UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
ENG. DE CONTROLE E AUTOMAÇÃO

ETHAR ANDRÉ SIRENA TEIXEIRA

ARQUITETURA IOT PARA APLICAÇÃO EM AUTOMAÇÃO RESIDENCIAL

Porto Alegre
2019

ETHAR ANDRÉ SIRENA TEIXEIRA

ARQUITETURA IOT PARA APLICAÇÃO EM AUTOMAÇÃO RESIDENCIAL

Trabalho de Conclusão de Curso (TCC-CCA) apresentado à COMGRAD-CCA da Universidade Federal do Rio Grande do Sul como parte dos requisitos para a obtenção do título de *Bacharel em Eng. de Controle e Automação* .

ORIENTADOR: Prof. Dr. Renato Ventura Bayan Henriques

Porto Alegre
2019

ETHAR ANDRÉ SIRENA TEIXEIRA

ARQUITETURA IOT PARA APLICAÇÃO EM AUTOMAÇÃO RESIDENCIAL

Este Trabalho de Conclusão de Curso foi julgado adequado para a obtenção dos créditos da Disciplina de TCC do curso *Eng. de Controle e Automação* e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____
Prof. Dr. Renato Ventura Bayan Henriques, UFRGS
Doutor pela Universidade Federal de Minas Gerais – Belo Horizonte, Brasil

Banca Examinadora:

Prof. Dr. Renato Ventura Bayan Henriques, UFRGS
Doutor pela Universidade Federal de Minas Gerais – Belo Horizonte, Brasil

Prof. Dr. Marcelo Götz, UFRGS
Doutor pela Universität Paderborn – Paderborn, Alemanha

Prof. Dr. Ivan Müller, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

_____
Marcelo Götz
Coordenador de curso
Eng. de Controle e Automação

Porto Alegre, dezembro de 2019.

AGRADECIMENTOS

Gostaria de agradecer primeiramente à Universidade Federal do Rio Grande do Sul e aos meus pais pela oportunidade que tive de realizar os estudos. Também quero agradecer a parceria dos colegas do curso que me acompanharam durante minha formação. Um agradecimento especial ao Maurício Simões Posser que me auxiliou imensamente neste trabalho desde a idealização até o momento em que foi concebido, além da Elipse Software por me proporcionar a estrutura necessária para a realização deste. Agradeço por fim ao meu orientador prof. Renato Ventura pelo apoio e por se mostrar presente sempre que precisei.

RESUMO

Em 2013 quase metade de todo o consumo energético no Brasil foi originário de edificações, o que evidencia a importância de se desenvolver soluções mais eficientes energeticamente nestas áreas. O objetivo do presente trabalho é desenvolver uma arquitetura de baixo custo para um cenário de Automação Residencial trazido para o contexto da Internet das Coisas (IoT) que visa proporcionar, além de eficiência energética, o conforto térmico nos ambientes, fator este que tem grande impacto na produtividade das pessoas envolvidas. Para isto, na arquitetura desenvolvida foram monitorados os indicadores de temperatura, umidade e luminosidade de dois ambientes de uma residência através de duas placas equipadas com sensores. Fazendo-se uso do protocolo de comunicação MQTT e de um sistema PIMS, os dados lidos pelos sensores puderam ser constantemente armazenados e apresentados em um painel por meio da Internet para visualização do usuário. Concluiu-se, por fim, que a integração entre os componentes utilizados nesta arquitetura proporcionou um sistema eficaz de monitoramento de ambientes residenciais com uma grande capacidade de armazenamento, podendo ser utilizado como base para um projeto de sistema de Automação Residencial dedicado a garantir eficiência energética e conforto térmico.

Palavras-chave: Automação Residencial, Internet das Coisas, Protocolo de comunicação MQTT, Sistemas PIMS.

ABSTRACT

In 2013 almost half of all energy consumption in Brazil originated from buildings, which highlights the importance of developing more energy efficient solutions in these areas. The objective of this work is to develop a low cost architecture for a Home Automation scenario brought to the context of the Internet of Things (IoT) that aims to provide, in addition to energy efficiency, thermal comfort in environments, a factor that has great impact on the productivity of the people involved. For this, in the developed architecture were monitored the temperature, humidity and light indicators of two environments of a residence through two plates equipped with sensors. Using the MQTT communication protocol and a PIMS system, the data read by the sensors could be constantly stored and displayed on a panel via the Internet for user viewing. Finally, it was concluded that the integration between the components used in this architecture provided an effective monitoring system for residential environments with a large storage capacity, and can be used as a basis for a home automation system project dedicated to ensuring energy efficiency and thermal comfort.

Keywords: Home Automation, Internet of Things, MQTT Protocol, PIMS System.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

IoT	*Internet of Things*
PIMS	*Plant Information Management System*
EPE	Empresa de Pesquisa Energética
M2M	*Machine-to-Machine*
QoS	*Quality of Service*
KPI	*Key Performance Indicator*
IDE	*Integrated Development Environment*
UTC	*Coordinated Universal Time*
EPM	*Elipse Plant Manager*
MQTT	*Message Queue Telemetry Transport*
TCP/IP	*Transmission Control Protocol / Internet Protocol*

1 INTRODUÇÃO

Segundo (LAMBERTS et al., s.d.), em 2001 o Brasil passou por uma crise energética devido ao grande crescimento no consumo e o pequeno crescimento da capacidade instalada. A preocupação com esta questão, ainda segundo (LAMBERTS et al., s.d.), colocou em foco a busca por eficiência energética não somente pelas questões econômicas, mas também pela necessidade de qualidade do ambiente para o usuário e a redução de impactos ambientais.

Tomando como base o Anuário Estatístico de Energia Elétrica, em 2013 48,3% do consumo energético no Brasil foi originário de edificações, evidenciando assim a importância de se desenvolver soluções mais eficientes energeticamente nestas áreas. De acordo com (CASTRO; JOTA; ASSIS, 2005) a implementação de sistemas de automação, como de condicionadores de ar e iluminação, podem levar a um uso mais racional de energia nas edificações. Ademais, através de uma pesquisa realizada por (SILVA, L. B. DA, s.d.) constatou-se que o desconforto térmico é o fator que mais tem impacto na variabilidade da produtividade de um indivíduo, sendo assim importante também que seja mantido o conforto térmico nos ambientes além de se garantir a economia de energia.

Segundo (TEZA, s.d.) o termo Automação Residencial, ou Domótica, designa e referencia a utilização de processos automatizados em casas, apartamentos e escritórios. O objetivo do presente trabalho é desenvolver uma arquitetura de baixo custo para o monitoramento de dados em um cenário de Automação Residencial trazido para o contexto da IoT (Internet das Coisas).

Os novos produtos de IoT no setor de Automação Residencial estão nos levando a uma visão do que é chamado de *Smart Home* (ROSE; ELDRIDGE; CHAPIN, 2015). No experimento prático deste trabalho serão efetuadas leituras de temperatura, umidade e iluminação em dois ambientes da casa do autor, seguindo casos típicos de projetos de Automação Residencial. Estes dados monitorados são essenciais para se desenvolver um sistema de racionamento de energia inteligente e manter constante o conforto térmico nos ambientes.

Na arquitetura aqui proposta, para efetuar estas leituras serão utilizadas duas placas, cada uma equipada com sensores de temperatura, umidade e iluminação. As placas irão enviar estes dados através do protocolo de comunicação MQTT (*Message Queue Telemetry Transport*) para um sistema informação, neste caso um sistema PIMS (*Plant Information Management System*), que irá armazená-los em um servidor remoto e apresentá-los por meio da Internet em um painel para visualização do usuário.

De acordo (CARVALHO et al., 2005) são muitos os benefícios gerados pela implantação de um sistema PIMS, como a centralização dos dados de processo, a visualização em tempo real do processo produtivo e o armazenamento de dados históricos, que pode chegar até 15 anos de dados devido ao seu algoritmo de compressão. Já o pro-

tocolo de comunicação MQTT foi utilizado pois, de acordo com (YUAN, 2017), ele é um protocolo de rede leve e flexível, as propriedades ideais para se desenvolver dentro de um sistema IoT.

A estrutura do presente trabalho é baseada em três capítulos: referencial teórico, metodologia e resultados. No capítulo do referencial teórico são explicados todos os conceitos aqui utilizados, explorando a história e as variadas definições contidas dentro das áreas de Automação Residencial e IoT. Sendo o sistema PIMS também explicado com mais detalhes, ao final deste capítulo já é possível se ter uma noção geral dos conteúdos envolvidos e a motivação da uso conjunto destes para se desenvolver a arquitetura aqui proposta.

No capítulo de metodologia é primeiro dada uma visão geral de como será a arquitetura IoT para Automação Residencial que será desenvolvida como experimento prático, demonstrando todos os componentes envolvidos nesta rede de comunicação. Depois é apresentado, através de um fluxograma, como irá funcionar o algoritmo que estará embarcado dentro das placas para elas executarem as leituras dos sensores e enviarem estes dados pela Internet.

Por fim, no capítulo dos resultados são apresentados os componentes utilizados para a montagem das placas, alguns detalhes do programa desenvolvido para elas, a configuração da comunicação pelo protocolo MQTT através de uma estrutura na nuvem, a configuração do sistema PIMS que foi utilizado e o painel construído para apresentar os dados através da Internet. Neste capítulo acompanha-se todas as etapas do processo de desenvolvimento desta arquitetura IoT e vislumbra-se os resultados obtidos ao longo de praticamente dois meses com o sistema em funcionamento.

2 REFERENCIAL TEÓRICO

2.1 Automação Residencial

De acordo com (TEZA, s.d.) a automatização pode ser considerada como qualquer processo que auxilie o Ser Humano nas suas tarefas, sendo elas comerciais, industriais, domésticas ou no campo. Também pode-se definir a automatização como o processo em que são utilizados dispositivos automáticos, eletrônicos e inteligentes para se dar a automação dos processos em questão. À partir disto, a automação se desenvolveu em diferentes áreas de atuação, sendo elas: Automação Industrial, Automação Comercial, Automação Predial e Automação Residencial.

O termo Automação Residencial designa e referencia a utilização de processos automatizados em casas, apartamentos e escritórios. Podem-se utilizar outras denominações sinônimas, tais como, Automação Doméstica, Automatização Residencial ou Domótica (TEZA, s.d.).

Por motivos econômicos e de escala de produção, o desenvolvimento dos sistemas de automação residencial só se dá início depois de seus similares nas áreas industrial e comercial, tendo seu conceito estabelecido em meados dos anos 60. É possível afirmar que este campo está em crescente desenvolvimento nos últimos anos, impulsionado principalmente pelas necessidades atuais de segurança, qualidade de vida e eficiência energética (FREITAS et al., 2010).

Ao projetar-se uma automação residencial, procura-se sempre a integração e interação entre dispositivos eletrônicos relacionados a variados sistemas domésticos. (TEZA, s.d.) em seu artigo, cita o que para ele são alguns dos principais sistemas de Automação Residencial que estão em evidência e estão sendo efetivamente estudados atualmente, que são:

- Segurança;
- Entretenimento;
- Controle de Iluminação;
- *Home Office*;
- Ar Condicionado e Aquecimento;
- Eletrodomésticos Inteligentes;
- Serviços Inteligentes;
- Infraestrutura;

- Controladores e Centrais de Automação;
- Funcionalidades Auxiliares.

Na área da segurança, alguns exemplos de aplicação são os alarmes, controle de acesso e reconhecimento facial. O controle de iluminação e o aquecimento são importantes na busca de economia de energia, o que pode ser motivado tanto por motivos ecológicos quanto monetários. Os eletrodomésticos inteligentes e outros serviços inteligentes, como portas e cortinas automáticas, são sistemas que visam facilitar a vida do usuário e trazer um maior conforto dentro da casa. Como funcionalidades auxiliares estão inclusos sistemas como sistemas de energia solar ou sistemas de irrigação de jardins e hortas (TEZA, s.d.).

Dentro dos sistemas de *Home Office* estão a telefonia e as redes domésticas, enquanto que os sistemas de áudio e vídeo, como por exemplo o *Home Theater*, são definidos como sistemas de entretenimento. A infraestrutura e as centrais de automação são comuns na construção de um sistema de automação residencial, nos quais estão inclusos o cabeamento dedicado ou estruturado e os *hardwares* e *softwares* de controle de integração (TEZA, s.d.).

Quanto mais efetivo o projeto de integração entre os vários sistemas maior serão os benefícios obtidos com a automação nas edificações, considerando aspectos de integração de operação de engenharia dos sistemas e infraestrutura inteligente. Sendo assim, o ponto chave da operação efetiva em um edifício inteligente é a integração entre os seviços, sistemas e estrutura (ARKIN; PACIUK, 1995). Esta constatação, apesar de direcionada a edifícios inteligentes, se extendida para o caso doméstico, o que é possível por suas similaridades, reforça a importância da integração entre diferentes sistemas para uma Automação Residencial eficiente.

2.1.1 Eficiência Energética

Tendo em vista o papel de grande importância da energia no desenvolvimento mundial, ela tem se tornado motivo de preocupação nas últimas décadas (LAMBERTS et al., s.d.). (LAMBERTS et al., s.d.) também afirma que o grande crescimento do consumo de energia elétrica nas últimas décadas, no Brasil e no mundo, tem colocado em foco a necessidade de um uso mais racional e eficiente da energia.

Em seu trabalho, (GELLER, 2003) traz algumas elucidações sobre este crescimento do consumo de energia no Brasil. Ele afirma que o uso total de energia no Brasil cresceu cerca de 250% no período de 1975 a 2000, o que foi provocado, principalmente, pela rápida industrialização e pelos crescentes serviços energéticos residenciais e comerciais. Complementarmente, ele destaca que a nível mundial, o uso de energia aumentou dez vezes desde 1900, sendo a maior parte desta energia proveniente de fontes não renováveis.

Os dados apresentados na Tabela 1, que foi retirada do Anuário Estatístico de Energia Elétrica do ano de 2014, ressaltam as relações de consumo de energia entre os diferentes setores do país ao longo do tempo (ANUÁRIO..., 2014). Nota-se que em 2013, 48,3% do consumo energético no Brasil foi originário de edificações, se somados os setores Residencial, Comercial e de Poder Público, evidenciando assim a importância de se desenvolver soluções mais eficientes energeticamente nestas áreas, que incluem desde sistemas de iluminação e aquecimento até equipamentos eletrodomésticos.

Segundo a EPE (Empresa de Pesquisa Energética), empresa pública brasileira vinculada ao Ministério de Minas e Energia, eficiência energética significa gerar a mesma quantidade de energia com menos recursos naturais ou obter o mesmo serviço (”realizar

	2009	2010	2011	2012	2013	Δ% (2013/2012)	Part. % (2013)
Brasil	**384.306**	**415.683**	**433.034**	**448.171**	**463.335**	**3,4**	**100,0**
Residencial	100.776	107.215	111.971	117.646	124.896	6,2	27,0
Industrial	161.799	179.478	183.576	183.475	184.609	0,6	39,8
Comercial	65.255	69.170	73.482	79.226	83.695	5,6	18,1
Rural	17.304	18.906	21.027	22.952	23.797	3,7	5,1
Poder público	12.176	12.817	13.222	14.077	14.608	3,8	3,2
Iluminação pública	11.782	12.051	12.478	12.916	13.512	4,6	2,9
Serviço público	12.898	13.589	13.983	14.525	14.847	2,2	3,2
Próprio	2.319	2.456	3.295	3.354	3.372	0,5	0,7

Fonte: (ANUÁRIO..., 2014).

Tabela 1: Evolução nos consumos de energia elétrica (GWh) por classe no Brasil.

trabalho") com menos energia (ANUÁRIO..., 2014). Adicionalmente, (ENERGY..., 2015) define que um sistema é mais eficiente energeticamente se este oferece mais serviços com a mesma energia consumida ou então os mesmos serviços com menos energia consumida.

Segundo (LAMBERTS et al., s.d.), em 2001 o Brasil passou por uma crise energética devido ao grande crescimento no consumo e o pequeno crescimento da capacidade instalada. A preocupação com esta questão, ainda segundo (LAMBERTS et al., s.d.), colocou em foco a busca por eficiência energética não somente pelas questões econômicas, mas também pela necessidade de qualidade do ambiente para o usuário e a redução de impactos ambientais.

2.1.2 Conforto Térmico

De acordo com (FROTA; SCHIFFER, 2001), quando as trocas de calor entre o corpo humano e o ambiente ocorrem com esforço mínimo, a sensação do indivíduo é de conforto térmico e sua capacidade de trabalho, desse ponto de vista, é máxima. Também de acordo com (FROTA; SCHIFFER, 2001), se as condições térmicas do ambiente causam no indivíduo sensação de calor ou de frio, é porque o seu organismo está perdendo menos ou mais calor que o necessário para manter a temperatura corporal constante. Com isso, a manutenção da temperatura passa a ser atingida a partir de um esforço adicional que sempre representa sobrecarga, com queda do rendimento no trabalho, até o limite, sob condições de rigor excepcionais, perda total de capacidade para realização de trabalho e/ou problemas de saúde.

Em seu livro, (FANGER, 1970) diz que a motivação pelo conforto térmico consiste no desejo do homem de sentir-se termicamente confortável, além de se justificar também no ponto de vista da performance humana. Segundo a (NBR..., 2005) o conceito de conforto térmico é a "satisfação psicofisiológica de um indivíduo com as condições térmicas do ambiente". Já segundo a norma ISO 7730: "conforto térmico é o estado de alma que expressa satisfação com o ambiente térmico".

Para (SILVA, P. C. P. DA, s.d.) o conforto térmico não é um conceito exato, pois ele não implica em uma temperatura exata. Em outras palavras, a sensação de conforto térmico não está associada a uma temperatura específica, mas sim a diversos fatores. Estes podem

ser separados em fatores mensuráveis, como a temperatura do ar ou a umidade relativa, e fatores não mensuráveis, como o estado mental ou os hábitos do indivíduo.

Segundo (FROTA; SCHIFFER, 2001) o homem é um animal homeotérmico, tendo seu organismo mantido a uma temperatura interna sensivelmente constante, a qual é da ordem de 37^{o}C com limites muito estreitos, variando normalmente entre 36,1^{o}C e 37,2^{o}C. Cerca de 20% da energia que o organismo obtém através do metabolismo é transformada em potencialidade de trabalho, o que, termodinamicamente falando, significa que o rendimento do corpo humano é muito baixo. A parcela restante de 80% é transformada em calor que deve ser dissipado para que o organismo seja mantido em equilíbrio e, tanto o calor produzido quanto o dissipado, dependem da atividade que este desenvolve.

Sendo assim, o organismo humano de acordo com (FROTA; SCHIFFER, 2001) experimenta sensação de conforto térmico quando perde o calor que foi produzido pelo metabolismo, compatível com sua atividade, para o ambiente, sem ter que recorrer a nenhum mecanismo interno de termorregulação. A termorregulação, apesar de ser o meio natural do organismo de controlar a perda de calor, representa um esforço extra que por sua vez acarreta em uma queda na potencialidade de trabalho.

Vários estudos e experimentos já foram realizados em relação aos efeitos do conforto térmico na saúde e nas atividades realizadas pelo ser humano. (NELSON; NILSSON; HOPKINS, s.d.) realizaram estudos de laboratório para analisar a produtividade, fadiga e estado psicológico de um grupo de pessoas em camâras de testes com temperatura e umidade relativa controlada. Como resultado, verificou-se, entre outras questões, que em ambientes frios a produtividade aumenta e a fadiga é mais dificilmente sentida pelo corpo humano.

Já (SILVA, L. B. DA, s.d.) realizou uma pesquisa na qual foi analisada a relação entre produtividade e conforto térmico dos digitadores de um centro de processamento de dados e cobrança. Os resultados revelaram que as condições termoambientais, isto é, as condições térmicas do ambiente, estão entre os fatores que implicam na produtividade dos trabalhadores, sendo o desconforto térmico o fator que mais tem impacto na variabilidade da produtividade dos mesmos.

2.2 Internet das Coisas

No ano de 1991, Mark Weiser (WEISER, 1991) apresentou para o mundo pela primeira vez o conceito de Internet das Coisas, levantando uma discussão sobre a tendência das tecnologias para o que ele chamou de Computação Ubíqua, termo que descreve uma computação onipresente que é imperceptível para as pessoas. Estes computadores estariam interligados em uma rede, formando um sistema inteligente e onipresente que auxilia as pessoas no cotidiano sem que elas se deem conta.

Depois de alguns anos o acrônimo IoT, ou Internet das Coisas, seria então criado pelo pesquisador britânico Kevin Ashton, no início dos anos 2000 (”UMA..., 2013). O termo foi utilizado pela primeira vez por ele durante o desenvolvimento de um projeto integrando RFID (*Radio-Frequency Identification*) e a Internet, no laboratório de AutoID do Massachusetts Institute of Technology (MIT) em 1999.

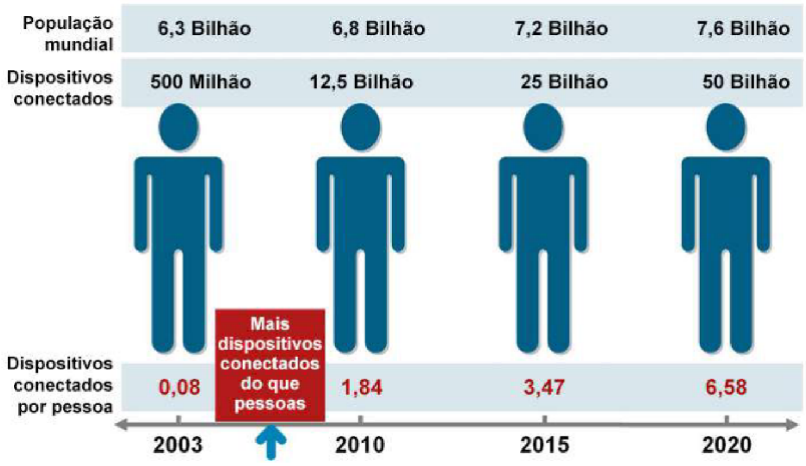
Segundo (SANTOS et al., 2016) a tecnologia IoT emergiu dos avanços da computação ubíqua e áreas correlacionadas como microeletrônica, comunicação e sensoriamento, tendo recebido bastante atenção tanto da academia quanto da indústria, devido ao seu potencial de uso nas mais diversas áreas das atividades humanas. (SANTOS et al., 2016) também diz que a conexão de objetos digitais com a rede mundial de computadores viabilizará, primeiro, controlar remotamente esses objetos e, segundo, permitirá que os

próprios objetos sejam acessados como provedores de serviços.

Apesar do termo Internet das Coisas já ser mundialmente difundido e estudado, não existe uma definição única universalmente aceita em relação ao seu significado. A Internet Architecture Board (IAB), por exemplo, afirma que o termo IoT denota uma tendência em que um grande número de dispositivos incorporados empregam serviços de comunicação oferecidos pelos protocolos da Internet (ARCHITECTURAL..., 2015). A IAB ainda afirma que muitos desses dispositivos, frequentemente chamados de "objetos inteligentes", não são operados diretamente por humanos, mas existem como componentes em edifícios ou veículos, ou estão espalhados no ambiente (ARCHITECTURAL..., 2015).

O dicionário Oxford (INTERNET..., 2019) define o termo IoT como: "A interconexão via Internet de dispositivos computacionais embarcados em objetos do cotidiano, permitindo que eles enviem e recebam dados". De acordo com a Cisco Internet Business Solutions Group (Cisco IBSG), no entanto, a IoT é simplesmente o momento em que mais "coisas ou objetos" se conectaram à Internet do que pessoas (EVANS, 2011).

Como pode ser observado na Figura 1, em 2003 havia aproximadamente 6,3 bilhões de pessoas vivendo no planeta e 500 milhões de dispositivos conectados à Internet. Com base na definição da Cisco IBSG, a IoT ainda não existia em 2003 porque o número de coisas conectadas era relativamente pequeno, já que dispositivos onipresentes, como smartphones, estavam recém sendo lançados (EVANS, 2011).



Fonte: Adaptado de (EVANS, 2011).

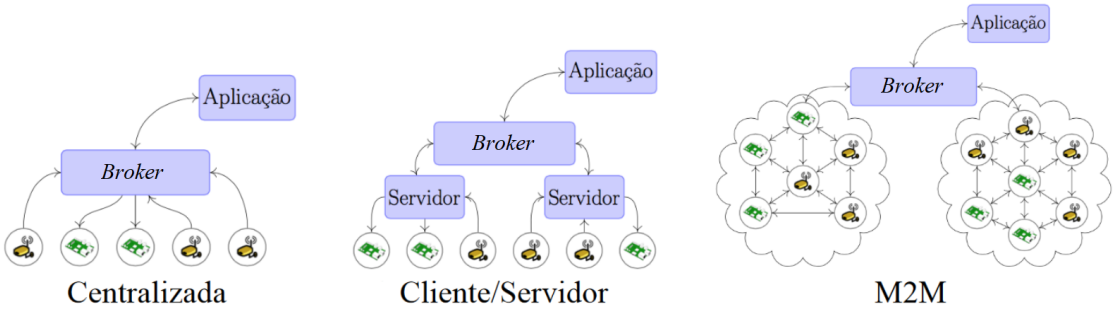
Figura 1: A Internet das Coisas "nasceu" entre 2008 e 2009.

A Cisco IBSG estima que a Internet das Coisas "nasceu" em algum momento entre os anos 2008 e 2009, quando o número de dispositivos conectados à Internet ultrapassou o número da população mundial. A Cisco IBSG ainda prevê que, até o ano de 2020, existirão 50 bilhões de dispositivos conectados à Internet (EVANS, 2011).

Segundo (ROSE; ELDRIDGE; CHAPIN, 2015) os produtos e serviços IoT atualmente estão sendo inseridos dentro de diversas áreas da indústria, como a tecnologia assistiva, saúde, automação residencial, agricultura, produção de energia, entre outros. De acordo com a Cisco, a previsão é de que o número de conexões de máquina-para-máquina (M2M do acrônimo em inglês *machine-to-machine*) aumente de 24% do total de dispositivos conectados à Internet em 2014 para 43% em 2019 (ROSE; ELDRIDGE; CHAPIN, 2015).

Em sua dissertação, (NEVES, s.d.) afirma que o paradigma da IoT se resume em uma rede heterogênea que integra coisas à Internet, o que proporciona uma mudança tanto na comunicação quanto nos serviços oferecidos via Internet. A troca de informação, centrada

na comunicação M2M entre as coisas, permite a coleta e o compartilhamento entre si de informações do mundo real, cooperação esta que possibilita o desenvolvimento de serviços personalizados (NEVES, s.d.).



Fonte: Adaptado de (MUSSIO; MAIA; LOPES, 2015).

Figura 2: Tipos de arquiteturas IoT.

A Figura 2 apresenta alguns exemplos de tipos de arquiteturas IoT envolvendo os dispositivos e a aplicação. A arquitetura centralizada é baseada unicamente no *Broker*, componente que irá distribuir as informações dos dispositivos para a aplicação. A arquitura Cliente/Servidor é baseada na centralizada, com exceção de que as informações são pré-processadas por servidores e por fim, a arquitetura M2M utiliza os próprios dispositivos como distribuidores de informações (MUSSIO; MAIA; LOPES, 2015).

2.2.1 *Smart Home*

Para (ROSE; ELDRIDGE; CHAPIN, 2015) a implementação em grande escala de dispositivos IoT promete transformar muitos aspectos da maneira como vivemos. Os novos produtos de IoT, como componentes de automação residencial e dispositivos de gerenciamento de energia, estão nos levando a uma visão do que é chamado de *Smart Home*, oferecendo assim mais segurança e eficiência energética (ROSE; ELDRIDGE; CHAPIN, 2015).

Segundo (DOMB, 2019) o termo *Smart Home* define uma residência que possui objetos, como por exemplo eletrodomésticos, iluminação, ar-condicionados, TVs, computadores, geladeiras e sistemas de segurança, que sejam capazes de se comunicarem entre si e serem controlados remotamente por horário, telefone, celular ou internet. Com uma ampla variedade de aplicações para o consumidor digital, existem três componentes essenciais em uma *Smart Home*: *hardware*, *software* e os protocolos de comunicação (DOMB, 2019).

Um sistema típico de *Smart Home*, de acordo com (DOMB, 2019), é equipado com um conjunto de sensores para medir as condições da casa, tais como: temperatura, umidade, iluminação e proximidade. Cada sensor se dedica a capturar uma ou mais medições, permitindo o armazenamento e visualização dos dados, para que assim o usuário possa consultá-los de qualquer lugar e a qualquer momento, o que é possibilitado através de um processador de sinais, uma interface de comunicação e um *host* em uma infraestrutura de nuvem (DOMB, 2019).

2.2.2 Protocolo de comunicação MQTT

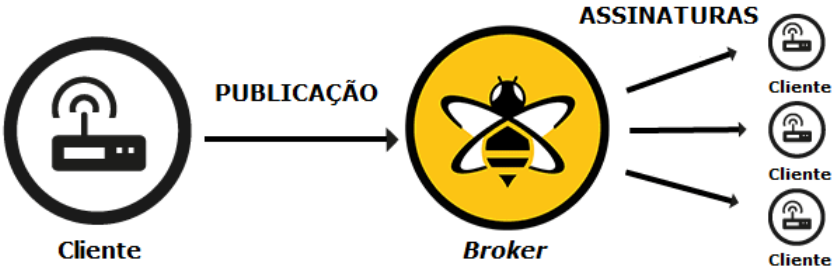
Em seu artigo (YUAN, 2017) afirma que, como a conexão com a Internet é um requisito para dispositivos IoT e o protocolo subjacente à esta é o TCP/IP (*Transmission Control*

Protocol / Internet Protocol), o MQTT (*Message Queue Telemetry Transport*), desenvolvido com base na pilha TCP/IP, acabou por tornar-se o padrão para comunicações de IoT. O protocolo de comunicação MQTT foi inicialmente criado pela IBM no final dos anos 90 e sua aplicação original era vincular sensores em oleodutos de petróleo a satélites (YUAN, 2017).

É um protocolo de mensagem com suporte para a comunicação assíncrona, isto é, uma comunicação que desacopla o emissor e o receptor da mensagem tanto no espaço quanto no tempo, sendo assim aplicável em ambientes com rede de conexão não confiável (YUAN, 2017). O protocolo MQTT se tornou oficialmente um padrão aberto OASIS (*Organization for the Advancement of Structured Information Standards*) no final de 2014, tendo assim suporte nas linguagens de programação populares com diversas implementações de *software* livre (YUAN, 2017).

De acordo com (YUAN, 2017) o MQTT é um protocolo de rede leve e flexível, as propriedades ideais para se desenvolver dentro de um sistema IoT. Por ser leve, o protocolo permite a implementação em um dispositivo altamente restrito em termos computacionais e em redes com largura de banda limitada e/ou alta latência. Já a sua flexibilidade faz com que ele suporte diversos cenários de aplicações para dispositivos e serviços de IoT (YUAN, 2017). Segundo (SKERRETT, 2015), diretor de marketing da fundação Eclipse: ”me parece que o MQTT se tornou o padrão a ser suportado por qualquer provedor sério de soluções para IoT”.

A comunicação entre os clientes no protocolo MQTT é feita no padrão de publicação e assinatura (Figura 3), também conhecido como *Pub/Sub*, diferindo da arquitetura tradicional de cliente e servidor em que o cliente se comunica diretamente com um ponto final (PUBLISH..., 2015). Na arquitetura *Pub/Sub*, os clientes, que podem ser publicadores e/ou assinantes, são desacoplados entre eles e nunca trocam mensagens diretamente, sendo necessário que as mensagens passem por um terceiro componente: o *Broker* (PUBLISH..., 2015). O trabalho deste componente é filtrar todas as mensagens recebidas pelos clientes que publicam e distribuí-las corretamente para os clientes que as assinam (PUBLISH..., 2015).



Fonte: Imagem adaptada de (MQTT..., 2015).

Figura 3: Arquitetura da comunicação por publicação e assinatura do protocolo MQTT.

Como (YUAN, 2017) explica, um cliente pode ser tanto um sensor IoT que executa leituras em campo quanto um aplicativo em um data center que processa dados IoT, enquanto que o *Broker* é um servidor que recebe e distribui as mensagens. As mensagens no MQTT são organizadas por tópicos: o cliente conectado ao *Broker* pode assinar qualquer tópico de mensagem já existente, assim como publicar mensagens em um tópico, existente ou não, enviando o conteúdo da mensagem junto do tópico de destino ao *Broker* (YUAN, 2017).

No protocolo MQTT, a palavra ”tópico” se refere a uma *string* no formato UTF-8 que o *Broker* usa para filtrar as mensagens para cada cliente conectado (MQTT..., 2019). A estrutura de um tópico, o qual é criado assim que é publicada a primeira mensagem endereçada a ele, consiste em um ou mais níveis hierárquicos, sendo cada nível de tópico separado por uma barra (”/”), como por exemplo: ”minha_casa/terreo/sala_de_estar/temperatura” (MQTT..., 2019).

Outra característica importante do protocolo de comunicação MQTT é a Qualidade de Serviço que, conhecido como QoS (*Quality of Service*), permite ao cliente escolher um nível de serviço adequado para sua lógica de aplicação e confiablidade de rede (QUALITY..., 2015). Existem três níveis de QoS que podem ser utilizados em serviços de publicação ou assinatura:

1. Nível 0 - No máximo uma vez;
2. Nível 1 - Ao menos uma vez;
3. Nível 2 - Exatamente uma vez.

No primeiro nível a mensagem é enviada apenas uma vez, sem garantia de recebimento do outro lado, enquanto que no segundo nível, adicionada uma mensagem de reconhecimento do outro lado, é garantido que a mensagem original será recebida, porém com chances de ser emitida repetidas vezes (QUALITY..., 2015). Já no terceiro nível se tem o maior nível de serviço do MQTT, no qual é garantido o recebimento da mensagem sem chances de se ter duplicatas, sendo este nível o mais seguro porém o mais lento em relação à comunicação na rede (QUALITY..., 2015). Portanto, por permitir a retransmissão de mensagens e garantir o recebimento destas, a Qualidade de Serviço facilita muito a comunicação em redes não confiáveis (QUALITY..., 2015).

2.2.3 Formato JSON

Na aplicação IoT desenvolvida por (DOMB, 2019), para se diminuir o uso da largura de banda, a comunicação foi feita através de mensagens no formato JSON. De acordo com sua página oficial, o JSON (*JavaScript Object Notation* - Notação de Objetos JavaScript) é uma formatação leve de troca de dados que usa convenções familiares às linguagens populares, como C++, C#, Java, JavaScript, Python, entre outras (INTRODUÇÃO..., s.d.). Basicamente, por ser um formato baseado em uma coleção de pares nome/valor, permite que sejam enviados múltiplos dados de forma estruturada em uma só mensagem de texto.

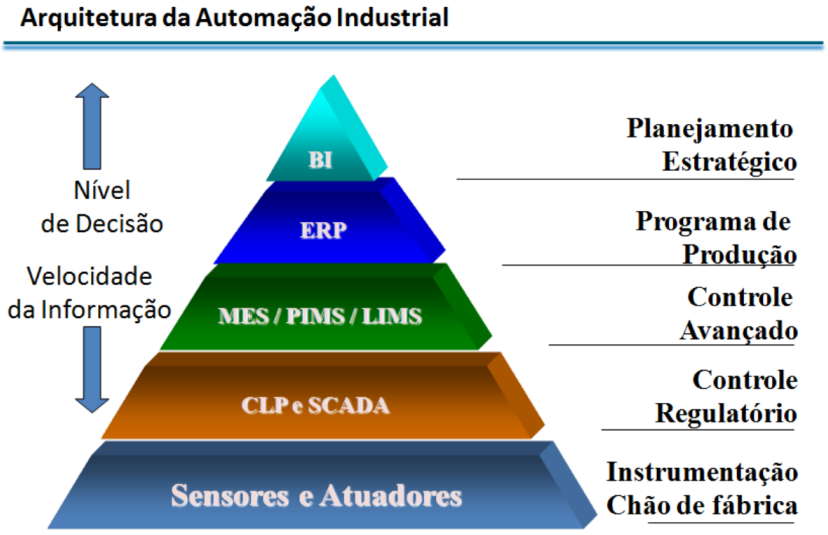
2.3 Sistema PIMS

De acordo com (ARAÚJO, s.d.) o sistema PIMS (*Plant Information Management System*) é caracterizado pela capacidade de coletar e centralizar dados de diferentes unidades da mesma planta em um único banco de dados, podendo armazená-los ao longo de muitos anos e disponibilizá-los de forma que o usuário possa desenvolver aplicações para o monitoramento e análise do processo de produção. Segundo (PIMS..., 2015) podem ser apresentadas como principais características dos sistemas PIMS:

- Coleta de dados de sistemas de chão de fábrica e sistemas corporativos;
- Alta e eficiente compressão dos dados;

- Capacidade de armazenamento histórico superior a 10 anos;
- Alta velocidade de resposta a consultas;
- Ferramentas clientes de utilização fácil e intuitiva, tais como telas sinóticas, relatórios, KPIs (Indicador-chave de desempenho), portal web, entre outros.

Considerando a Pirâmide da Automação apresentada na Figura 4, é possível dizer que o PIMS está no intermédio da interação e troca de dados entre o campo e as tomadas de decisões gerenciais (ARAÚJO, s.d.). Utilizando-se um sistema PIMS, segundo (CARVALHO et al., 2005), pode-se visualizar tanto os dados de tempo real quanto os dados históricos da planta a partir de uma estação de trabalho, seja montando tabelas, gráficos de tendência ou relatórios dinâmicos, concentrando a informação e possibilitando uma visão unificada de todo o processo produtivo.



Fonte: (ARAÚJO, s.d.).

Figura 4: Pirâmide da Automação.

De acordo (CARVALHO et al., 2005) são muitos os benefícios gerados pela implantação de um sistema PIMS, como a centralização dos dados de processo, a visualização em tempo real do processo produtivo e o armazenamento de dados históricos, que pode chegar até 15 anos de dados devido ao seu algoritmo de compressão. Sendo assim, tanto o engenheiro de processo quanto o operador da planta pode encontrar respostas para o comportamento positivo ou negativo do processo, de forma que, alcançando o pleno conhecimento da planta, este pode atuar em busca dos melhores resultados (CARVALHO et al., 2005).

Uma das características mais importantes do sistema PIMS, segundo (CARVALHO et al., 2005), é a sua grande capacidade de compressão de dados históricos sem grandes perdas das informações, tendo tipicamente uma relação de compressão da ordem de 10:1, sendo razões de 20:1 comuns também. Ao invés de amostrar o dado em intervalos fixos, sendo a única maneira de melhorar a compressão aumentando o período de amostragem, os sistemas PIMS amostram a curva nos pontos em que existem mudanças significativas acontecendo, conseguindo assim uma alta taxa de compressão sem perda da qualidade do dado (CARVALHO et al., 2005).

3 METODOLOGIA

O objetivo do presente trabalho é desenvolver uma arquitetura de baixo custo para um cenário de Automação Residencial que, trazido para o contexto da Internet das Coisas, representa um projeto de *Smart Home*. Serão efetuadas leituras de temperatura, umidade e iluminação, seguindo casos típicos de monitoramento de dados em projetos de Automação Residencial. Estes dados são essenciais para se desenvolver um sistema de racionamento de energia inteligente, o que é importante dada a preocupação atual com a eficiência energética em edificações e residências. Além disso, com a coleta destes dados também é possível manter constante o conforto térmico nos ambientes, resultando em uma maior produtividade e qualidade de vida para o usuário.

Assim como foi dito na Seção 2.1, quanto mais efetiva a integração entre variados sistemas maiores serão os benefícios obtidos através do projeto de automação. A arquitetura aqui proposta é capaz de:

1. Ler constantemente os dados de duas placas, cada uma equipada com sensores de temperatura, umidade e iluminação;

2. Enviar estes dados para um *Broker* MQTT na nuvem a cada leitura;

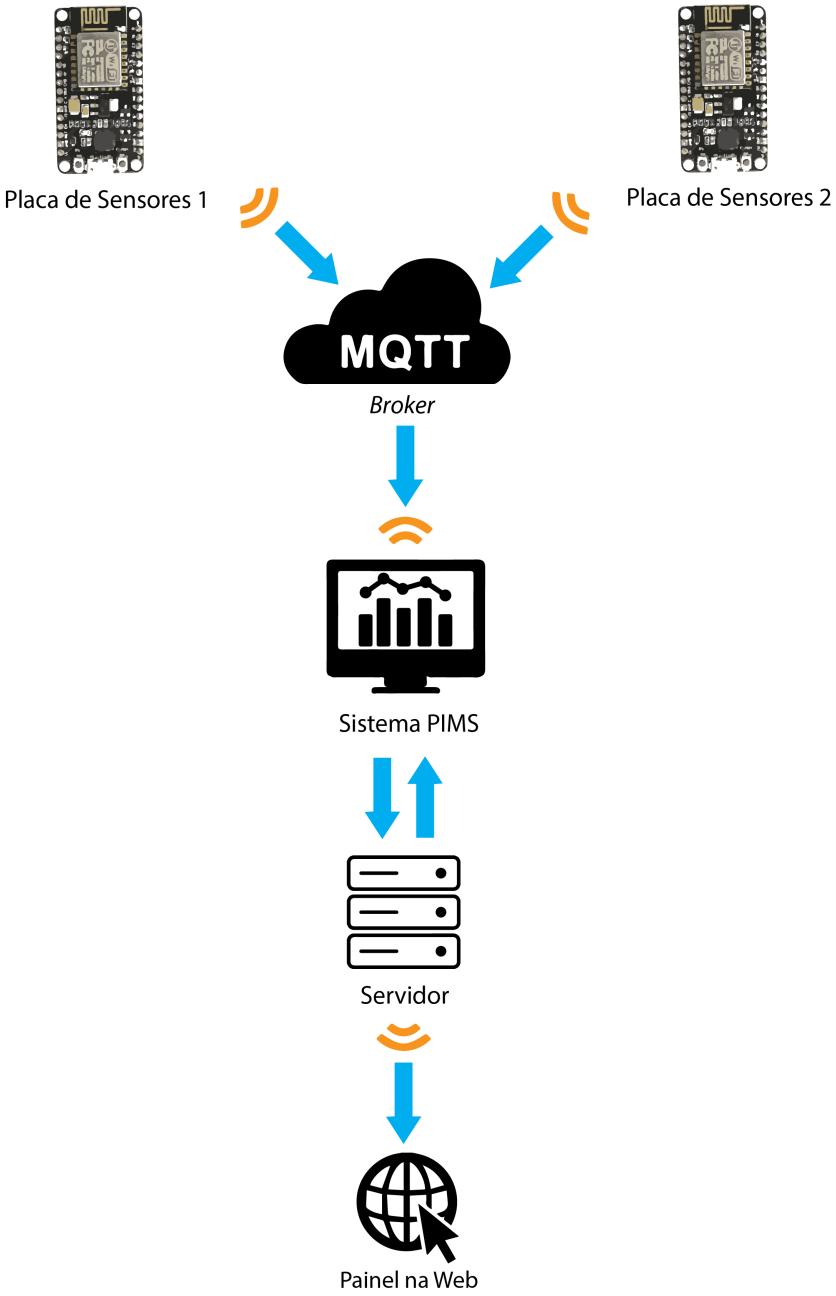
3. Receber e armazenar estes dados em um sistema PIMS conectado a um servidor remoto;

4. Apresentar on-line, através deste servidor, os dados históricos e atuais em um painel na Web para visualização.

3.1 Arquitetura IoT

O fluxograma da Figura 5 ilustra a arquitetura que foi desenvolvida, permitindo a comunicação entre as placas de sensores, o *Broker* MQTT na nuvem e o sistema PIMS no servidor remoto, apresentando por fim os dados em um painel na Web. Como não há necessidade de comunicação entre os dispositivos, a arquitetura foi construída em cima do modelo centralizado da Figura 2, no qual os dispositivos são representados pelas placas e a aplicação é representada pelo sistema PIMS junto do painel.

A integração entre estes sistemas permite que os dados sejam armazenados e processados remotamente, fazendo com que não haja necessidade de gastos com banco de dados ou processamento dentro do âmbito residencial. Outro benefício é que, como o *Broker* está localizado na nuvem, são poucas as chances de ocorrerem problemas de instabilidade de conexão com o mesmo.



Fonte: Autor.

Figura 5: Arquitetura IoT.

Para simular um cenário de Automação Residencial, uma das placas foi alocada no quarto e a outra na sala de uma residência, ambas executando constantemente as leituras de temperatura, umidade e luminosidade. As placas, conectadas por *Wi-Fi* na Internet da casa, enviam os valores lidos pelos sensores ao *Broker* através de mensagens endereçadas a tópicos específicos.

O sistema PIMS, conectado a um servidor remoto, quando devidamente configurado se conecta ao *Broker* e assina os tópicos das leituras das placas. Dessa forma o PIMS recebe estes dados sempre que são publicados pelas placas e os armazena como dados históricos para posteriores visualizações e análises.

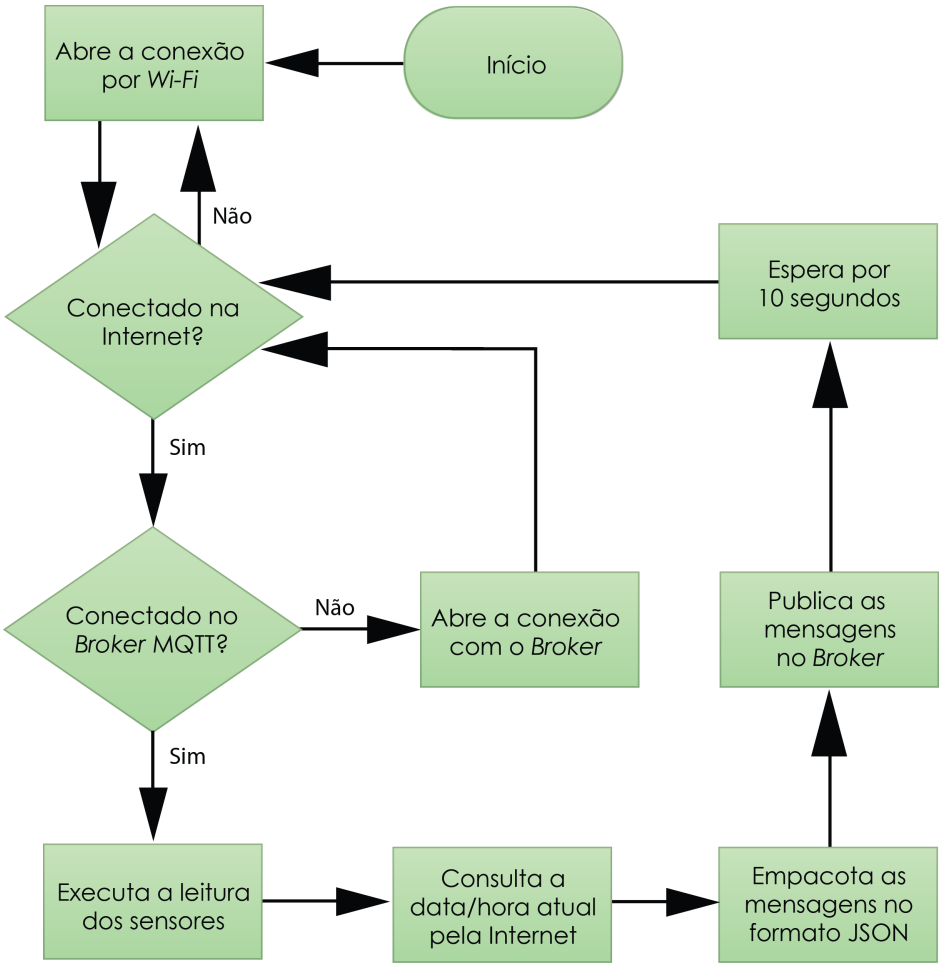
Por fim, o servidor do PIMS também hospeda um portal na Web que contém um painel

com gráficos e indicadores relacionados às leituras dos sensores on-line. Este painel pode ser acessado de qualquer lugar pelo usuário final através de um computador ou celular, desde que tenha conexão com a Internet.

3.2 Algoritmo das Placas

O algoritmo embarcado dentro das placas deve, essencialmente, executar a leitura dos sensores de temperatura, umidade e luminosidade e enviar os valores lidos para o *Broker* MQTT na nuvem. Este processo de coleta e envio de dados deve ser repetido indefinidamente e, caso ocorra uma falha na conexão com a Internet ou com o *Broker*, ele deve ser capaz de parar o processo até se restabelecer a conexão e com isso iniciá-lo novamente.

O fluxograma da Figura 6 demonstra o algoritmo que foi utilizado como base para se elaborar o programa que fica rodando no microprocessador de cada placa. Daqui pra frente, nesta seção, será feita uma descrição do funcionamento deste algoritmo que foi implementado nas duas placas, as quais têm comportamento semelhante, diferindo apenas no endereçamento das mensagens.



Fonte: Autor.

Figura 6: Algoritmo das placas de sensores.

Quando a placa é ligada ocorre um processo de inicialização no qual são configura-

das as variáveis de conexão com a Internet e com o *Broker*, assim como as variáveis de leitura dos sensores. Após isto é feita a primeira tentativa de abertura da conexão por *Wi-Fi* da placa com um roteador ligado à Internet, entrando-se assim no ciclo principal do programa.

No ciclo principal, de início, o estado da conexão da placa com a Internet é testada e caso a conexão não esteja estabelecida, esta é reaberta até que esteja. Após estabelecida com sucesso a conexão com a Internet, desta vez é testado o estado da conexão da placa com o *Broker* MQTT e, caso ela não esteja estabelecida, é feita uma primeira tentativa de abertura da conexão com o *Broker*.

Antes de ser testado novamente o estado da conexão com o mesmo para seguir-se em frente no ciclo, primeiro é testado o estado da conexão com a Internet pois num ponto de vista hierárquico a placa deve primeiro estar conectada à ela para depois se conectar ao *Broker* que está alocado na nuvem. Dessa forma uma nova tentativa de conexão com o *Broker* só é feita se a conexão com a Internet estiver efetivamente estabelecida, previnindo-se que o programa fique trancado caso se perca a conexão com a mesma durante este processo.

Feita a conexão com o *Broker*, o programa parte para a leitura dos dados dos sensores de temperatura, umidade e luminosidade, os quais têm seu valores guardados em variáveis internas. Na próxima etapa a placa consulta, através da Internet, a data/hora (variável padrão que contém o conjunto de data e horário) atual e a guarda em uma variável interna para ser posteriormente enviada juntamente das leituras dos sensores. Isto é feito para se garantir que, mesmo que ocorram atrasos ou falhas na comunicação, as leituras sejam sempre corretamente interpretadas, em relação ao momento em que foram efetuadas, quando recebidas na outra ponta da arquitetura.

Após isto entra-se na etapa em que são montadas as mensagens de texto que serão enviadas ao *Broker*, na qual os valores lidos pelos três sensores são transformados em três mensagens distintas enderaçadas cada uma a um tópico diferente: um para a temperatura, um para a umidade e um para luminosidade. Para se diminuir o uso da largura de banda por parte dos clientes e se simplificar a estrutura de tópicos no *Broker*, cada mensagem contém um conjunto de informações relacionadas à leitura do sensor, que por sua vez são empacotadas no formato JSON.

Este conjunto de informações contido em cada mensagem representa, na forma de texto, os seguintes dados: o valor lido pelo sensor, o momento em que foi lido (através do valor de data/hora) e um fator de qualidade binário (0 ou 1) que indica se ocorreu erro de leitura. A estrutura resultante das mensagens neste formato serão apresentadas na Figura 8 da Seção 4.2.

Com as mensagens construídas, na etapa seguinte elas são publicadas no *Broker* em seus devidos tópicos, os quais levam em seus nomes a identificação da placa e o sensor lido, como por exemplo ”Placa1/Temperatura”. Após isto, o programa fica em modo de espera por 10 segundos na etapa final antes que seja recomeçado o ciclo principal e, caso as conexões ainda estejam estabelecidas, no ciclo seguinte o programa vai direto para a etapa de leitura dos sensores e assim indefinidamente, efetuando uma leitura a cada aproximadamente 10 segundos até que se ocorra algum erro de conexão.

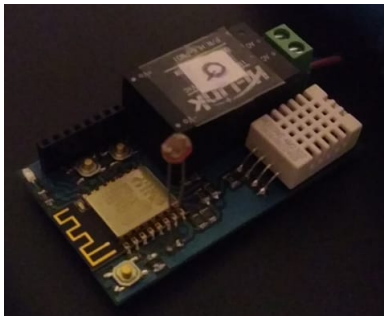
Como as medições de temperatura, umidade e luminosidade variam pouco em relação ao tempo no contexto residencial, esta frequência de coleta de dados de 10 segundos é bem satisfatória para estes casos. Espera-se que o algoritmo, quando implementado nas placas, faça-as executarem este ciclo de coleta e envio de dados de forma repetitiva e independente de qualquer interação humana para o seu correto funcionamento.

4 RESULTADOS E DISCUSSÃO

Como prova de conceito foi feito um experimento na casa do autor com a execução prática da arquitetura proposta neste trabalho, simulando um projeto de Automação Residencial inserido no contexto da Internet das Coisas. Este capítulo destina-se a apresentar as ferramentas utilizadas e os resultados obtidos através deste experimento. Na primeira seção serão apresentadas as placas montadas e seus componentes, assim como alguns detalhes dos programas que foram implementados nelas. Na sequência será abordado o *Broker* MQTT, localizado na nuvem, que foi utilizado neste experimento. Por fim, nas seções seguintes, serão demonstrados os resultados obtidos com o sistema PIMS utilizado e com o painel que está ligado à Internet.

4.1 Placas de Sensores

Para montar as placas de sensores foram utilizados 4 componentes em cada: um módulo ESP8266, um sensor LDR (*Light Dependent Resistor* - Resistor Dependente da Luz), um sensor DHT22 e um pequeno conversor de tensão. O módulo ESP8266 foi escolhido como microcontrolador pelo seu baixo custo e também por já conter um módulo *Wi-Fi* para se conectar à Internet. Para a leitura de luminosidade, o sensor LDR é também a escolha mais barata, assim como o sensor DHT22, que é o responsável pela leitura de temperatura e umidade. O conversor CA-CC por sua vez é utilizado para que a placa possa ser conectada diretamente na rede elétrica.



Fonte: Autor.

Figura 7: Placa com o módulo ESP8266 e os sensores LDR e DHT22.

O sensor LDR é composto por um resistor fotossensível, como já diz o nome, que altera sua resistência em relação à luminosidade captada por este e tem seu valor lido através do conversor AD do módulo ESP8266. Já o sensor DHT22, composto por um sensor capacitivo de umidade junto de um termistor, tem seus sinais lidos através de entradas

digitais. O conversor alimenta a placa com uma tensão de 5V com corrente contínua e tem como entrada uma tensão de 110V com corrente alternada provieniente da rede elétrica residencial.

Foram montadas duas placas, uma para ficar no quarto e outra para ficar na sala da residência em questão. A Figura 7 apresenta uma foto da placa montada que ficou alocada no quarto.

O algoritmo apresentado na Seção 3.2 foi implementado na linguagem C++ através da IDE (*Integrated Development Environment* - Ambiente de Desenvolvimento Integrado) do Arduino, um *software* livre que também funciona com o ESP8266 se corretamente configurado.

O programa desenvolvido segue fielmente o algoritmo em questão e resultou em duas versões diferentes, uma para a placa do quarto e outra para a placa da sala. Algumas bibliotecas tiveram que ser utilizadas no programa para possibilitar a execução de todas as etapas do processo: para a conexão por *Wi-Fi* utilizou-se a biblioteca ”ESP8266WiFi.h”, para a conexão com o *Broker* utilizou-se a biblioteca ”PubSubClient.h”, para a leitura do sensor DHT22 foi utilizada a biblioteca ”DHT.h” e para a consulta da data/hora a biblioteca ”time.h”.

A biblioteca do sensor DHT22 já faz por si só o tratamento dos dados lidos por este e os apresenta nas seguintes unidades de engenharia: ^{o}C para a temperatura e % para a umidade relativa. A leitura do sensor LDR, no entanto, resulta em algum valor entre 0V e 5V, sendo necessário um tratamento adicional para se transformar este dado lido em um valor na unidade de lux (densidade de intensidade luminosa). Isto se deu por meio de uma aproximação linear do comportamento exponencial que este tipo de sensor naturalmente tem em relação à intensidade luminosa, relacionando o valor lido pelo sensor LDR com a medida em lux captada por outro sensor posicionado no mesmo local, sendo neste caso utilizado o sensor de luminosidade de um smartphone.

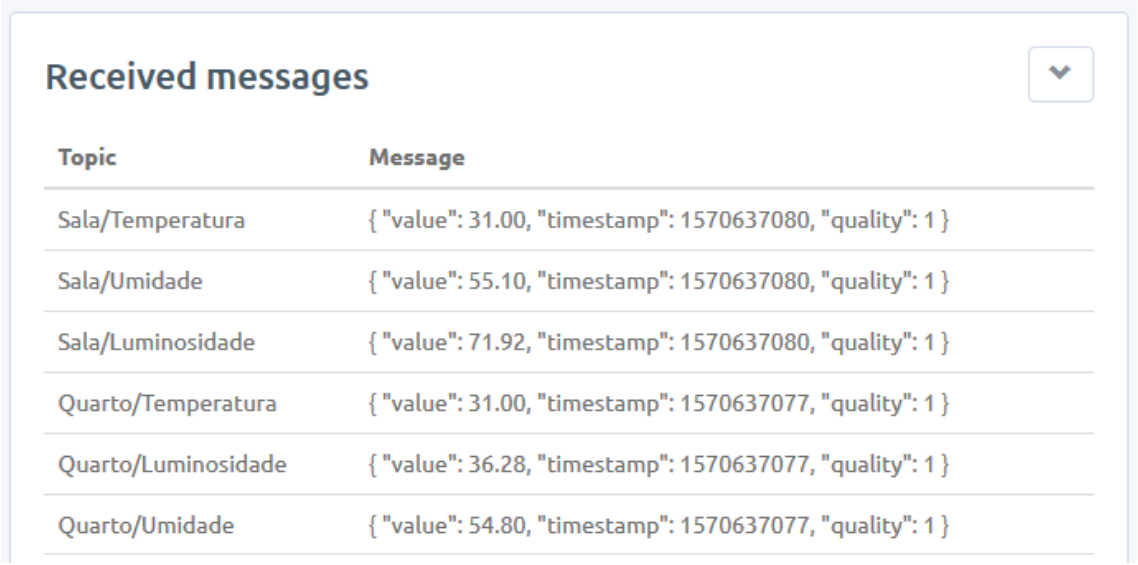
Em relação à conexão com o *Broker*, cada código define uma identidade única diferente para cada placa e as mensagens são publicadas por estas com qualidade de serviço de índice zero, que é o máximo suportado pela biblioteca. Os códigos das duas versões do programa em questão estão apresentados detalhadamente no Apêndices A e B e devem ser consultados se desejado um total entendimento do processo de leitura dos sensores e da comunicação das placas com o *Broker*.

4.2 *Broker* MQTT

Seguindo a construção da arquitetura IoT proposta para este experimento, foi utilizado um *Broker* alocado na nuvem através do portal CloudMQTT. O CloudMQTT é um *Broker* MQTT distribuído globalmente pela Internet, permitindo ao usuário criar e configurar um *Broker* próprio de forma simples e automatizada. Existe nele um plano grátis que suporta no máximo 5 conexões simultâneas de clientes por *Broker*, o que é suficiente para a aplicação em questão pois esta terá apenas 3 clientes: as duas placas e o sistema PIMS.

Na janela configuração do *Broker* criado é possível se consultar o endereço gerado, assim como as diferentes portas de conexão, o usuário e a senha. Estes dados são utilizados pelos clientes na hora de se conectar ao *Broker*, os quais devem ter uma identidade única cada. A identidade de cliente definida para a placa do quarto foi ”ESP_Quarto” e para a placa da sala ”ESP_Sala”.

Dentro do portal CloudMQTT também pode-se consultar as mensagens que estão sendo recebidas pelo *Broker* em tempo de execução, o que está demonstrado na Figura



Fonte: Autor.

Figura 8: Mensagens publicadas no *Broker* no formato JSON.

8. Na figura em questão a primeira coluna apresenta o tópico da mensagem e a segunda coluna o corpo da mensagem e, como pode se observar, existem três tópicos para cada ambiente da casa, sendo um para cada sensor. Também pode se observar a estrutura padrão das mensagens que estão no formato JSON: o campo "*value*" representa o valor lido pelo sensor, o campo "*timestamp*" (equivalente em inglês para estampa de tempo) representa a data/hora da leitura e o campo "*quality*" representa o fator de qualidade desta.

O fator de qualidade tem seu valor nulo quando uma falha na leitura é detectada pela placa e a data/hora é enviada ao *Broker* no formato UNIX, o qual é um valor inteiro padronizado globalmente que representa a quantidade de segundos transcorridos desde 1º de janeiro de 1970 no fuso horário UTC (*Coordinated Universal Time* - Tempo Universal Coordenado). Este conjunto de dados empacotados segue o padrão de variáveis do tipo séries temporais, as quais são convencionalmente utilizadas em sistemas de informação como o PIMS e em sistemas de controle como os supervisórios.

4.3 Sistema PIMS

Com as placas de sensores em funcionamento e o *Broker* MQTT devidamente configurado, resta implementar o terceiro cliente do *Broker*: o sistema PIMS. Para isto foi utilizado o EPM (*Elipse Plant Manager*), sistema PIMS da empresa brasileira Elipse Software. Este foi o *software* escolhido pois ele pode ser baixado gratuitamente na versão de demonstração, a qual tem como limitação a criação de no máximo vinte *tags* (variáveis), que é mais do que suficiente para o experimento. Além disso no EPM é possível receber dados de variáveis por meio de uma interface de comunicação para o protocolo MQTT, a qual foi desenvolvida pelo autor durante o período em que trabalhou como estagiário nesta empresa. Complementarmente, foi utilizado um servidor remoto dedicado da Elipse Software para se receber e armazenar estes dados 24/7 (24 horas por dia, 7 dias por semana).

Primeiramente foi configurada a interface de comunicação MQTT do EPM, que é a

ferramenta responsável pela conexão com o *Broker* e a assinatura nos tópicos. A identidade de cliente definida para o PIMS foi ”ecc_user” e a interface também foi configurada para que a estampa de tempo recebida seja interpretada à partir do formato UNIX e o fator de qualidade seja interpretado à partir de valores binários.

Address | Queries

Insert | Delete

Name	Query	Topic	Value Field	Timestamp Field	Quality Field
SLuminosidade	Query	Sala/Luminosidade	value	timestamp	quality
SUmidade	Query	Sala/Umidade	value	timestamp	quality
STemperatura	Query	Sala/Temperatura	value	timestamp	quality
QLuminosidade	Query	Quarto/Luminosidade	value	timestamp	quality
QUmidade	Query	Quarto/Umidade	value	timestamp	quality
QTemperatura	Query	Quarto/Temperatura	value	timestamp	quality

Fonte: Autor.

Figura 9: Assinaturas nos tópicos de leitura dos sensores.

Tendo a conexão do PIMS com o *Broker* sido estabelecida através desta interface de comunicação, é possível assinar os tópicos de interesse para receber os dados das placas. As assinaturas foram feitas então, com qualidade de serviço um, nos seis tópicos que representam as leituras dos sensores localizados na residência em questão.

Esta interface do EPM suporta o recebimento de dados empacotados no formato JSON, no qual cada campo do conjunto, após ser configurado corretamente, define um parâmetro da variável de leitura (valor, estampa de tempo e qualidade). A Figura 9 demonstra as assinaturas feitas nos tópicos de leitura dos sensores, assim como o endereçamento de cada campo das mensagens em relação aos parâmetros equivalentes das variáveis.

Name: etcc* Description: all items EU: All Domain: All Filter

Drag a column header here to group by that column.

Name	Description	EU	Data...	RT	REC	Compre...	RT-Value	RT-Timestamp	RT-Q...	Data Server	Data Address
Type a name here to add...				☑	☑	☑					
ETCC_QUmidade	Umidade relativa do quarto.	%	Float	☑	☑	☐	55.7	2019-10-07T14:01:13	Good	ECC_EPM	ETCC_MQTT/Query.QUmi...
ETCC_QLuminosidade	Iluminância do quarto.	Lux	Float	☑	☑	☐	129.02	2019-10-07T14:01:13	Good	ECC_EPM	ETCC_MQTT/Query.QLumi...
ETCC_QTemperatura	Temperatura do quarto.	°C	Float	☑	☑	☐	29.7	2019-10-07T14:01:13	Good	ECC_EPM	ETCC_MQTT/Query.QTem...
ETCC_SUmidade	Umidade relativa da sala.	%	Float	☑	☑	☐	54.9	2019-10-07T14:01:11	Good	ECC_EPM	ETCC_MQTT/Query.SUmid...
ETCC_SLuminosidade	Iluminância da sala.	Lux	Float	☑	☑	☐	131.8	2019-10-07T14:01:11	Good	ECC_EPM	ETCC_MQTT/Query.SLumi...
ETCC_STemperatura	Temperatura da sala.	°C	Float	☑	☑	☐	30.2	2019-10-07T14:01:11	Good	ECC_EPM	ETCC_MQTT/Query.STemp...

Fonte: Autor.

Figura 10: Variáveis de leitura dos sensores no PIMS.

Os resultados da configuração feita podem ser vistos na Figura 10 e na Figura 11. A primeira apresenta uma tabela com as variáveis criadas para cada sensor e seus parâmetros, tais como nome, descrição e unidade de engenharia, assim como o valor, estampa de tempo e qualidade da última atualização de cada um. A segunda apresenta os gráficos que foram montados no PIMS para se visualizar os dados históricos de cada sensor com detalhes (curva azul para o quarto e curva laranja para a sala), sendo que nesta imagem são visualizadas as leituras feitas pelos sensores do dia 24 até o dia 29 de setembro de 2019.

Fonte: Autor.

Figura 11: Gráficos dos dados históricos dos sensores no PIMS.

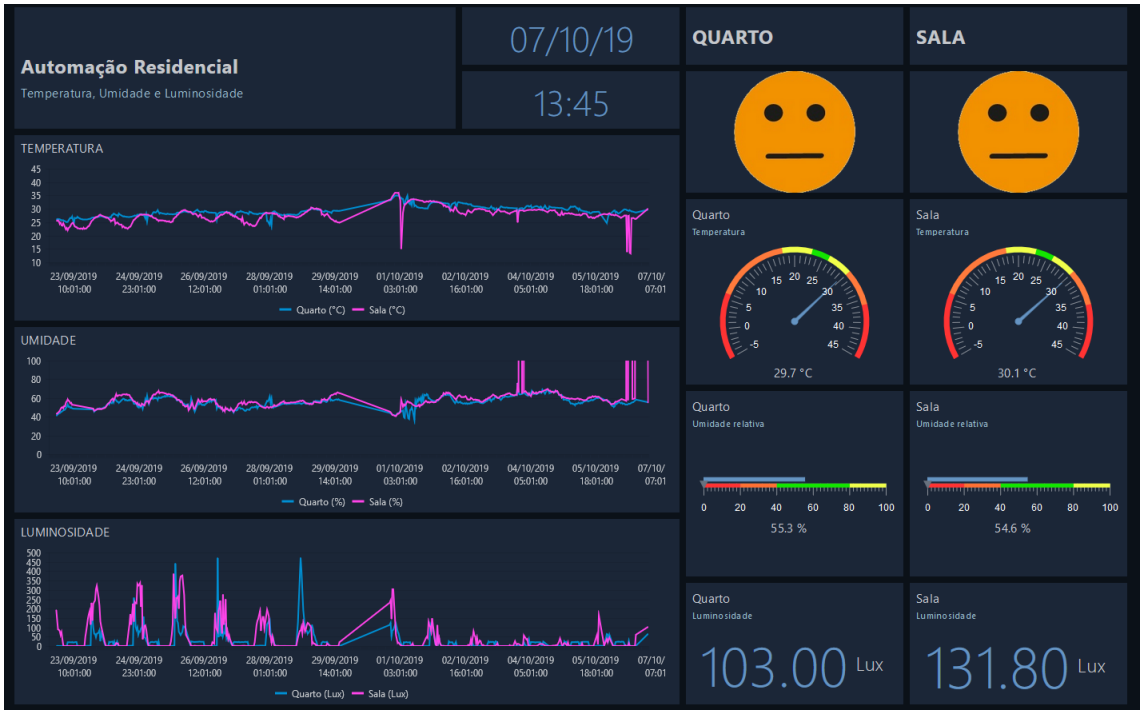
A Figura 11 também demonstra que os dados lidos são fiéis à realidade quando interpretados no contexto da residência em questão. Por exemplo, no gráfico de luminosidade as curvas são como esperado, pois tanto no quarto quanto na sala observa-se a dinâmica de variação da luz em relação ao dia e à noite. Considerando que a sala da residência recebe bastante luz solar no dia-a-dia, é possível observar também que a oscilação da curva de temperatura da sala acompanha a curva de luminosidade da mesma, o que faz sentido pois a temperatura deste ambiente é bastante influenciada pela incidência de luz solar.

4.4 Painel na Web

O EPM tem incluso no seu pacote de produto o *software* EPM Portal, o qual permite ao usuário utilizar um portal projetado para navegadores para se criar painéis de controle contendo as variáveis previamente definidas no PIMS. Este portal pode ser utilizado em um ambiente local, em uma rede off-line ou em uma rede on-line, dependendo de como este é hospedado. Como o servidor do PIMS neste caso é um servidor remoto dedicado conectado à Internet, este foi utilizado para hospedar o portal de forma que possa ser acessado pelo usuário através da Internet.

Foram criados então dois painéis com as leituras dos sensores: uma versão padrão (Figura 12) e uma outra versão simplificada para visualização em smartphone (Figura 14). A versão padrão do painel contém o horário e data atuais, os gráficos mostrando os dados históricos dos sensores, os valores atuais de cada sensor e um indicativo de conforto térmico para cada ambiente da residência.

Nos gráficos do painel o usuário tem a liberdade de alterar o eixo temporal para o período que quiser, sendo que o período demonstrado na Figura 12 vai do dia 23 de setembro até o dia 7 de outubro de 2019. Os valores atuais dos sensores de temperatura e umidade são apresentados na forma de indicadores visuais, os quais contém diferentes colorações para cada faixa de medida. A faixa verde em cada um determina os valores que estão inseridos dentro do conforto térmico baseado na norma ISO 9241, a qual estabelece



Fonte: Autor.

Figura 12: Painel na Web com os gráficos temporais, valores atuais dos sensores e indicativos de conforto térmico.

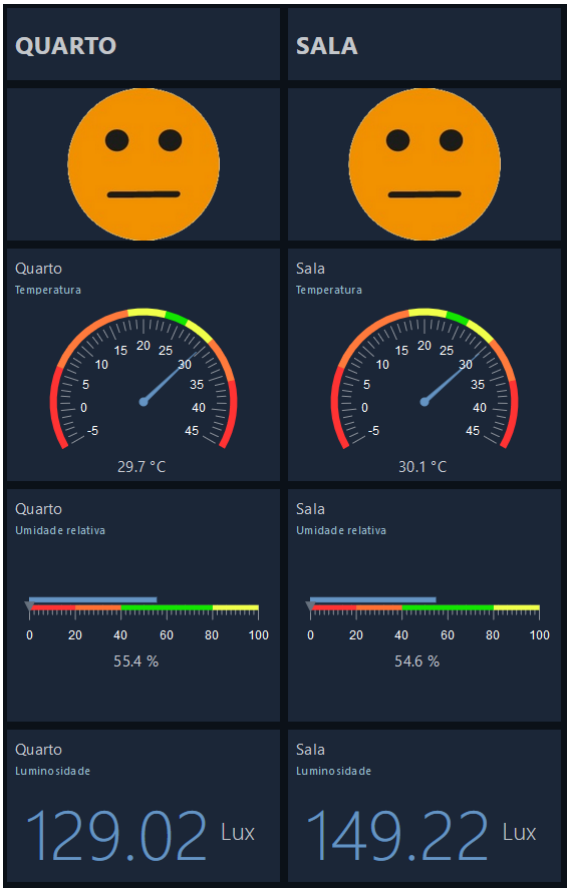
que o ideal é manter a umidade relativa entre 40% e 80% e a temperatura entre 23°C e 26°C no inverno, que foi o caso utilizado como referência como pode ser observado na figura em questão.

Fonte: Autor.

Figura 13: Representações do indicativo de conforto térmico.

Os indicativos de conforto térmico, por sua vez, são representados por imagens caricatas para fácil visualização e podem ser de três tipos: uma cara feliz na cor verde que representa o conforto térmico ideal, uma cara neutra na cor laranja que representa um conforto térmico razoável ou uma cara triste na cor vermelha que representa um conforto térmico ruim (Figura 13). O primeiro tipo é mostrado quando tanto a temperatura quanto a umidade estão dentro de suas faixas verdes, o terceiro tipo ocorre quando qualquer um dos indicadores está dentro da faixa vermelha e o segundo tipo é mostrado para o resto dos casos. As variáveis que definem os indicativos de conforto térmico foram desenvolvidas dentro do EPM na linguagem de programação Python (Apêndice C).

Para demonstrar a capacidade de armazenamento do sistema PIMS foi tirada outra foto do painel na Web, desta vez no dia 22 de novembro de 2019, a qual está apresentada na Figura 15. Nesta pode-se visualizar nos gráficos os dados históricos dos sensores do

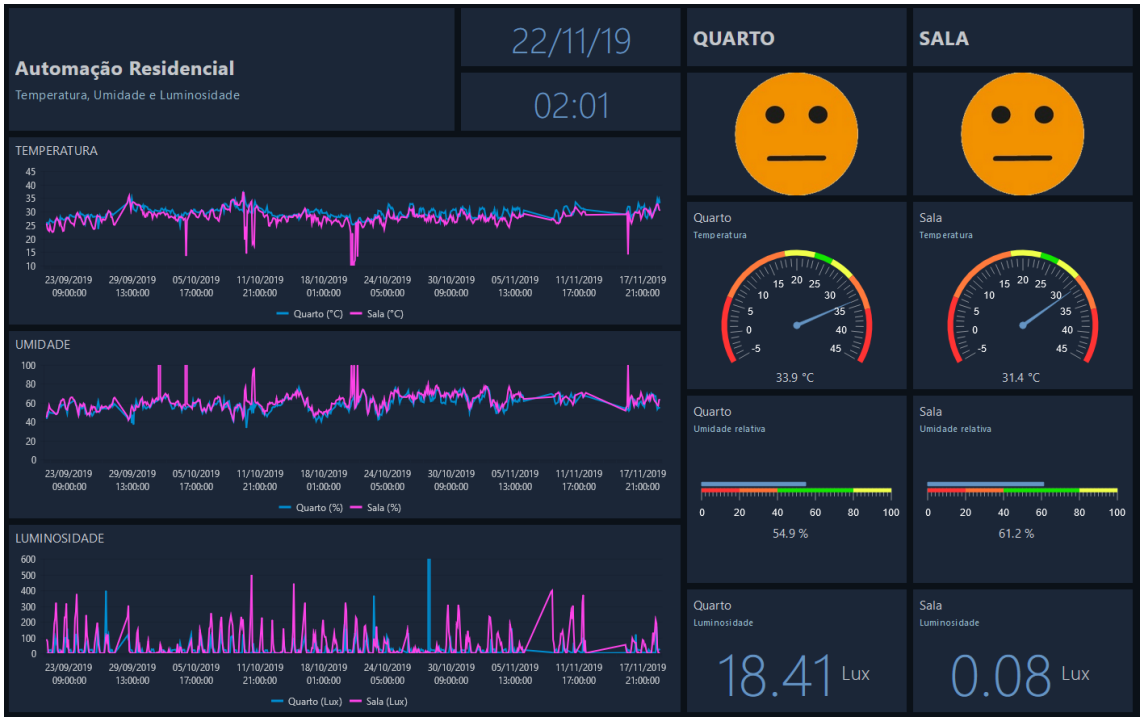


Fonte: Autor.

Figura 14: Versão do painel para celular apenas com os valores atuais dos sensores e indicativos de conforto térmico.

dia 23 de setembro até o dia 22 de novembro de 2019, sendo no total quase dois meses de dados armazenados, podendo ainda assim continuar gravando esses dados durante muitos anos.

Em relação à algumas partes dos gráficos na Figura 15 em que as curvas são lineares, estas representam períodos em que o servidor do PIMS estava desconectado. Devido aos gráficos serem apresentados na forma interpolativa (ligando um ponto a outro), quando este tipo de caso ocorre eles ligam automaticamente o último ponto gravado antes de o servidor cair com o primeiro ponto gravado assim que ele volta ao ar, o que explica essas faixas que são visualmente apresentadas de forma linear mas que não representam os valores lidos de fato.



Fonte: Autor.

Figura 15: Painel no dia 22 de novembro de 2019.

5 CONCLUSÃO

Monitorar os indicadores de temperatura, umidade e luminosidade é essencial para se implementar um sistema de Automação Residencial que vise proporcionar conforto térmico e eficiência energética dentro dos ambientes em que são aplicados. O objetivo do trabalho de se desenvolver uma arquitetura de baixo custo para ser aplicada a um projeto de Automação Residencial, inserido no contexto da Internet das Coisas, foi atingido com sucesso.

Fazendo-se uso de equipamentos baratos que são facilmente encontrados no mercado, junto de um servidor na nuvem sem custo e de *softwares* livres, foi possível realizar em dois ambientes diferentes as leituras de temperatura, umidade e luminosidade e enviá-las através da Internet a um sistema PIMS para armazenamento e visualização. Ao longo dos dois meses em que o experimento prático esteve em funcionamento nenhuma manutenção foi necessária nas placas de sensores, com excessão de algumas pequenas alterações nos códigos, o que mostra que o algoritmo desenvolvido resultou de fato em um sistema eficiente e independente de qualquer interação humana.

O protocolo de comunicação MQTT se mostrou uma ótima escolha para este tipo de arquitetura de *Smart Home* devido à sua simplicidade, robustez e compatibilidade com os mais variados sistemas. O sistema PIMS utilizado permitiu armazenar os dados de leitura dos sensores e apresentá-los através da Internet para serem visualizados na forma de gráficos e indicadores, de forma que seja possível utilizá-los para posteriores análises.

A integração entre os componentes contidos na arquitetura desenvolvida proporcionou um sistema eficaz de monitoramento de ambientes residenciais com uma grande capacidade de armazenamento, o qual é escalável e pode ser utilizado em sistemas de controle de temperatura e iluminação.

O sensor utilizado para se medir a luminosidade, apesar de barato, não é o indicado para este tipo de aplicação pois ele tem uma resposta não-linear, além de ser sensível também à temperatura do ambiente. Uma sugestão para trabalhos futuros é substituir o sensor LDR por um fotodiodo ou um fototransistor para ser utilizado como sensor de luminosidade, obtendo assim uma resposta mais linear em relação à intensidade da luz captada.

Outra sugestão para trabalhos futuros é monitorar a potência utilizada, através de medidores de corrente, em luzes e condiconadores de ar, para que, junto dos indicadores de temperatura, umidade e luminosidade, seja desenvolvido um sistema de controle visando a eficiência energética e conforto térmico. Como foi observado nos resultados, em certos ambientes a incidência da luz solar tem grande influência na temperatura destes, então um sistema automatizado de cortinas nestes casos, por exemplo, possibilitado através de atuadores e um sistema de coleta de dados como o desenvolvido aqui, poderia por si só contribuir muito, economizando energia e mantendo a temperatura confortável.

REFERÊNCIAS

ARAÚJO, R. O. *Saúde e gerenciamento de ativos utilizando dados do Process Information Management System (PIMS)*. Universidade Federal de Ouro Preto, Ouro Preto, 2014.

ARKIN, H.; PACIUK, M. Service system integration in intelligent buildings. In: INTELLIGENT BUILDING CONGRESS, 1., 1995, Tel-Aviv. *Proceedings*. Tel-Aviv: IB/IC, 1995. p. 19–30.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. *NBR 15220-1*: Desempenho Térmico de edificações Parte 1: Definições, símbolos e unidades. Rio de Janeiro: ABNT, 2005. v. 1.

CARVALHO, F. B. DE et al. Sistemas PIMS - Conceituação, usos e benefícios. In: TECNOLOGIA em Metalurgia e Materiais. São Paulo: [s.n.], 2005. v. 1, p. 1–5.

CASTRO, M. N.; JOTA, F. G.; ASSIS, E. S. A Automação como Ferramenta para Eficiência Energética em Edificações. In: CONGRESSO BRASILEIRO DE EFICIÊNCIA ENERGÉTICA, 1., 2005, Belo Horizonte. *Anais*. Belo Horizonte: CBEE/ABEE, 2005. p. 268–273.

DOMB, M. Smart Home Systems Based on Internet of Things. In:

EMPRESA DE PESQUISA ENERGÉTICA. *Anuário Estatístico de Energia Elétrica 2014*: ano base 2013. Rio de Janeiro: EPE, 2014.

EVANS, D. The Internet of Things: How the Next Evolution of the Internet Is Changing Everything. In:

FANGER, P. O. *Thermal Comfort*. New York: McGraw-Hill Book Company, 1970.

FREITAS, C. C. S. DE et al. Automação Residencial - Uma abordagem em relação as atuais tecnologias e perspectivas para o futuro. v. 5, 2010.

FROTA, A. B.; SCHIFFER, S. R. *Manual de Conforto Térmico*. São Paulo: Studio Nobel, 2001. v. 5.

GELLER, H. S. *Revolução Energética - Políticas Para Um Futuro Sustentável*. Rio de Janeiro: Relume Dumara, 2003.

GRUPO VISION. *PIMS – Plant Information Management System*. [S.l.], 2015. Disponível em: <http://www.grupovision.com.br/blog/pims/>. Acesso em: 20 out. 2019.

HIVEMQ 1. *Publish & Subscribe - MQTT Essentials: Part 2*. [S.l.], 2015. Disponível em: <https://www.hivemq.com/blog/mqtt-essentials-part2-publish-subscribe/>. Acesso em: 20 out. 2019.

HIVEMQ 2. *MQTT Publish, Subscribe & Unsubscribe - MQTT Essentials: Part 4*. [S.l.], 2015. Disponível em: <https://www.hivemq.com/blog/mqtt-essentials-part-4-mqtt-publish-subscribe-unsubscribe/>. Acesso em: 20 out. 2019.

HIVEMQ 3. *MQTT Topics & Best Practices - MQTT Essentials: Part 5*. [S.l.], 2019. Disponível em: <https://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices/>. Acesso em: 20 out. 2019.

HIVEMQ 4. *Quality of Service 0,1 & 2 - MQTT Essentials: Part 6*. [S.l.], 2015. Disponível em: <https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/>. Acesso em: 20 out. 2019.

INTERNATIONAL ENERGY AGENCY. *ENERGY EFFICIENCY - Market Report 2015*: Market Trends and Medium-Term Prospects. Paris: iea, 2015.

JSON. *Introdução ao JSON*. [S.l.]. Disponível em: <https://www.json.org/json-pt.html>. Acesso em: 20 out. 2019.

LAMBERTS, R. et al. Casa Eficiente: Consumo e Geração de Energia. In: 2010. v. 2.

"UMA introdução à Internet da Coisas (IoT)": Parte 1. da "Série de IoT". In: LOPEZ RESEARCH.

MUSSIO, L. D.; MAIA, R. F.; LOPES, G. W. Um Estudo de Arquiteturas IoT para Dispositivos Embarcados. São Bernardo do Campo, v. 5, 2015.

NELSON, T. M.; NILSSON, T. H.; HOPKINS, G. W. *Thermal Comfort: Advantages and Deviations*. New York: American Society of Heating, Refrigerating e Air-Conditioning Engineers. v. 93, p. 1039–1054.

NEVES, J. P. *Uma abordagem para gerenciamento de fluxo de dados da Internet da Coisas*. Dissertação (Mestrado em Ciência da Computação) – Universidade Federal do Paraná, Curitiba, 2017.

OXFORD. *Internet of things*. [S.l.], 2019. Disponível em: <https://www.lexico.com/en/definition/internet_of_things>. Acesso em: 20 out. 2019.

RFC 7452. *Architectural Considerations in Smart Object Networking*. [S.l.], 2015. Disponível em: <https://tools.ietf.org/html/rfc7452>. Acesso em: 20 out. 2019.

ROSE, K.; ELDRIDGE, S.; CHAPIN, L. The Internet of Things: An Overview: Understanding the Issues and Challenges of a More Connected World. In:

SANTOS, B. P. et al. *Internet das Coisas: da Teoria à Prática*. 2016. Universidade Federal de Minas Gerais, Belo Horizonte.

SILVA, L. B. DA. *Análise da relação entre produtividade e conforto térmico: o caso dos digitadores do centro de processamento de dados e cobrança da Caixa Econômica Federal do estado de Pernambuco*. Tese (Doutorado em Engenharia de Produção) – Universidade Federal de Santa Catarina, Florianópolis, 2001.

SILVA, P. C. P. DA. *Análise do comportamento térmico de construções não convencionais através de simulação em VisualDOE*. Dissertação (Mestrado em Engenharia Civil) – Universidade do Minho, Braga, 2006.

SKERRETT, I. *Case Study MQTT: Why Open Source and Open Standards Drives Adoption*. [S.l.], 2015. Disponível em: <https://ianskerrett.wordpress.com/2015/03/04/case-study-mqtt-why-open-source-and-open-standards-drives-adoption/>. Acesso em: 20 out. 2019.

TEZA, V. R. *Alguns aspectos sobre a Automação Residencial - Domótica*. Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de Santa Catarina, Florianópolis, 2002.

WEISER, M. The computer for the 21st century. In: 3. v. 265, p. 94–105.

YUAN, M. *Conhecendo o MQTT*: Por que o MQTT é um dos melhores protocolos de rede para a Internet das Coisas? [S.l.], 2017. Disponível em: <https://www.ibm.com/developerworks/br/library/iot-mqtt-why-good-for-iot/index.html>. Acesso em: 20 out. 2019.

APÊNDICE A CÓDIGO EM C++ DA PLACA DO QUARTO

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

#define MQTTPUBTOPIC_T      "Quarto/Temperatura"
#define MQTTPUBTOPIC_H      "Quarto/Umidade"
#define MQTTPUBTOPIC_L      "Quarto/Luminosidade"

#define LDRPIN A0
#define MAX_ADC_READING 1023
#define ADC_REF_VOLTAGE 3.3
#define REF_RESISTANCE 100
int const bSetup  = 4;
int const ledPin  = 5;
boolean altPin = 1;

#define ID_MQTT  "ESP_Quarto"

// WIFI
const char* SSID = "xxxxxxxx";
const char* PASSWORD = "xxxxxxxx";

// MQTT
const char* BROKER_MQTT = "soldier.cloudmqtt.com";
int BROKER_PORT = 13102;
const char* mqttUser = "xxxxxxx";
const char* mqttPassword = "xxxxxxx";

// Tipo do sensor DHT
#define DHTTYPE DHT22        // DHT22(AM2302/ AM2321)

// Configuracao dos pinos do ESP8266 NodeMcu
#define PINDHT 13

// Inicializacao das variaveis medidas
```

```
float tmp = 0.0;      // Temperatura
float hum = 0.0;      // Umidade
float lum = 0.0;      // Luminosidade
String timestamp = "";

// Variaveis e objetos globais
WiFiClient espClient;
PubSubClient MQTT(espClient);
DHT dht(PINDHT, DHTTYPE);

// Prototipos
void initSerial();
void initWiFi();
void initMQTT();
void initTime();
void InitIO(void);
void reconnectWiFi();
void reconnectMQTT();
void VerificaConexoesWiFIEMQTT(void);
void readSensor2Mqtt();
void pubMqttBroker();
void AlternateLed();
String ValueToJSON(String);

void setup()
{
    // Inicializacoes:
    InitIO();
    initWiFi();
    initTime();
    initMQTT();
}

void InitIO(void)
{
    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, HIGH);
    Serial.begin(9600);
    dht.begin();
}

void initWiFi()
{
    delay(10);
    Serial.println("------Conexao WI-FI------");
    Serial.print("Conectando-se na rede: ");
    Serial.println(SSID);
    Serial.println("Aguarde");
```

```
    WiFi.mode(WIFI_STA);
    reconnectWiFi();
}

void initMQTT()
{
    MQTT.setServer(BROKER_MQTT, BROKER_PORT);
}

void initTime()
{
  configTime(-(3 * 3600), 0, "pool.ntp.org", "time.nist.gov
    ");
  while (!time(nullptr)) {
    Serial.print(".");
    delay(1000);
  }
  time_t now = time(nullptr);
  timestamp = now;
}

void reconnectWiFi()
{
    if (WiFi.status() == WL_CONNECTED)
        return;

    WiFi.begin(SSID, PASSWORD);

    while (WiFi.status() != WL_CONNECTED)
    {
      AlternateLed();
      delay(500);
      Serial.print(".");
    }

    Serial.println();
    Serial.print("Conectado com sucesso na rede ");
    Serial.print(SSID);
    Serial.println("IP obtido: ");
    Serial.println(WiFi.localIP());
    digitalWrite(ledPin, LOW);

    initTime();
}

void reconnectMQTT()
{
```

```
    while (!MQTT.connected())
    {
        Serial.print("␣Tentando␣conectar␣ao␣Broker␣MQTT:␣")
            ;
        Serial.println(BROKER_MQTT);
        MQTT.connect(ID_MQTT, mqttUser, mqttPassword);

        AlternateLed();

        if (!MQTT.connected())
        {
          Serial.println("Falha␣ao␣reconectar␣no␣broker.");
          Serial.println("Havera␣nova␣tentatica␣de␣conexao␣
              em␣2s");
          reconnectWiFi();
          delay(2000);
        }

        delay(500);
        digitalWrite(ledPin, LOW);
    }

    Serial.println("MQTT␣conectado␣com␣sucesso");
}

void AlternateLed()
{
  if (altPin == 1)
  {
    digitalWrite(ledPin, LOW);
    altPin = 0;
  }
  else
  {
    digitalWrite(ledPin, HIGH);
    altPin = 1;
  }
}

String ValueToJSON(String value)
{
  String messageJSON;

  String quality;
  if (value == "999")
    quality = "0";
  else
    quality = "1";
```

```
  messageJSON = "{ \"value\": " + value +", \"timestamp\": "
     " + timestamp + ", \"quality\": " + quality + " }";

  return messageJSON;
}

void VerificaConexoesWiFIEMQTT(void)
{
    reconnectWiFi();
    if (!MQTT.connected())
    {
        reconnectMQTT();
    }
}

void readSensor2Mqtt()
{
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  float l = analogRead(LDRPIN);

  float Vout = (l * (ADC_REF_VOLTAGE/1024));
  float rLdr = (REF_RESISTANCE * (ADC_REF_VOLTAGE - Vout))/
     Vout;
  float lux = 47951162.33 * pow(rLdr, -1.865874281);

  // Checando se qualquer leitura falhou
  if (isnan(h) || isnan(t))
  {
    for (int i=0; i<5; i++){
          t = dht.readTemperature();
          h = dht.readHumidity();
          if (t > 0 || h > 0 ){
                tmp = t;
                hum = h;
                lum = lux;
                break;
           }
           else
           {
            Serial.println("Falha na leitura do sensor!");
            tmp = 999;
            hum = 999;
            lum = lux;

           }
      delay(2000);
```

```
        }
    }
    else{
      tmp = t;
      hum = h;
      lum = lux;

      delay(1000);
    }
    Serial.print(F("Humidity: "));
    Serial.print(h);
    Serial.print(F("%  Temperature: "));
    Serial.print(t);
    Serial.print(F(" C "));
    Serial.print(F("Luminosity: "));
    Serial.println(String(lum) + "lux");
}

void pubMqttBroker()
{
    String sTmp = String(tmp);
    String sHum = String(hum);
    String sLum = String(lum);

    String jsonTmp;
    String jsonHum;
    String jsonLum;

    if (WiFi.status() == WL_CONNECTED)
    {
      time_t now = time(nullptr);
      timestamp = now;
      Serial.println(timestamp);

      jsonTmp = ValueToJSON(sTmp);
      jsonHum = ValueToJSON(sHum);
      jsonLum = ValueToJSON(sLum);

      char* msgMqtt_T = new char[jsonTmp.length() + 1];
      char* msgMqtt_H = new char[jsonHum.length() + 1];
      char* msgMqtt_L = new char[jsonLum.length() + 1];

      strcpy(msgMqtt_T, jsonTmp.c_str());
      strcpy(msgMqtt_H, jsonHum.c_str());
      strcpy(msgMqtt_L, jsonLum.c_str());

      if (MQTT.connected())
      {
```

```
        MQTT.publish(MQTTPUBTOPIC_T, msgMqtt_T, true);
        MQTT.publish(MQTTPUBTOPIC_H, msgMqtt_H, true);
        MQTT.publish(MQTTPUBTOPIC_L, msgMqtt_L, true);
        Serial.println("Dados publicados.");
      }
      else
        Serial.println("Falha em publicar.");

      delete [] msgMqtt_T;
      delete [] msgMqtt_H;
      delete [] msgMqtt_L;
    }
}


// Programa principal
void loop()
{
    VerificaConexoesWiFIEMQTT();
    Serial.println("Conexoes verificadas.");
    readSensor2Mqtt();
    Serial.println("Leitura executada.");
    pubMqttBroker();

    delay(10000);
}
```

APÊNDICE B CÓDIGO EM C++ DA PLACA DA SALA

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

#define MQTTPUBTOPIC_T      "Sala/Temperatura"
#define MQTTPUBTOPIC_H      "Sala/Umidade"
#define MQTTPUBTOPIC_L      "Sala/Luminosidade"

#define LDRPIN A0
#define MAX_ADC_READING 1023
#define ADC_REF_VOLTAGE 3.3
#define REF_RESISTANCE 100
int const bSetup  = 4;
int const ledPin  = 5;
boolean altPin = 1;

#define ID_MQTT  "ESP_Sala"

// WIFI
const char* SSID = "xxxxxxxx";
const char* PASSWORD = "xxxxxxxx";

// MQTT
const char* BROKER_MQTT = "soldier.cloudmqtt.com";
int BROKER_PORT = 13102;
const char* mqttUser = "xxxxxxx";
const char* mqttPassword = "xxxxxxx";

// Tipo do sensor DHT
#define DHTTYPE DHT22        // DHT22(AM2302/ AM2321)

// Configuracao dos pinos do ESP8266 NodeMcu
#define PINDHT 13

// Inicializacao das variaveis medidas
```

```
float tmp = 0.0;      // Temperatura
float hum = 0.0;      // Umidade
float lum = 0.0;      // Luminosidade
String timestamp = "";

// Variaveis e objetos globais
WiFiClient espClient;
PubSubClient MQTT(espClient);
DHT dht(PINDHT, DHTTYPE);

// Prototipos
void initSerial();
void initWiFi();
void initMQTT();
void initTime();
void InitIO(void);
void reconnectWiFi();
void reconnectMQTT();
void VerificaConexoesWiFIEMQTT(void);
void readSensor2Mqtt();
void pubMqttBroker();
void AlternateLed();
String ValueToJSON(String);

void setup()
{
    // Inicializacoes:
    InitIO();
    initWiFi();
    initTime();
    initMQTT();
}

void InitIO(void)
{
    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, HIGH);
    Serial.begin(9600);
    dht.begin();
}

void initWiFi()
{
    delay(10);
    Serial.println("------Conexao WI-FI------");
    Serial.print("Conectando-se na rede: ");
    Serial.println(SSID);
    Serial.println("Aguarde");
```

```
    WiFi.mode(WIFI_STA);
    reconnectWiFi();
}

void initMQTT()
{
    MQTT.setServer(BROKER_MQTT, BROKER_PORT);
}

void initTime()
{
  configTime(-(3 * 3600), 0, "pool.ntp.org", "time.nist.gov
    ");
  while (!time(nullptr)) {
    Serial.print(".");
    delay(1000);
  }
  time_t now = time(nullptr);
  timestamp = now;
}

void reconnectWiFi()
{
    if (WiFi.status() == WL_CONNECTED)
        return;

    WiFi.begin(SSID, PASSWORD);

    while (WiFi.status() != WL_CONNECTED)
    {
      AlternateLed();
      delay(500);
      Serial.print(".");
    }

    Serial.println();
    Serial.print("Conectado com sucesso na rede ");
    Serial.print(SSID);
    Serial.println("IP obtido: ");
    Serial.println(WiFi.localIP());
    digitalWrite(ledPin, LOW);

    initTime();
}

void reconnectMQTT()
{
```

```
    while (!MQTT.connected())
    {
        Serial.print("␣Tentando␣conectar␣ao␣Broker␣MQTT:␣")
            ;
        Serial.println(BROKER_MQTT);
        MQTT.connect(ID_MQTT, mqttUser, mqttPassword);

        AlternateLed();

        if (!MQTT.connected())
        {
          Serial.println("Falha␣ao␣reconectar␣no␣broker.");
          Serial.println("Havera␣nova␣tentatica␣de␣conexao␣
              em␣2s");
          reconnectWiFi();
          delay(2000);
        }

        delay(500);
        digitalWrite(ledPin, LOW);
    }

    Serial.println("MQTT␣conectado␣com␣sucesso");
}

void AlternateLed()
{
  if (altPin == 1)
  {
    digitalWrite(ledPin, LOW);
    altPin = 0;
  }
  else
  {
    digitalWrite(ledPin, HIGH);
    altPin = 1;
  }
}

String ValueToJSON(String value)
{
  String messageJSON;

  String quality;
  if (value == "999")
    quality = "0";
  else
    quality = "1";
```

```
  messageJSON = "{ \"value\": " + value +", \"timestamp\": "
     " + timestamp + ", \"quality\": " + quality + " }";

  return messageJSON;
}

void VerificaConexoesWiFIEMQTT(void)
{
    reconnectWiFi();
    if (!MQTT.connected())
    {
        reconnectMQTT();
    }
}

void readSensor2Mqtt()
{
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  float l = analogRead(LDRPIN);

  float Vout = (l * (ADC_REF_VOLTAGE/1024));
  float rLdr = (REF_RESISTANCE * (ADC_REF_VOLTAGE - Vout))/
     Vout;
  float lux = 47951162.33 * pow(rLdr, -1.865874281);

  // Checando se qualquer leitura falhou
  if (isnan(h) || isnan(t))
  {
    for (int i=0; i<5; i++){
          t = dht.readTemperature();
          h = dht.readHumidity();
          if (t > 0 || h > 0 ){
                tmp = t;
                hum = h;
                lum = lux;
                break;
           }
           else
           {
            Serial.println("Falha na leitura do sensor!");
            tmp = 999;
            hum = 999;
            lum = lux;

           }
      delay(2000);
```

```
        }
    }
    else{
      tmp = t;
      hum = h;
      lum = lux;

      delay(1000);
    }
    Serial.print(F("Humidity: "));
    Serial.print(h);
    Serial.print(F("%  Temperature: "));
    Serial.print(t);
    Serial.print(F(" C "));
    Serial.print(F("Luminosity: "));
    Serial.println(String(lum) + "lux");
}

void pubMqttBroker()
{
    String sTmp = String(tmp);
    String sHum = String(hum);
    String sLum = String(lum);

    String jsonTmp;
    String jsonHum;
    String jsonLum;

    if (WiFi.status() == WL_CONNECTED)
    {
      time_t now = time(nullptr);
      timestamp = now;
      Serial.println(timestamp);

      jsonTmp = ValueToJSON(sTmp);
      jsonHum = ValueToJSON(sHum);
      jsonLum = ValueToJSON(sLum);

      char* msgMqtt_T = new char[jsonTmp.length() + 1];
      char* msgMqtt_H = new char[jsonHum.length() + 1];
      char* msgMqtt_L = new char[jsonLum.length() + 1];

      strcpy(msgMqtt_T, jsonTmp.c_str());
      strcpy(msgMqtt_H, jsonHum.c_str());
      strcpy(msgMqtt_L, jsonLum.c_str());

      if (MQTT.connected())
      {
```

```
      MQTT.publish(MQTTPUBTOPIC_T, msgMqtt_T, true);
      MQTT.publish(MQTTPUBTOPIC_H, msgMqtt_H, true);
      MQTT.publish(MQTTPUBTOPIC_L, msgMqtt_L, true);
      Serial.println("Dados publicados.");
    }
    else
      Serial.println("Falha em publicar.");

    delete [] msgMqtt_T;
    delete [] msgMqtt_H;
    delete [] msgMqtt_L;
  }
}


// Programa principal
void loop()
{
    VerificaConexoesWiFIEMQTT();
    Serial.println("Conexoes verificadas.");
    readSensor2Mqtt();
    Serial.println("Leitura executada.");
    pubMqttBroker();

    delay(10000);
}
```

APÊNDICE C CÓDIGO EM PYTHON DOS INDICADORES DE CONFORTO TÉRMICO

```
3 if T.Value <= 6 or T.Value >= 36 or H.Value <= 20 else (1
    if T.Value >= 23 and T.Value <= 26 and H.Value >= 40
  and H.Value <= 80 else 2)

'''
T.Value representa a medida de temperatura do ambiente
H.Value representa a medida de umidade do ambiente
O valor 1 representa a cara feliz e verde
O valor 2 representa a cara neutra e laranja
O valor 3 representa a cara triste e vermelha
'''
```