockLow = false;
       Serial.println("---");
       Serial.print("motion detected at ");
       Serial.print(millis()/1000);
       Serial.println(" sec");
                        //Se o envio estiver ativo comunica à central o estado
       if(active==1){
       snprintf (msg, 75, "Alarm-ON");
       client.publish("sisge/security", msg);
       }
       delay(50);
       }
       takeLowTime = true;
     }

   if(digitalRead(pirPin) == 0){
     digitalWrite(BUILTIN_LED, HIGH); //Desliga o LED interno caso não haja ativação do PIR

     if(takeLowTime){
      lowIn = millis();
      takeLowTime = false;
      }
     if(!lockLow && millis() - lowIn > pause){
         lockLow = true;
         Serial.print("motion ended at ");      //output
         Serial.print((millis() - pause)/1000);
         Serial.println(" sec");
                          //Comunica à central estado
         snprintf (msg, 75, "Alarm-OFF");
          client.publish("sisge/security", msg);
         delay(50);
         }
```

```
      }
      }

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  pirSensor();

}
```

Anexo E – Principais *scripts* da aplicação SISGE

Utilizados *widgets* disponíveis em:

https://github.com/Shopify/dashing

https://github.com/FlorianZ/hadashboard

Script de atualização dos *widget* da aplicação.

```
 # encoding: UTF-8
require 'net/http'
require 'json'
require 'mysql2'
#URL de acesso aos servicos REST do node-red
urlLive = 'http://localhost:1880/services/rest/getEnergy'
url5Min = 'http://localhost:1880/services/rest/getLast5Min'
url24h = 'http://localhost:1880/services/rest/getLast24h'
urlgetList = 'http://localhost:1880/services/rest/getList'
urlgetContracted = 'http://localhost:1880/services/rest/getContracted'

#Indicadores de grandezas eletricas
SCHEDULER.every '5s' do
        liveResponse = Net::HTTP.get_response(URI.parse(urlLive))
        liveResponseText = liveResponse.body
        parsedLive = JSON.parse(liveResponseText)
        send_event('volt',   { value: parsedLive[0]["VOLT"] })
        send_event('watt',   { value: parsedLive[0]["TOTAL_POWER"] })
        send_event('amper',   { value: parsedLive[0]["TOTAL_CURRENT"] })
        send_event('today', { value: parsedLive[0]["TOTAL_TO_ENERGY"] })
        data = [
          { label: "Consumo", value: parsedLive[0]["TOTAL_TO_ENERGY"] },
          { label: "Produção", value: parsedLive[0]["PRO_TO_ENERGY"] },
        ]
        send_event('energy_now', { value: data })
        send_event('current_cost',   { value: parsedLive[0]["PRICE"] })
        send_event('hour_type',   { value: parsedLive[0]["HOUR_TYPE"] })
        send_event('tarif_type',   { value: parsedLive[0]["TYPE"] })
        send_event('cycle',   { value: parsedLive[0]["CYCLE_TYPE"] })
```

```
        send_event('contracted',   { value: parsedLive[0]["CONTRACTED"] })
        send_event('cost_today',   { value: parsedLive[0]["DAY"] })
        send_event('cost_week',   { value: parsedLive[0]["WEEK"] })
        send_event('cost_month',   { value: parsedLive[0]["MONTH"] })
        send_event('cost_week',   { value: parsedLive[0]["WEEK"] })
        send_event('cost_year',   { value: parsedLive[0]["YEAR"] })
        send_event('contracted_price',   { value: parsedLive[0]["POWER_PRICE"] })
        send_event('service',   { value: parsedLive[0]["SERVICE"] })
end

#Grafico ultimos 5 minutos
SCHEDULER.every '5s' do
   fiveMinResponse = Net::HTTP.get_response(URI.parse(url5Min))
        fiveMinResponseText = fiveMinResponse.body
        parsedFiveMin  = JSON.parse(fiveMinResponseText)
        series = [];
        series[0]=parsedFiveMin[0];
   send_event('last5min', series: series)
end

#Grafico ultimas 12 horas
SCHEDULER.every '300s' do
        h24Response = Net::HTTP.get_response(URI.parse(url24h))
        h24ResponseText = h24Response.body
        parsed24h       = JSON.parse(h24ResponseText)
        series = [];
        series[0]=parsed24h[0];
   send_event('last24h', series: series)
 end

#listas de consumo atual e diario
SCHEDULER.every '10s' do
        getListResponse = Net::HTTP.get_response(URI.parse(urlgetList))
        getListResponseTxt = getListResponse.body
        parsedList      = JSON.parse(getListResponseTxt)
```

```
   send_event('list_now', { items: parsedList[0] })
          send_event('list_today', { items: parsedList[1] })
 end

#lista de precos da energia
SCHEDULER.every '600s' do
          getContractedResponse = Net::HTTP.get_response(URI.parse(urlgetContracted))
          getContractedResponseTxt = getContractedResponse.body
          parsedgetContracted      = JSON.parse(getContractedResponseTxt)
   send_event('list_hours', { items: parsedgetContracted })
 end
```

Script para interação com os serviços *REST* criados no *Node-RED*.

```
require 'json'

#URL da aplicacao Dashing
host_uri = 'http://localhost:3030'

app = NRApp.new()
#Servico de atualizacao dos sensores
get '/services/rest' do
  app.getSensorStatus(params['sensorId'],params['deviceType'], params)
end
#Servico de comando dos atuadores
post '/services/rest' do
  app.sendCommand(params['sensorId'],params['deviceType'], params['command'],params)
end
#Servico de atualizacao dos estado dos atuadores
get '/services/rest/status' do
  app.setSensorStatus(params['sensorId'],params['status'], params)
end
#Servico de atualizacao dos estado do dimmer
get '/services/rest/level' do
  app.setSensorLevel(params['sensorId'],params['level'], params)
end
```

Anexo F – DVD-ROM

Aplicações SISGE:

- *Pasta Projeto_Dashing* – ficheiros relativos ao projeto *Dashing.*
    - Pasta sisge/*jobs* contém os scripts utilizados;
    - Pasta sisge/*widgets* contém os *widgets* utilizados;
    - Pasta sisge/*dashboards* contém as páginas principais.
    - Pasta sisge/*lib* contém o script de ligação ao *Node-RED*

Documentos consultados:

- *Atmel ATmega640V-1280V-1281V-2560V-2561V.pdf*
- *Atmel AVR465 Single-Phase PowerEnergy Meter.pdf*
- *ERSE-PreçosRef_BTN.pdf*
- *ESPRESSIF SMART CONNECTIVITY PLATFORM ESP8266.pdf*
- *HC-SR501 PIR.pdf*
- *IBM MQTT.pdf*
- *ST - UNDERSTANDING POWER FACTOR 1999.pdf*
- *ST -UNDERSTANDING POWER FACTOR 2003.pdf*
- *W5100 Datasheet.pdf*
- *YHDC SCT-013-000.pdf/YHDC SCT-013-030.pdf*

Export *Node-Red:*

- Pasta *Node-Red* – contém a extração do s fluxos da aplicação *Node-RED*

Figuras utilizadas no projeto:

- Pasta *figuras*

Projetos *Arduino*:

- Pasta *Projetos Arduino IDE*

Projeto *Eagle*

- Pasta *Eagle* – contém o projeto da placa de circuito impresso.

Scripts *MySQL* (*SQL*)*:*

- Pasta *db/Dados* – Contém os scripts com dados para inserção nas tabelas;
- Pasta *db/Tabelas* - Contém os scripts para criação das tabelas;
- Pasta *db/Vistas* - Contém os scripts para criação das vistas.
- Pasta *db/Triggers* - Contém os scripts para criação *Triggers* e eventos;