LabSensing: Um Sistema de Sensoriamento para Laboratórios Científicos com Computação Inteligente nas Bordas

Kaylani Bochie e Miguel Elias M. Campista *

[1]Grupo de Teleinformática e Automação (GTA)
PEE/COPPE-DEL/Poli
Universidade Federal do Rio de Janeiro (UFRJ)

{kaylani,miguel}@gta.ufrj.br

Resumo. *Este trabalho propõe o LabSensing, um sistema de sensoriamento distribuído para monitoramento de laboratórios científicos. O LabSensing utiliza uma estratégia de comunicação assíncrona para transferência de dados e reduz o tráfego injetado na rede através de políticas de eliminação de dados redundantes nas bordas. As motivações técnicas que levaram à escolha das ferramentas, assim como o processo de desenvolvimento do sistema a partir da arquitetura proposta são discutidos. Os resultados experimentais mostram o funcionamento do sistema e a efetividade obtida com a computação nas bordas para redução da carga de dados na rede.*

Abstract. *This paper proposes LabSensing, a distributed sensing system for scientific laboratory monitoring. LabSensing employs an asynchronous communication strategy to transfer data and reduces the traffic injected in the network using policies to eliminate redundant data at the edges. The technical motivations leading to tools selection, as well as the design and development of the system taking into account the proposed architecture are discussed. Experimental results show the operation of the system and the impact obtained using edge computing to reduce the network data load.*

1. Introdução

A explosão de dispositivos eletrônicos tem impulsionado o paradigma de Internet das Coisas (*Internet of Things* – IoT), ao possibilitar diversas aplicações com base em sensoriamento. A popularização dessas aplicações leva a um aumento do número de dispositivos conectados à Internet e ao crescimento do volume de dados gerado e encaminhado pelas redes de comunicação. Diversos conceitos e abordagens vem sendo aplicados no desenvolvimento de ferramentas e sistemas nesse contexto, tais como a Computação na Borda (*Edge Computing*) [Shi et al., 2016]. Apesar de ambos os conceitos, IoT e Computação na Borda, costumarem ser aliados a soluções baseadas em nuvens computacionais de larga escala, os mesmos princípios podem ser aplicados a redes locais. A ideia é igualmente distribuir e melhor utilizar o poder computacional disponível em dispositivos embarcados e sistemas distribuídos.

*O presente trabalho foi realizado com apoio do CNPq; da FAPERJ; da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES), Código de Financiamento 001; e da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processos n^o 15/24494-8 e 15/24490-2.

Em redes locais, um dos problemas de implementação do sensoriamento é a má utilização dos recursos computacionais dos nós de borda. Essa prática gera sobrecarga no nó central da rede, responsável pela recepção dos dados gerados, contrastando com a subutilização dos nós de sensoriamento. Ao invés de cumprir um papel meramente de sensores, esses nós podem ser usados para verificar os dados coletados e, apenas se necessário, enviá-los ao nó central. A ideia é distribuir o processamento pelas bordas da rede, agregando inteligência para: (i) reduzir o volume de dados enviados até o nó central e (ii) aumentar a escalabilidade do sistema mesmo em ambientes locais. Outra vantagem do deslocamento do processamento para as bordas da rede é a redução do tempo de resposta, que se torna útil se os sensores esperarem alguma forma de retorno do sistema.

Na literatura existem trabalhos que também propõem sistemas de monitoramento. Por exemplo, Kodali e Gorantla discutem escolhas de software e hardware para uma rede de sensores sem fio [Kodali e Gorantla, 2017]. Também são analisadas as possíveis aplicações para redes de baixo custo e os méritos da tecnologia utilizada. Alqinsi et al. constroem um sistema de monitoramento através do protocolo MQTT (*Message Queue Telemetry Transport*) [Alqinsi et al., 2018]. Saha e Majumdar desenvolvem um sistema de monitoramento para centros de dados baseado em nuvem [Saha e Majumdar, 2017]. O sistema proposto neste trabalho segue a tendência usando, além das técnicas encontradas na literatura de sistemas IoT, conceitos utilizados em computação na borda aplicados a redes locais, tais como distribuição de poder computacional nos dispositivos de sensoriamento. Isso tem como consequência o aumento de confiabilidade do sistema, devido à independência de serviços remotos e agentes externos.

Este artigo propõe o LabSensing, um sistema para sensoriamento de dados em laboratórios científicos complementar a trabalhos anteriores [Kodali e Gorantla, 2017, Alqinsi et al., 2018, Saha e Majumdar, 2017]. O LabSensing reúne conceitos de IoT e de Computação na Borda e os aplica a sistemas implementados em redes locais. Essa característica elimina o uso das nuvens computacionais comerciais, reduzindo os custos da operação do sistema. Dessa forma, tanto os sensores quanto os nós de borda e o servidor central são instalados no próprio laboratório. A visualização dos dados é feita a partir de uma interface *web*, acessível tanto internamente quanto externamente ao laboratório. O sistema LabSensing permite o monitoramento do laboratório a partir de nós sensores diversos e evita que todos os dados sejam enviados ao servidor central através de processamento nas bordas. O LabSensing é construído com hardware de baixo custo e software livre, além de permitir a adição de componentes de forma modular, desde que estes possam implementar o MQTT, adotado por permitir a troca de mensagens em sistemas locais com restrições de hardware [Oasis, 2014]. O sistema proposto é também capaz de variar as políticas de processamento nas bordas através de parâmetros de controle no código de cada um dos nós. Os resultados experimentais mostram que o pré-processamento realizado nas bordas da rede alivia o volume de dados transmitido. Essa redução do tráfego, aliada ao baixo custo dos dispositivos, faz com que o sistema seja ideal para utilização em redes locais, porém ainda extensível para ambientes de maior escala.

Este trabalho está organizado da seguinte forma: a Seção 2 descreve a arquitetura do LabSensing e apresenta as ferramentas utilizadas para a sua implementação. A Seção 3 apresenta os experimentos práticos conduzidos e a análise dos resultados obtidos. Por fim, a Seção 4 conclui este trabalho e aponta as direções futuras.

2. Arquitetura e Implementação do LabSensing

Esta seção descreve inicialmente a arquitetura do LabSensing e, em seguida, as opções de implementação.

2.1. Arquitetura

A arquitetura do LabSensing segue uma divisão em camadas típica de sistemas de IoT. Ele é composto por uma camada de *percepção*, uma de *comunicação* e uma de *armazenamento e exibição dos dados*. A camada de percepção, responsável pelo sensoriamento do ambiente monitorado, é composta por nós sensores distribuídos, chamados de nós de percepção. Esses sensores são implementados em dispositivos capazes de reduzir a carga de dados injetada na rede, segundo políticas de filtragem predeterminadas. Essas políticas podem ser usadas para atribuir "inteligência" aos nós de borda. Já a camada de comunicação, responsável pela troca de mensagens entre nós produtores e consumidores de dados, utiliza um substrato composto por redes sem fio e cabeadas. Os sensores utilizam redes sem fio devido a facilidades de instalação física, mas essa decisão não representa uma limitação da arquitetura. A camada de comunicação ainda possui um Nó de Concentração conectado ao Broker através de uma interface local (*localhost*). Os dados enviados através da rede são armazenados em um nó central que executa a camada de armazenamento e exibição dos dados do sistema de IoT. O Nó Concentrador da arquitetura executa clientes que recebem, processam e armazenam os dados da base de dados disponível. Por fim, um *dashboard* é adicionado para visualização em tempo real dos dados por parte dos usuários. Essa possibilidade de visualização é uma alternativa a consultas diretas a base de dados. A Figura 1 apresenta uma visão geral da arquitetura, na qual as setas representam a direção do fluxo de dados desde os nós de percepção até a visualização em um *dashboard*.

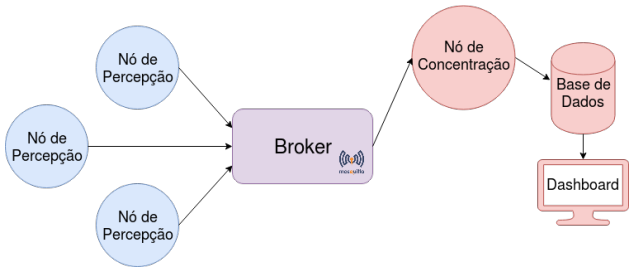


Figura 1. Arquitetura do LabSensing composta por nós de percepção (nós sensores), interconexão entre os componentes e um servidor central responsável pelo armazenamento, processamento e exibição dos dados.

2.2. Implementação

A implementação do LabSensing prioriza o emprego de hardware de baixo custo e software livre. Esta seção apresenta o hardware utilizado, o software e a implementação da inteligência nas bordas da rede. A apresentação da implementação está organizada a partir da camada de comunicação, uma vez que uma decisão importante da arquitetura foi a comunicação assíncrona entre os nós consumidores e produtores de dados. Todos os códigos utilizados para programar os microcontroladores podem ser encontrados em um repositório GitHub [Bochie, 2019].

Camada de comunicação: É utilizada para transferência de mensagens. Nesse sentido, o protocolo MQTT é selecionado para proporcionar a comunicação entre os nós sensores e o nó central de forma assíncrona. O MQTT foi escolhido por ser um protocolo baseado no paradigma *Publish/Subscribe* para comunicação entre dispositivos de rede com limitações de banda e poder computacional. A operação típica do MQTT envolve agentes do tipo cliente e Broker. O papel do Broker é receber as mensagens dos clientes produtores e redirecioná-las aos clientes consumidores interessados. As mensagens são publicadas em tópicos pelos clientes produtores e consumidas apenas por aqueles que estejam inscritos nos tópicos correspondentes. Isso possibilita a comunicação Dispositivo a Dispositivo (D2D) através do Broker. O protocolo MQTT também suporta três níveis de QoS (*Quality of Service*): no QoS 0 (*At Most Once*) a entrega das mensagens não é garantida; no QoS 1 (*At Least Once*) o Broker garante que os clientes inscritos no tópico recebam no mínimo uma cópia da mensagem; no QoS 2 (*Exactly Once*), o Broker garante que os clientes inscritos no tópico recebam apenas uma cópia da mensagem enviada.

Os sensores utilizam redes Wi-Fi, por questões de facilidade de instalação, enquanto o Nó de Concentração foi implementado pelo software Node-RED na mesma máquina onde o Broker foi instalado. O nó utiliza a interface local (*localhost*) para se conectar ao Broker. Sobre a configuração da rede, diversos pontos de acesso Wi-Fi são espalhados pelo laboratório utilizando *Raspberry Pis*, que aumentam a cobertura de rede e permitem uma melhor distribuição dos sensores pelo ambiente de monitoramento. Para implementações em redes locais, é possível que o número de endereços disponíveis seja insuficiente para um número elevado de sensores. Este problema, porém, pode ser contornado por pontos de acesso Wi-Fi que utilizem NAT (*Network Address Translation*).

O software Mosquitto [Eclipse, 2019] foi escolhido para implementação do Broker por ser livre e ter extenso suporte na comunidade acadêmica. Os clientes MQTT podem ser dispositivos diversos, enquanto o Broker deve ser capaz de lidar com um grande número de conexões. Estas escolhas não impedem a adição de nós atuadores ao sistema.

Camada de percepção: É responsável por produzir os dados de sensoriamento vindos de dispositivos diversos. A implementação desta camada se deu a partir de nós baseados no módulo NodeMCU, clientes produtores de dados enviados através do MQTT.

O NodeMCU é uma plataforma IoT de software livre baseada no SoC (*System on Chip*) Wi-Fi ESP8266. O hardware é baseado no módulo ESP-12. Apesar do termo NodeMCU se referir apenas ao firmware ao invés da placa de desenvolvimento, ele é comumente utilizado para descrever toda a plataforma de prototipagem, envolvendo hardware e software. Além disso, cada nó de sensoriamento é composto pelo módulo NodeMCU e por um *array* de sensores. A quantidade e tipo de cada sensor dependem da função do nó no ambiente de sensoriamento. A implementação atual do LabSensing possui quatro nós de sensoriamento equipados com os sensores listados na Tabela 1.Esses nós foram distribuídos pelo laboratório para monitoramento diverso, sendo que dois deles foram implementados de forma a possibilitar a análise de desempenho do sistema, são eles:

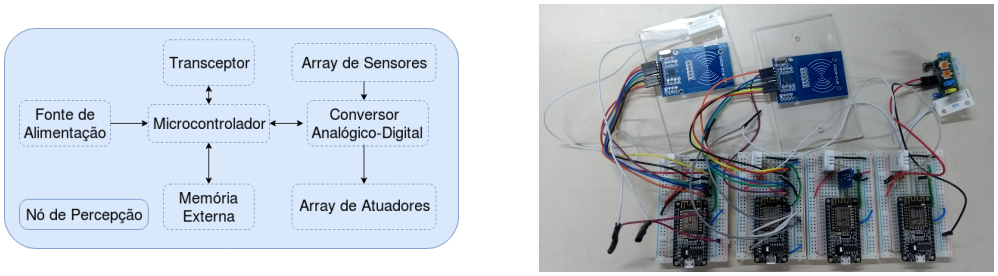
- **Nó Síncrono:** envia todos os dados coletados em um intervalo fixo de tempo, definido por um parâmetro no arquivo de configuração do sistema. Entre intervalos de transmissão, o nó assume um estado de *deep sleep* para economia de energia.
- **Nó Assíncrono:** envia os dados coletados apenas quando um evento pré-programado é detectado, ou seja, é orientado a eventos. Os eventos escolhidos

foram a mudança de estados detectada pelo sensor de porta, identificação de novo cartão RFID e mudanças abruptas de temperatura.

Tabela 1. Sensores em cada um dos quatro nós de sensoriamento da rede.

Nós	Sensores				
	Leitor RFID (MFRC522)	**Temperatura e Umidade (DHT22)**	**Chama**	**Movimento PIR**	**Porta *Reed Switch***
Nó 1 - Assíncrono	√	√			√
Nó 2 - Síncrono	√	√			√
Nó 3 - Dados		√	√		√
Nó 4 - Professor		√		√	

A Figura 2(a) mostra o projeto geral dos nós sensores com os elementos necessários de cada um, enquanto a Figura 2(b) mostra de fato os quatro nós sensores implementados. O transceptor de todos os nós deve suportar comunicação Wi-Fi, sendo que nem todos os nós possuem o mesmo *array* de sensores. A troca de sensores, porém, é simples e pode ser realizada caso haja necessidade. Na Figura 2(b), da esquerda para a direita, observa-se o Nó Assíncrono, o Nó Síncrono, o nó posicionado na sala do centro de dados do laboratório (Nó Dados) e o nó da sala de um professor (Nó Professor).



(a) (b)

Figura 2. (a) Esquema geral de um nó de sensoriamento. (b) Nós implementados e utilizados nos experimentos.

Como mencionado anteriormente, o LabSensing reduz a quantidade de dados injetados na rede através de processamento nas bordas. Para tal, são definidas duas políticas para a transmissão de dados que são simplesmente o envio periódico de mensagens e o envio de mensagens baseado em limiar. A segunda política apenas envia dados quando uma alteração acima de um limiar predefinido é detectada, levando em conta os dados gerados no ambiente sensoreado. A política é implementada diretamente no código do nó pertinente. No caso de teste, o nó busca detectar quaisquer mudanças de estado em seus sensores: uma porta abrindo ou fechando, um usuário introduzindo um cartão RFID no leitor ou uma variação significativa de temperatura. A ideia do envio baseado em limiar é reduzir de forma simples a carga na rede.

Camada de armazenamento e exibição dos dados: Executada em um nó central, esta camada necessita de um cliente MQTT para obter os dados produzidos na borda, um banco de dados para garantir persistência dos dados recebidos e uma ferramenta para visualização. Assim, o Node-RED, ferramenta de programação visual criada com o intuito de agir como intermediário entre diferentes protocolos, APIs (*Application Programming Interface*) e hardwares, é usado como nó de concentração. No LabSensing, o cliente

Node-RED está inscrito nos tópicos onde os nós de percepção publicam suas leituras. O nó de concentração ainda processa a informação recebida para posteriormente escrever na base de dados. Os dados são recebidos em formato JSON (*JavaScript Object Notation*) com campos do tipo *string* e são passados em formato JSON com tipos corretos para a base de dados. A implementação do *flow* Node-RED desenvolvido pode ser encontrada em um repositório GitHub [Bochie, 2019].

A ferramenta também pode ser utilizada para exibir as informações recebidas em um painel acessado via navegador *web*. Dessa forma, uma ferramenta de visualização é utilizada para permitir que um usuário do sistema analise quaisquer alterações no ambiente. Caso algum sensor detecte situações que necessitem de atuação de um administrador, alarmes podem ser disparados.

Os dados coletados para posteriormente serem exibidos são armazenados em uma base de dados especializada em dados temporais. A base de dados InfluxDB, desenvolvida pela InfluxData [InfluxData, 2019], é usada por ser uma plataforma aberta, desenvolvida para recuperação rápida de dados. Essas características a habilitam para o uso em aplicações de IoT. Já o *dashboard*, definido na arquitetura, é outra ferramenta desenvolvida pela InfluxData, chamada Grafana, desenvolvida para realizar consultas a bases de dados temporais. Uma página *web* é usada para visualizar o conteúdo de cada base de dados. O Grafana também possibilita a alteração dos intervalos temporais de interesse, o que facilita a detecção de erros na transmissão de mensagens e na base de dados.

3. Resultados Experimentais

Dois nós de sensoriamento (Nó Assíncrono e Nó Síncrono) foram projetados para medir a diferença de desempenho do sistema com as duas políticas de transmissão implementadas. Em ambos os casos, os nós possuem o mesmo hardware, mas operam de forma descrita na Seção 2. Os dados foram coletados ao longo de 24 horas no laboratório GTA (Grupo de Teleinformática e Automação) da Poli/COPPE, UFRJ. Porém, os dados gerados pelo Nó Assíncrono não foram coletados em intervalos regulares. As políticas de transmissão dos nós sensores tiveram seus parâmetros alterados para análise do impacto de tais políticas na redução dos dados coletados nas bordas.

A Tabela 2 mostra o número de mensagens inseridas na base de dados no servidor central em função das políticas de transmissão. O Nó Síncrono foi programado para entrar em modo *deep sleep* em intervalos regulares definidos por um parâmetro no código do nó. As medidas exibidas na Tabela 2 foram coletadas durante o mesmo período de 2 horas em dias úteis no laboratório. Já o Nó Assíncrono foi programado para enviar mensagens caso detecte eventos e com o limiar de variação de temperatura ajustado para 1^oC. O número de mensagens injetadas na rede é estimado a partir do número de entradas na base de dados no servidor que, para o Nó Síncrono, resulta em um valor exato já que o nó envia dados em intervalos conhecidos. Caso haja um intervalo sem a inserção de uma nova entrada na base de dados, isso pode ser assumido como uma perda na transmissão já que as mensagens usam QoS 0. No caso do Nó Assíncrono, o número de mensagens injetadas na rede é uma estimativa, já que a base de dados apresenta apenas as mensagens enviadas com sucesso e não há como saber quando uma nova mensagem é gerada. Para detectar perdas, seria necessário registrar também as mensagens enviadas no nó transmissor, o que ainda não é feito neste trabalho. Assume-se que isso não é impactante já que a rede não

apresenta perdas relevantes em sua operação normal.

A comparação dos resultados entre os Nós Síncrono e Assíncrono mostra que o uso da política de redução usada no Nó Assíncrono permite uma redução significativa no número de entradas na base de dados. Em relação ao *deep sleep* usado no Nó Síncrono, percebe-se que há uma redução esperada conforme aumenta-se o intervalo de economia de energia. Vale ressaltar que a mudança no intervalo do *deep sleep* do Nó Síncrono não tem nenhuma relação influência no Nó Assíncrono. Os números diferentes listados na tabela relativos ao Nó Assíncrono são uma mera consequência do número de eventos observados durante os experimentos.

Tabela 2. Número de entradas na base de dados para as duas políticas de transmissão.

Nó Síncrono	Nó Assíncrono
198 (5 segundos em *deep sleep*)	9
165 (10 segundos em *deep sleep*)	23
129 (15 segundos em *deep sleep*)	16

A Figura 3 apresenta os dados coletados pelo Nó Assíncrono e pelo Nó Síncrono durante 4,5 horas em uma tarde de funcionamento regular do laboratório. O intervalo foi escolhido por mostrar bem como os nós refletem os dados medidos durante um período de variação de temperatura. Nota-se o nível de granularidade superior do Nó Síncrono, o que é esperado, já que a política empregada por este nó envia todas as mensagens geradas. Entretanto, mesmo com um número inferior de mensagens, o Nó Assíncrono consegue acompanhar as medidas do Nó Síncrono com um número bem menor de mensagens enviadas na rede. As duas políticas, porém, não são excludentes já que a decisão entre qual utilizar depende da aplicação sensoreada.

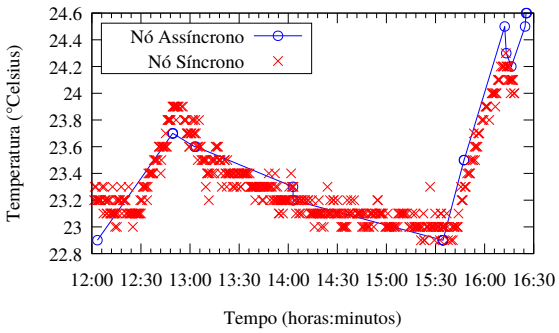


Figura 3. Mensagens inseridas na base de dados pelos Nós Assíncrono e Síncrono.

O consumo de energia dos dois nós foi monitorado por 24 horas durante o funcionamento regular do laboratório. O Nó Síncrono consume 795 miliwatts-hora em 24 horas. Este valor foi obtido enquanto o nó opera em estado *deep sleep* por 15 segundos entre transmissões e em estado normal por 7 segundos durante coleta de dados e transmissões. O Nó Assíncrono consumiu 1750 miliwatts-hora em 24 horas. A diferença de consumo entre os dois nós para os parâmetros selecionados é de 955 miliwatts-hora por dia. Observa-se que o consumo do Nó Assíncrono é maior devido a necessidade de monitorar o ambiente sem intervalos para dormir. Essa política de transmissão se justifica

caso seja necessário detectar eventos e processá-los sem risco de perda. Uma política híbrida, síncrona e assíncrona, pode ser vislumbrada dependendo das características do ambiente monitorado, por exemplo, o sensor pode usar a política assíncrona durante o dia e a síncrona durante a noite.

4. Conclusão e Trabalhos Futuros

Este trabalho apresentou o LabSensing, um sistema para monitoramento de laboratórios científicos. O sistema proposto é suficientemente genérico para ser utilizado em diferentes cenários e aplicações. O sistema é composto por nós sensores, encaminhamento de mensagens usando a abordagem *Publish/Subscribe* e armazenamento e exibição de dados em servidor centralizado.

A escalabilidade do sistema é concluída, visto que *Brokers* MQTT podem suportar milhares de conexões concorrentes e que o sistema proposto reduz o número de conexões feitas ao *Broker*. Os resultados avaliaram duas políticas de transmissão implementadas nos nós sensores localizados nas bordas da rede. As políticas implementadas foram: enviar mensagens periódicas até o nó consumidor ou enviar mensagens apenas se o valor medido ultrapassar um limiar predefinido. As medidas indicaram que a quantidade de mensagens enviadas é de fato reduzida pela transmissão por limiar, mas que se esses nós não forem capazes de dormir, o seu consumo de energia se torna superior. Os trabalhos futuros permitirão testar outras políticas, tais quais intervalos estimados por algoritmos de regressão linear implementados nos nós de sensoriamento e políticas estimadas por técnicas de *Machine Learning* implementadas no servidor, além de expandir a rede de testes já implementada.

Referências

Alqinsi, P., Matheus Edward, I. J., Ismail, N. e Darmalaksana, W. (2018). IoT-Based UPS monitoring system using MQTT protocols. Em *2018 4th International Conference on Wireless and Telematics (ICWT)*, p. 1–5.

Bochie, K. (2019). Labsensing. `https://github.com/kaylani2/labsensing`. Acessado em 19/03/2019.

Eclipse, F. (2019). Eclipse Mosquitto. `https://mosquitto.org/`.

InfluxData (2019). Influxdata. `https://www.influxdata.com/`. Acessado em 19/03/2019.

Kodali, R. K. e Gorantla, V. S. K. (2017). Weather tracking system using MQTT and SQLite. Em *2017 3rd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, p. 205–208.

Oasis (2014). MQTT Version 3.1.1. Relatório técnico, Oasis.

Saha, S. e Majumdar, A. (2017). Data centre temperature monitoring with ESP8266 based Wireless Sensor Network and cloud based dashboard with real time alert system. Em *2017 Devices for Integrated Circuit (DevIC)*, p. 307–310.

Shi, W., Cao, J., Zhang, Q., Li, Y. e Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646.