Universidade Federal de Pernambuco
Centro de Informática

Graduação em Engenharia da Computação

Estudo de Lacunas da Meta-Plataforma KNoT para IoT

Danilo Alfredo Marinho de Souza

Trabalho de Graduação

Recife
10/07/2017

Universidade Federal de Pernambuco
Centro de Informática

Danilo Alfredo Marinho de Souza

Estudo de Lacunas da Meta-Plataforma KNoT para IoT

Trabalho apresentado ao Programa de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação.

Orientador: *Prof. Kiev Santos da Gama*

Recife
10/07/2017

Agradecimentos

Primeiro agradeço a meus pais, Ailton Alfredo e Walkyria Paiva, que me ensinaram tudo e sem quem eu não seria nada. Agradeço também aos meus eternos companheiros e amigos, meus irmãos Caio Cézar e João Lucas. Agradeço à minha pequena princesa, minha irmã Maria Gabriela, cuja doçura e vivacidade muitas vezes iluminou meu caminho e à Maria Eduarda, mais uma alegria na minha vida que está por vir.

Agradeço imensamente ao Professor Kiev, que me pôs nos trilhos durante esse processo atribulado de escrever o TG, e a todos do CESAR que tiraram um tempo de seus afazeres para me ajudar: Marcela, Victor Aquino, Claudio Takahasi, Paulo Serra, entre outros. Agradeço a todos os docentes e funcionários do Centro de Informática, por contribuírem com um centro de excelência. Agradeço ao professor Carlos Mello, que mesmo com muitas ocupações, foi diligente ao cobrar nossa diligência para que tudo corresse bem.

Agradeço também a todos os colegas colaboradores e diretores da Avantia Tecnologia e Segurança, especialmente meus amigos do Avantia Labs. Agradecimento ainda mais especial a Silvio Aragão, Gervásio Neto, Saulo Batista e Marcelo Couto, que foram verdadeiros mentores para mim.

Agradeço a todos os amigos que fiz ao longo dos anos, muitos para numerar e temo esquecer algum. Vocês foram de suma importância com o companheirismo, parceria em projetos e nos momentos de leveza. Tem sido uma longa jornada, e espero que este trabalho seja um encerramento digno.

If you wish to make an apple pie from scratch, you must first invent the universe.

—CARL SAGAN

Resumo

O conceito de Internet das Coisas surge da presença ubíqua de dispositivos eletrônicos conectados à internet capazes de interagir com o ambiente, gerar e compartilhar informações. É um paradigma que potencializa aplicações como automação residencial, automação industrial, *smart cities*, serviços de telessaúde, entre outros. Em 2025, espera-se um impacto econômico de 2,7 a 6,2 trilhões de dólares. Para atingir essa marca, serão necessárias plataformas de IoT capazes de gerenciar os dispositivos e os dados gerados, fornecendo armazenamento, processamento e análise dos dados. Mineraud et al. [1] avaliou o cenário atual de IoT comparando diversas plataformas, destacando as lacunas existentes para se atingir as expectativas de usuários de aplicações IoT. Dentro deste contexto, o trabalho irá realizar uma análise da plataforma KNoT de Internet das Coisas, avaliando sua capacidade e potencial de preencher as lacunas identificadas. A avaliação do KNoT seguiu os mesmos critérios do trabalho de Mineraut et al., e o objetivo principal é posicionar o KNoT diante das lacunas apresentadas pelo trabalho.

Palavras-chave: Internet das Coisas; Plataforma; *gap analysis*; middleware; KNoT

Abstract

The idea of the Internet of Things comes from the ubiquitous presence of internet connected electronic devices capable of interacting with the environment, and of sharing and producing information. It's a paradigm that enable applications such as home and industrial automation, smart cities, telehealth services, etc. It's expected that in 2025, Internet of Things will have an economic impact of 2.7 to 6.2 trillion dollars. In order to achieve that scale, IoT platforms will be needed, as middleware that provides services like device management, data processing and analysis, e database capabilities. Mineraud et. al [1] evaluated IoT's current landscape, pointing out gaps that exist between current platforms and user's expectations of IoT applications. With that being said, this thesis will analyze the KNoT IoT Platform, evaluating its ability to fulfill the identified gaps. KNoT's evaluation will follow the same criteria as Mineraud's work.

Keywords: Internet of things; platform; gap analysis; middleware; KNoT

Sumário

Lista de Figuras

Lista de Tabelas

CAPÍTULO 1

Introdução

O conceito de Internet das Coisas surge da presença ubíqua de dispositivos eletrônicos conectados à internet e entre si, capazes de interagir com o ambiente, gerar e compartilhar informações[12]. Um grande trunfo do paradigma de Internet das Coisas é transformar objetos físicos comuns em dispositivos inteligentes, capazes de realizar ações coordenadas de sensoriamento e atuação no ambiente em que estão inseridos, além de proporcionar comunicação Machine-to-Machine (M2M) e Machine-to-Person (M2P) [13] em larga escala. Esta transformação só é viável explorando avanços consideráveis em diversas tecnologias subjacentes ao paradigma IoT (Internet of Things).

A tecnologia de rádio tornou-se ubíqua no mundo moderno, devido à necessidade cada vez mais presente de um meio de comunicação disponível em qualquer lugar e a qualquer momento. Tecnologias de comunicação sem fio tiveram papel fundamental, de tal modo que há quase um rádio para cada ser humano no planeta [14]. Avanços subsequentes reduziram o volume, custo e consumo de energia dos rádios, tornando possível a integração de rádios em praticamente qualquer objeto, e.g., sistemas RFID ou redes de sensores.

A flexibilidade e ubiquidade de dispositivos inteligentes e conectados possibilita diversas aplicações domésticas e comerciais. Por exemplo, em automação residencial é possível controlar remotamente e automatizar sistemas de climatização, abertura de garagens, iluminação e até mesmo eletrodomésticos comuns como geladeira e torradeira. Smart cities, serviços de telessaúde, automação industrial, segurança e monitoramento, etc., também são áreas de aplicação potencializadas pelo uso de IoT (Internet of Things).

O potencial do IoT é tamanho que o Conselho Nacional de Inteligência dos EUA o listou como uma das seis tecnologias disruptivas do futuro [15]. Segundo relatório do Conselho, "a partir de 2025, pontos de rede residirão em objetos do dia-a-dia - embalagens de comida, móveis, documentos, etc.". Segundo o IHS Markit [2], está estimada uma presença de 75 bilhões de dispositivos conectados em 2025, como mostrado na Figura 1.1 O impacto da presença desses dispositivos na rede já é sentido atualmente. Nos EUA, monitoramento de tráfego em redes celulares apontou para um aumento de 250% no tráfego M2M no ano de 2011 [16].

Portanto, espera-se um crescimento econômico significativo advindo das aplicações IoT, especialmente nas áreas de saúde e manufatura. Em estimativa para 2025, projeta-se um impacto econômico de 2,7 a 6,2 trilhões de dólares anuais [3]. A Figura 1.2 mostra a divisão do mercado de aplicações IoT em 2025.

Naturalmente, para a Internet das Coisas transformar este potencial cenário em realidade, muitos desafios tecnológicos devem ser superados. Em uma análise do cenário atual de Internet das Coisas, [Mineraud et. al, 2016] avaliou a maturidade de várias plataformas de IoT, destacando as lacunas presentes em suas soluções em relação a um cenário ideal do paradigma IoT.

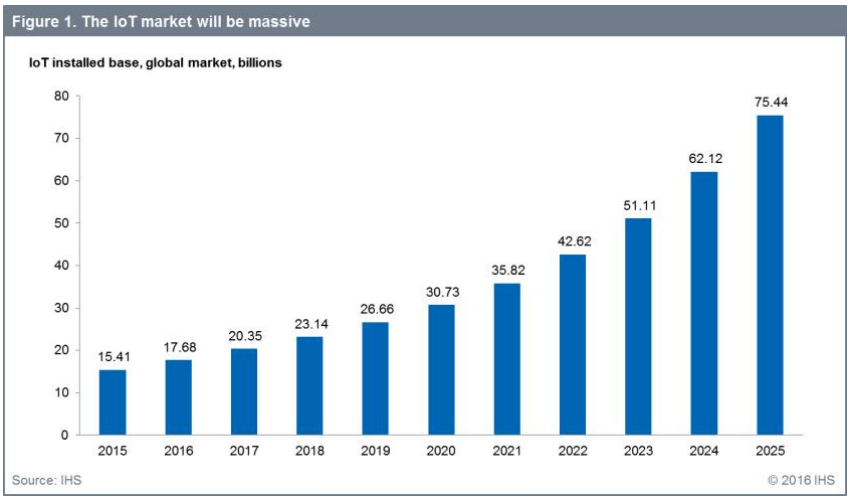


Figura 1.1 Projeção do mercado global de IoT [2]

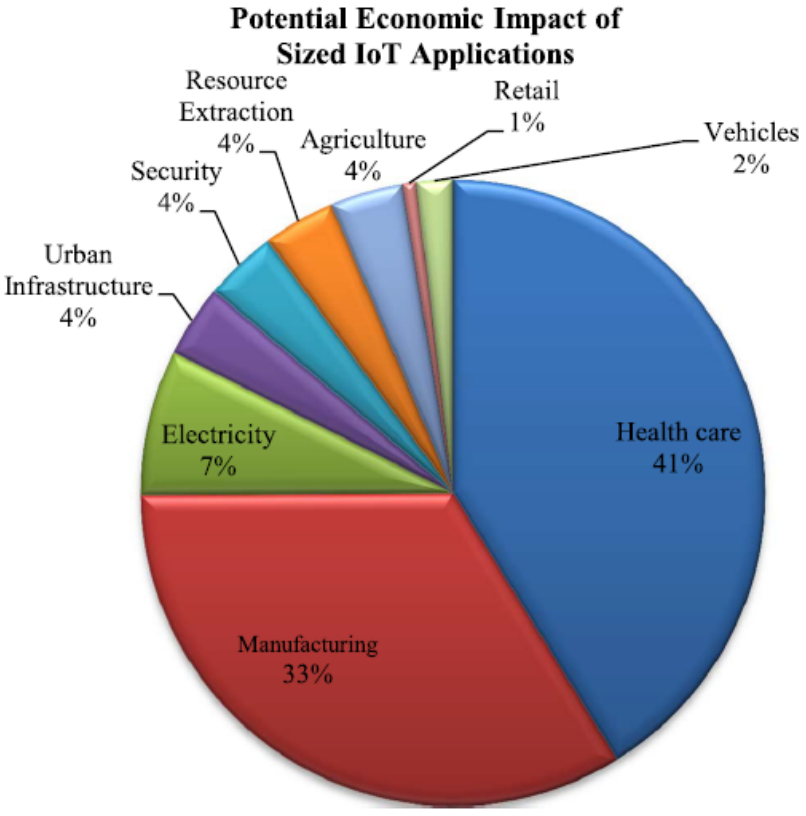


Figura 1.2 Divisão do mercado de aplicações IoT em 2025 [3]

Esta análise leva em consideração diferentes desafios das tecnologias subjacentes a uma plataforma IoT, como: suporte a dispositivos heterogêneos, privacidade e propriedade de dados, processamento e compartilhamento de dados, suporte ao desenvolvedor [1].

Neste contexto surge a meta-plataforma KNoT, desenvolvida pelo C.E.S.A.R. em parceria com a UFPE e a Avantia Tecnologia e Segurança. O objetivo do KNoT é unir e integrar diferentes plataformas de hardware e software voltadas para Internet das Coisas [17].

O objetivo deste trabalho, portanto, é situar a meta-plataforma KNoT no cenário atual de IoT, buscando avaliar o papel do KNoT em preencher as lacunas do mercado em relação aos desafios gerados pelo crescimento significativo de soluções e plataformas em IoT. Como objetivos específicos, espera-se:

- Avaliar o suporte do KNoT à inserção de dispositivos heterogêneos em um único sistema. A avaliação será focada na facilidade de integração de novos dispositivos, no método de autenticação e gerenciamento de dispositivos e nas características dos protocolos de comunicação subjacentes.
- Discutir a política de armazenamento e privacidade de dados do KNoT, em relação ao nível de privilégios dos usuários durante o acesso.
- Avaliar o suporte do KNoT a analíticos de dados.
- Avaliar como o KNoT insere semântica nos dados gerados.
- Discutir a natureza open-source do KNoT, avaliando o impacto no suporte ao desenvolvedor e no compartilhamento de soluções pela comunidade.

O trabalho será estruturado da seguinte maneira: o capítulo 2 fará uma contextualização e descrição do estado da arte em Internet das Coisas. No capítulo 3 será feita uma análise comparativa de plataformas de IoT. No capítulo 4 será feita uma descrição detalhada da plataforma KNoT, explorando sua arquitetura básica e seus componentes. No capítulo 5, será feita uma análise do KNoT baseada nos critérios do capítulo 3. O capítulo 6 concluirá o trabalho, apresentando considerações finais e trabalhos futuros.

CAPÍTULO 2

Contexto e Estado da Arte

2.1 Dispositivos IoT

Os dispositivos IoT, ou as "coisas" da Internet das Coisas, estão no cerne do paradigma. Além do propósito principal de conectividade entre diversos dispositivos, espera-se uma maneira de descobrir estes dispositivos numa rede e utilizar seus serviços de maneira encapsulada e abstrata, sem necessitar do usuário o conhecimento de protocolos e tecnologias subjacentes aos dispositivos. A arquitetura SOA para plataformas IoT foi projetada para lidar não só com estas necessidades, mas também para suportar o dinamismo de um sistema IoT, em que há mobilidade dos dispositivos na rede e constante evolução da tecnologia por trás dos dispositivos [12] [18].

Entretanto, algumas tecnologias chave para realizar o cenário ideal de IoT ainda estão a ser desenvolvidas, como identificação universal para dispositivos IoT e integração, interpretação e troca de dados semânticos [19]. [Xiao et. al, 2014] identifica quatro desafios que impedem o desenvolvimento dessas tecnologias: 1) Larga escala de cooperação, uma vez que espera-se milhares e até milhões de dispositivos conectados. 2) Heterogeneidade de dispositivos. 3) Dispositivos com configuração desconhecida e/ou heterogêneas. 4) Conflitos semânticos [20].

As próximas seções serão dedicadas a explorar a complexidade e diversidade de dispositivos IoT, a partir dos seguintes aspectos:

1. Poder computacional
2. Tecnologias de comunicação
3. Privacidade e segurança

2.1.1 Poder computacional

Devido à natureza das aplicações IoT, espera-se que um dispositivo IoT tenha:

- Mobilidade, para aplicações como monitoramento de cargas em transporte ou monitoramento de sinais vitais de um paciente.
- Vida útil prolongada e autonomia, pois em alguns casos o dispositivo será instalado em locação remota ou de difícil acesso, exigindo um dispositivo durável que não precise ser recarregado com frequência.

Tabela 2.1 Comparação de plataformas de hardware em relação ao poder computacional [11]

Plataforma	CPU	GPU	Clock	Tamanho	Memória	RAM
SparkFun Blynk Board	Tensilica L106 32-b	Não	26 MHz	51mm x 42 mm	4MB	128Kb
Arduino Yun	ATmega32u4	Não	16MHz	73mm x 53mm	32KB	64MB DDR2
Raspberry Pi 3	ARM Cortex-A53 64-b Quad-Core	VideoCore IV @ 300/400MHz	1.2 GHz	85mm x 56mm	Micro-SD	1GB LPDDR2
cloudBit	Freescale i.MX233 (ARM926EJ-S core)	Não	454 MHz	55mm x 19mm	Micro-SD	64MB
Photon	STM32F205 ARM Cortex M3	Não	120MHz	36.5mm x 20.3mm	1MB	128KB
BeagleBone Black	AM335x ARM Cortex-A8	PowerVR SGX530	1GHz	86mm x 56mm	4GB, micro-SD	512MB
Pinoccio	ATmega256RFR2	Não	16MHz	70mm x 25mm	256KB	32KB
UDOO	Freescale i.MX 6 ARM Cortex-A9 e Atmel SAM3X8E ARM Cortex-M3	Vivante GC 2000 para 3-D + GC 355 para 2-D + GC 320 para 2-D	1GHz	110mm x 85mm	Micro-SD	1GB
Samsung Artik 10	ARM A15x4 e A7x4	Mali-T628 MP6 core	1.3 ou 1 GHz	39mm x 29mm	16GB	2GB

- Tamanho reduzido, pois a ideia do paradigma IoT é embarcar inteligência e comunicação em objetos e ambientes cotidianos. Dessa maneira, o dispositivo IoT deve estar integrado ao ambiente de modo a passar relativamente despercebido.

Placas integradas e sistemas embarcados voltados para IoT estão cada vez mais sendo a primeira escolha da indústria para prototipação e desenvolvimento de dispositivos IoT [11]. A tabela 2.1 compara as especificações de processamento e memória de algumas das plataformas mais relevantes no mercado. Pode-se observar na tabela dois tipos gerais de plataformas de hardware para IoT: microcontroladores com poder computacional bem reduzido e microcomputadores relativamente mais robustos, mas com especificações modestas comparando com computadores pessoais modernos.

2.1.2 Tecnologias de Comunicação

O papel da tecnologia de comunicação em IoT é conectar dispositivos heterogêneos de modo a fornecer algum serviço específico. Com a proporção de rádios e humanos tornando-se próxima de 1 [14] e com o número gigantesco de dispositivos conectados estimados para um futuro próximo [2], espera-se tecnologias capazes de atender as demandas do IoT. Ou seja, espera-se tecnologias low power, de baixo custo, móveis e tolerantes a ambientes ruidosos e/ou com baixa conectividade.

As tecnologias a seguir não representam exaustivamente o universo de tecnologias de comunicação sem fio, mas são exemplos relevantes para o contexto do IoT por:

1. Ilustrarem a diversidade de redes em que se pode embarcar a IoT, como por exemplo redes WAN, PAN e WLAN.

2. Representarem tecnologias que foram importantes para a ascensão do paradigma IoT ou tecnologias promissoras que podem contribuir para seu avanço.

2.1.2.1 RFID

O conceito fundamental do RFID (Radio Frequency Identification, ou identificação por radiofrequência em português) é um M2M básico, entre a tag e o leitor. Uma tag RFID identifica um objeto, e o leitor consulta o objeto enviando um sinal que energiza a tag, que por sua vez retorna ao leitor suas informações de identificação. Uma tag RFID é um microchip combinado com uma antena, cuja fonte de energia pode ser o leitor (tag passiva) ou uma bateria (tag ativa)[20]. O leitor pode ser qualquer dispositivo com rádio transmissor e receptor e um sistema computacional capaz de acessar um banco de dados para identificar o objeto lido. As Figuras 2.1 e 2.2 ilustram a comunicação entre tags de diferentes tipos e o leitor.

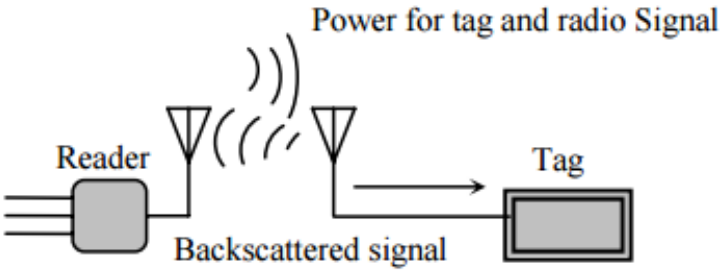


Figura 2.1 Comunicação entre tag passiva e leitor [4]

A tecnologia RFID é comumente utilizada em aplicações que necessitam de rastreio e identificação de assets de interesse. Por exemplo, a Boeing utiliza caixas com tags RFID para identificar equipamentos aeronáuticos [21]. Tags passivas podem ser implantadas em animais para rastreio [4]. No Apple Pay, antenas RFID em iPhones ou no Apple Watch permitem ao usuário fazer compras sem tirar a carteira do bolso [22].

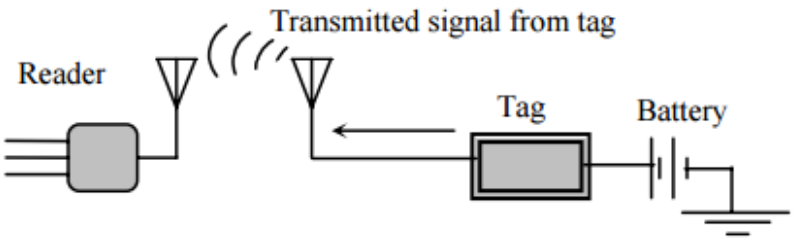


Figura 2.2 Comunicação entre tag ativa e leitor [4]

2.1.2.2 Bluetooth Low Energy

A tecnologia Bluetooth já é amplamente utilizada em dispositivos sem fio como mouses, teclados, headsets, sistemas de som automotivos e smartphones. Considerando apenas o uso do bluetooth em smartphones, cujo número de usuários chegou a 2.6 bilhões em 2015 [23], já há evidências de que esta é uma tecnologia ubíqua e com forte penetração no mercado.

O Bluetooth Low Energy (BLE) é um avanço na tecnologia bluetooth tradicional, sendo projetado para baixo consumo de energia e baixo custo [24]. Devido a essas características, tornou-se uma tecnologia muito atraente para aplicações de sensoriamento com beacons e monitoramento, com uma projeção de 4.9 bilhões de dispositivos bluetooth a ser lançados no mercado em 2018 [25]. Um protocolo relevante para BLE é o iBeacon [26]. Um dispositivo dotado deste protocolo, como o da Figura 2.3 realiza transmissões periódicas de dados de identificação e de sensores.

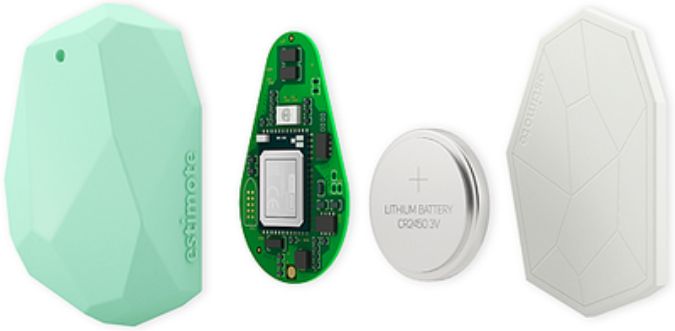


Figura 2.3 Beacon Estimote [5]

2.1.2.3 ZigBee

O ZigBee é uma suíte de protocolos de comunicação para redes WPAN, baseada no IEEE 802.15.4. O principal objetivo do padrão IEEE 802.15.4 é oferecer uma rede de baixo custo,

curto alcance e de baixo consumo energético para aplicações que não demandam alta vazão de dados [6]. A Figura 2.4 sumariza as características de uma rede WLAN comum, uma rede WPAN Bluetooth e o padrão LR-WPAN proposto.

	WLAN	BT-based WPAN	Low-rate WPAN
Range	~100 m	~10–100 m	10 m
Data throughput	~2–11 Mb/s	1 Mb/s	<0.25 Mb/s
Power consumption	Medium	Low	Ultra low
Size	Larger	Smaller	Smallest
Cost/complexity	> 6	1	0.2

Figura 2.4 Comparação entre tipos de redes sem fio [6]

A ZigBee Alliance, um consórcio de empresas de tecnologia que especificou o ZigBee, oferece uma série de padrões de desenvolvimento de aplicações IoT, em um esforço para mitigar os problemas de interoperabilidade de dispositivos.

As características do ZigBee o fazem muito útil para construção de Redes de Sensores sem Fio, que podem ser empregados em diversas aplicações, como [27]:

- Aplicações militares: monitorar forças inimigas ou aliadas, monitoramento do campo de batalha e detecção de ataques nucleares ou bioquímicos.
- Aplicações ambientais: monitoramento de incêndios florestais, monitoramento de fauna e flora.
- Saúde: monitoramento remoto de dados de pacientes, rastrear pessoas em hospitais.
- Automação residencial.

2.1.2.4 WiFi

A tecnologia Wi-Fi, baseada no padrão IEEE 802.11, já tornou-se parte do dia-a-dia nos grandes centros urbanos. O uso comercial e residencial da tecnologia para se conectar à Internet é ubíquo, com aproximadamente metade da população mundial usuária da internet [28]. Usualmente, as exigências do usuário comum de Internet não se aplicam à Internet das Coisas, uma vez que as aplicações multimídia e de navegação na Web demandam alta vazão de dados e não há tantas restrições de custo, tamanho e consumo de energia.

Entretanto, existem diversas propostas de plataformas e aplicações que utilizam do WiFi no contexto de IoT. Um exemplo é o ESP32 [29], um System-on-Chip integrando Bluetooth 4.0, WiFi 2.4GHz e uma unidade microcontroladora, que atende as demandas de ser baixo custo e baixo consumo, mas mantendo uma alta vazão de dados. Aplicações de larga escala contendo redes de sensores, como smart grids [30], podem necessitar da alta vazão de dados oferecida pelo WiFi.

2.1.2.5 LoRa™

O LoRa™ e o LoRaWAN são, respectivamente, especificações das camadas física e de enlace e da camada de rede de um protocolo de comunicação sem fio caracterizado pela capacidade de longo alcance de transmissão, na faixa de quilômetros de distância, e baixa taxa de dados, na faixa de alguns kbps. É uma tecnologia que dá suporte a rede LPWAN, i.e., *Low Power Wide Area Network* ou Rede de longa distância de baixo consumo de energia [7]. As principais características de uma rede LPWAN são [31]:

- O dispositivo deve operar com o mínimo de consumo de bateria possível. Idealmente, a troca de bateria deve ser muito rara. Como se espera uma rede LPWAN com milhões de dispositivos, devido ao longo alcance de transmissão, a troca de bateria de todos eles seria bastante custosa.
- Limitações econômicas são relevantes. A implementação de um dispositivo em uma rede deve ser de baixo custo, para estimular o amplo uso e diminuir custos de manutenção. Isso implica em arquiteturas e protocolos simples, pois a complexidade destes dispositivos torna-se limitada.
- Pela característica de baixa transmissão de dados e a demanda por baixo consumo, o dispositivo só deve ficar ativo durante a transmissão e recepção de dados. A implicação disso é a restrição de protocolos de rede sincronizados ou em malha. Dá-se, então, preferência ao ALOHA [32].
- Segurança na transferência de dados.
- Uma rede LPWAN geralmente possui baixa mobilidade, mas os dispositivos podem estar em ambientes com muitas mudanças na características do canal, como uma estrada. Dessa maneira, deve haver uma certa robustez na modulação.
- A localização dos objetos é uma informação valiosa, preferencialmente sem uso de GPS, que consome muita energia.

A Figura 2.5 ilustra a arquitetura básica do LoRa™. Uma arquitetura em estrela é utilizada, de modo a economizar mais energia, pois nesta arquitetura um dispositivo, *end node*, realiza suas operações sem se preocupar com a presença de outros dispositivos. Os dispositivos simplesmente transmitem os dados, que podem ser recebidos por um ou mais gateways. Estes gateways, por sua vez, remetem os dados a um servidor Cloud, responsável por realizar a análise de dados e fornecer sua visualização para aplicações.

2.2 Processamento e Compartilhamento de Dados

Com o escopo esperado da IoT, teremos milhares de dispositivos conectados transmitindo e recebendo dados em uma única aplicação. Naturalmente, visualizar estes dados de maneira bruta é impraticável e ineficiente. Portanto, para extrair valor dos dispositivos IoT, é necessário

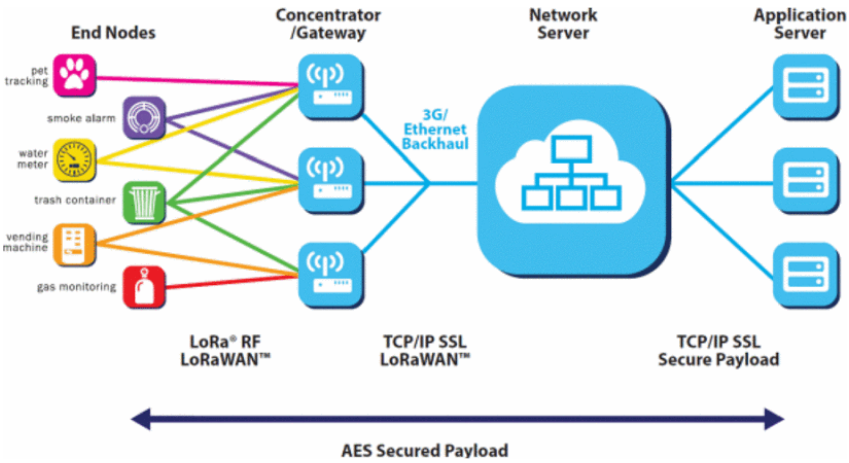


Figura 2.5 Arquitetura da rede LoRa^{TM} [7]

desenvolver métodos e plataformas capazes de lidar com uma quantidade massiva de dados vindo de fontes heterogêneas. A seguir, veremos como é possível lidar com os dados da IoT a partir de duas demandas básicas: 1) compreender a informação representada pelo dado e 2) processar e analisar os dados.

2.2.1 Semântica na Internet das Coisas

Ao descrever a Web Semântica, Berners-Lee apontou que "novos desenvolvimentos irão alavancar novas funcionalidades significativas uma vez que máquinas estão cada vez mais capazes de entender e processar dados"[33]. No contexto de Internet das Coisas, isso significa que os dispositivos IoT enviarão informações em um formato semântico ao invés do dado bruto. Com o significado do dado codificado diretamente na mensagem, o receptor não precisa ter conhecimento específico do dispositivo IoT, podendo utilizar o dado de maneira generalizada [8].

Aplicar semântica em IoT, entretanto, introduz cenários que não acontecem na Web Semântica tradicional, uma vez que os dispositivos IoT costumam ter recursos mais limitados, como processamento, memória, vazão de dados e energia. O principal desafio é propor um formato de dados que seja compatível com os modelos da Web Semântica, e.g. JSON, XML ou RDF, e que acrescente o mínimo de *overhead* possível. Su et al. [8] avaliou diversos formatos de dados propostos para IoT em relação ao consumo de recursos dos dispositivos. Os principais resultados da pesquisa estão nas Figuras 2.6, 2.7 e 2.8 a seguir:

Os resultados descritos acima demonstram o impacto do formato semântico de dados na performance geral de um dispositivo IoT.

2.2.2 Big Data e Internet das Coisas

Um volume cada vez maior de dados está sendo gerando, como consequência de processos de negócios automatizados, monitoramento de atividade de usuários de serviços, sensores, finanças, entre outros motivos. As redes sociais também gerou verdadeiros registros da vida de cada usuário, com *posts* detalhando atividades, eventos idos, gostos pessoais, fotografias e opiniões

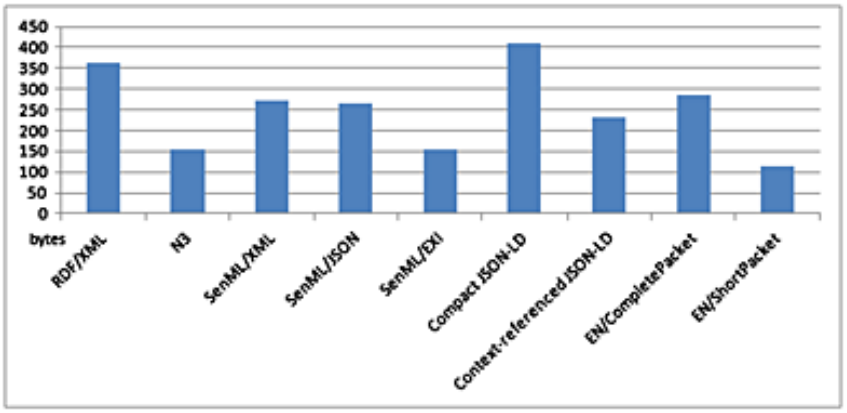


Figura 2.6 Comparação de tamanho de pacotes de diferentes formatos de dados [8]

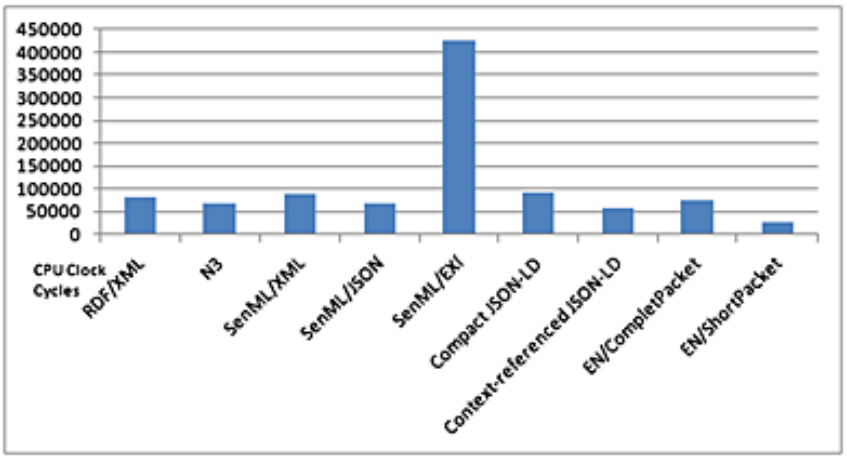


Figura 2.7 Comparação dos ciclos de CPU necessários para gerar mensagens nos diferentes formatos de dados [8]

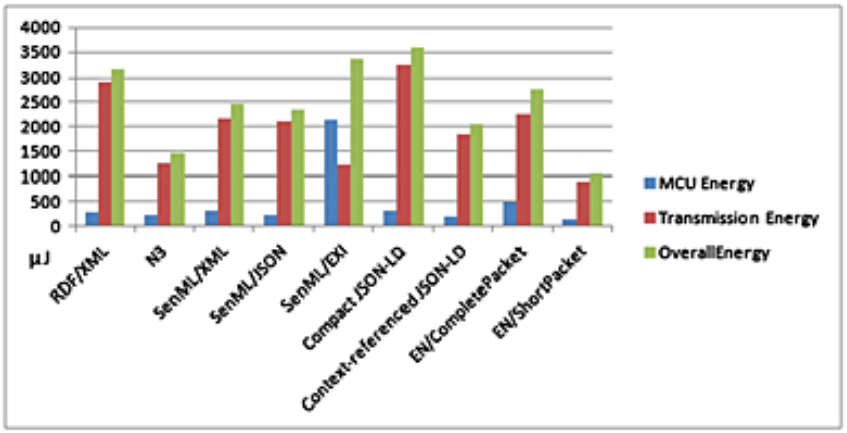


Figura 2.8 Comparação do consumo de energia de diferentes formatos de dados [8]

em geral [34]. A este conjunto massivo de dados dá-se o nome de Big Data [35]. O nome indica os desafios impostos às tecnologias de hardware e software usuais em relação à capacidade de armazenar, gerenciar e processar essa quantidade de dados em um tempo razoável.

A computação em nuvem surge como uma potencial solução para os desafios trazidos pelo Big Data. A computação em nuvem consiste de um modelo de acesso *on-demand* a uma coleção de recursos computacionais configuráveis, e.g. redes, servidores, armazenamento, aplicações e serviços [36]. Dessa maneira, a computação em nuvem permite ao desenvolvedor escalar sua aplicação como for mais conveniente e de maneira confiável.

Como a Internet das Coisas envolve milhares de dispositivos conectados gerando e compartilhando informações, o cenário do Big Data em IoT é inevitável. Portanto, o uso de recursos da computação em nuvem para soluções IoT é igualmente inevitável.

Plataformas como Apache Hadoop e SciDB foram projetadas para analíticos de Big Data, entretanto as demandas de Big Data de IoT são grandes demais para serem atendidas pelas ferramentas disponíveis. Uma abordagem viável para o Big Data no IoT é filtrar os dados de interesse. Técnicas como Análise de Componentes Principais (PCA), redução de dimensionalidade, seleção de características e métodos de computação distribuídas são apropriadas para lidar com este problema [37].

Embora a computação em nuvem seja importante para lidar com o Big Data no IoT, Al-Fuqaha et. al aponta alguns desafios que o cenário específico da IoT impõe [9]:

- Sincronização: A sincronização de diferentes plataformas de computação em nuvem é um desafio, pois existe a demanda de fornecer serviços em tempo real por cima das plataformas.
- Padronização: A interoperabilidade entre diferentes serviços é um desafio significativo.
- Balanceamento: As diferentes infra-estruturas de um ambiente nuvem e o ambiente IoT impõe certos desafios.
- Confiabilidade: Como dispositivos IoT possuem diferentes níveis de segurança, manter uma uniformidade na segurança do serviço em nuvem é um desafio.

Outra importante questão específica ao cenário de IoT é de onde vem os dados. O Big Data do IoT é gerado principalmente nos nós mais remotos da rede, que são os dispositivos IoT. A transmissão dessa quantidade imensa de dados requer bastante banda. Isso pode ser mitigado com o pré-processamento e armazenamento localizado de dados, entretanto os dispositivos IoT são limitados demais para oferecer essas funcionalidades. Propôs-se, então, o modelo de *Edge Cloud*, ou *Fog*, uma arquitetura de nuvem híbrida, com o objetivo de entregar serviços com baixa latência e com eficiência no uso de banda ao usuário final. A *Edge Cloud* consiste de, então, uma interface entre os nós remotos da rede e as centrais de dados, permitindo aplicações com restrições de tempo real com mais facilidade [38].

A Figura 2.9 ilustra o papel das centrais de dados na nuvem e da *Edge Cloud* na entrega de serviços IoT ao usuário final.

Al-Fuqaha et. al [9] lista algumas funcionalidades da computação em *fog* úteis aos desenvolvedores de IoT:

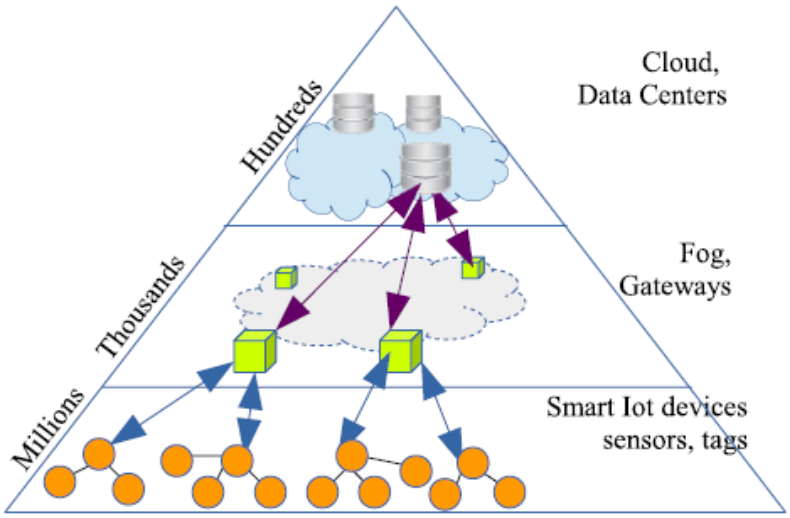


Figura 2.9 Esquema básico de uma fog, ou *edge cloud* no fornecimento de serviços IoT [9]

- Localização: Os recursos da *fog* encontram-se fisicamente mais próximos dos nós remotos, diminuindo a latência na transmissão de dados.
- Distribuição: É possível distribuir o processamento de dados em diversas *fogs*.
- Escalabilidade: A *Fog* permite melhor escalibidade da aplicação IoT, uma vez que é bem menos custoso implementar uma nova *fog* para aliviar a carga de processamento do que um data center na nuvem.
- Tempo real: A *Fog* melhora a performance de sistemas de tempo real, ao diminuir a latência de transmissão.
- Pré-processamento de dados nas camadas intermediárias

2.3 Plataformas IoT

A adoção em massa da IoT presume o surgimento de um ecossistema sustentável em que empresas podem coletivamente criar, distribuir e consumir aplicações IoT [39]. Tal ecossistema pode ser definido como "uma rede de compradores, fornecedores e desenvolvedores de serviços e produtos relacionados" [40]. Para poder explorar ao máximo o potencial deste ecossistema, uma coleção compartilhada de recursos é necessária, o seja, um conjunto de componentes, módulos e protocolos reutilizáveis que serão compartilhados por múltiplas aplicações.

Inserido num ecossistema como esse, um tipo de aplicação IoT que pode surgir seria desenvolvida de maneira ad hoc e oportunística, utilizando serviços web e streams de dados fornecidas por terceiros que rodam sobre objetos inteligentes. Aplicações deste tipo, utilizando por exemplo uma abordagem de web mashup [41], fariam do IoT um Sistema de Sistemas (SoS ou System-of-Systems). Uma definição de SoS vem de Maier e Rechtin [42], em que um SoS é

tido como uma composição de sistemas em que os sistemas constituintes são individualmente detectados, selecionados e compostos em tempo de execução para construir um sistema resultante mais complexo. Cada sistema componente seria gerenciado de maneira independente, resultando em um sistema final mais complexo e maior que a soma de suas partes.

Gerenciar a construção de um complexo Sistema de Sistemas envolve o uso de algum tipo de middleware que fornece serviços capazes de atender requisitos não funcionais de tais sistemas, como [43] [44]:

1. Escala. Devido ao escopo esperado do IoT, é necessário lidar com tarefas trabalhando sobre milhões de dispositivos, com restrições de memória, processamento e tempo.

2. Alta heterogeneidade de hardware e software, exigindo mecanismos de interoperabilidade em diversos domínios de aplicação.

3. Dinamismo e mobilidade da rede. A topologia de uma infra-estrutura IoT pode ser bastante dinâmica, com nós de rede móveis que podem entrar e sair da rede intermitentemente, ou mesmo dispositivos com localização incerta. Portanto, deve ser fornecido algum serviço de descoberta dinâmica de dispositivos, com flexibilidade para suportar as mudanças de topologia.

4. Gerenciamento de Big Data: embora um dispositivo IoT transmita poucos dados, milhões de dispositivos coletivamente geram a um cenário de Big Data. Além disso, existem questões de segurança e privacidade de dados neste contexto IoT que devem ser trabalhadas.

Mineraud et. al [1] define uma plataforma IoT como o middleware e a infra-estrutura que permite aos usuários finais interagir com objetos inteligentes. Uma plataforma de IoT pode ser utilizada, portanto, para facilitar o desenvolvimento de aplicações e integração de dispositivos com tecnologias subjacentes distintas. Uma plataforma IoT forneceria funcionalidades como: fornecimento de serviços para gerenciar diversos dispositivos, armazenamento e recuperação de dados, analíticos de dados, disparo de alertas e alarmes, entre outros [45].

Como uma plataforma IoT tem necessidades específicas, surge a necessidade de uma arquitetura bem definida. Embora ainda não haja consenso na academia ou no mercado em relação às especificações de tal arquitetura, nota-se que boa parte delas segue a abordagem SOA (Service Oriented Architecture). Neste contexto, serviços são componentes de software com interfaces bem definidas, que devem funcionar independente da linguagem de programação ou plataforma subjacente e devem ser auto-contidos, i.e., cada serviço tem uma finalidade específica [46], permitindo reusabilidade de componentes, integração mais robusta e facilidade de implementação.

Atzori et. al [12] propõe um modelo de arquitetura IoT baseada em SOA, baseando-se nas arquiteturas propostas anteriormente, como mostrado na Figura 2.10. Segue, então, uma breve descrição top-down das camadas propostas:

- Camada de aplicações: Embora não faça parte, estritamente falando, do middleware, essa camada explora todas as funcionalidades da plataforma. Através de protocolos de serviço web padrão, as aplicações entregam ao usuário final o valor da plataforma.

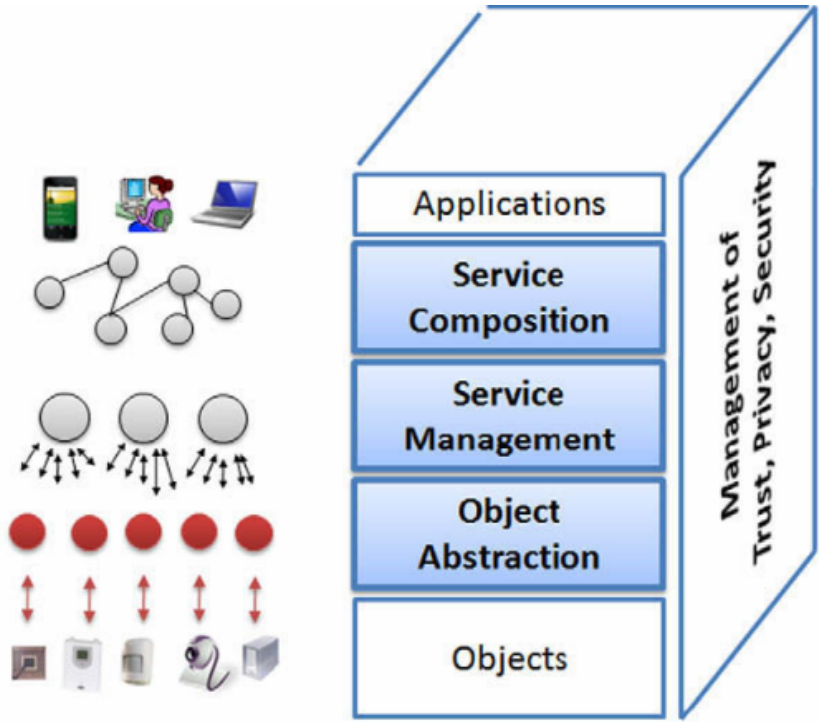


Figura 2.10 Arquitetura SOA para plataformas IoT [9]

- Camada de composição de serviço: Nessa camada, não existe a noção de dispositivos, apenas de serviços. Dessa maneira, o que a camada oferece é a composição de serviços modularizados para montar uma aplicação de propósito específico. Tem-se, então, um fluxo de trabalho de processos de negócios, que podem ser descritos através de linguagens padronizadas como o BPEL (Business Process Execution Language) [46].

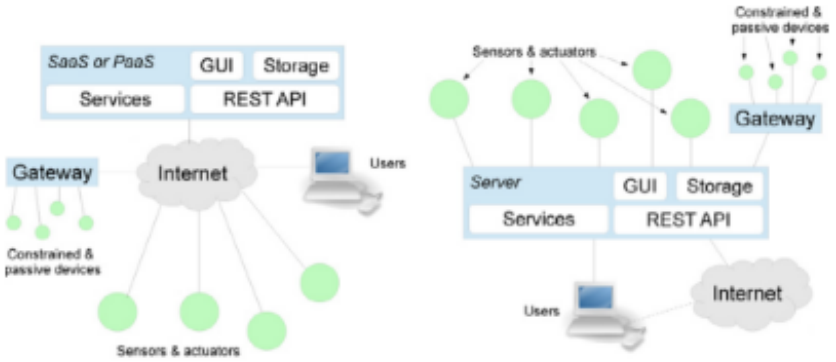
- Camada de gerenciamento de serviços: Esta camada fornece as funcionalidades de cada objeto gerenciado pela plataforma, e.g., descobrimento dinâmico de objetos, monitoramento de status, configuração, agendamento de atividades, etc. Além disso, essa camada associa cada objeto com um catálogo de serviços disponíveis.

- Camada de abstração de objetos: Como a Internet das Coisas prevê uma vasta gama de objetos fornecendo funções específicas à sua maneira, é esperado algum intermediário capaz de padronizar a maneira como esses dispositivos são detectados e acessados por uma plataforma. Dados de um dispositivo IoT podem ser transmitidos de diversas maneiras, por RFID, Wi-Fi, ZigBee, Bluetooth e assim por diante. Nesta camada, então, é fornecido um wrapper dividido em duas sub-camadas: uma responsável por abstrair as especificidades de cada dispositivo em uma interface de serviço web unificada, e outra responsável por traduzir a interface para os comandos específicos aceitos por cada dispositivo.

- Objetos: Nesta camada estão as Coisas da Internet das Coisas. Dispositivos sensores e atuadores com um módulo de comunicação capaz de acessar, direta ou indiretamente, a

Web. É recomendado estabelecer mecanismos plug-in-play padronizados para configurar dispositivos heterogêneos [47].

Podemos ainda classificar plataformas IoT em relação à infra-estrutura sobre a qual a plataforma reside. Isto é, os serviços da plataforma podem ser fornecidos através da nuvem, ou de um servidor local, como mostra a Figura 2.11.



(a) Cloud-based platform (b) Local platform

Figura 2.11 Tipos de plataforma IoT [1]

O gateway descrito acima age como um intermediário para dispositivos limitados e/ou passivos incapazes de se conectar diretamente à internet. Muitas plataformas definem um gateway proprietário compatível com determinados tipos de dispositivos IoT e protocolos. O gateway pode desempenhar as seguintes funções numa plataforma IoT: fornecer conectividade a dispositivos, tradução de protocolos, filtragem e processamento de dados, etc.[48]. Entretanto, os gateways dependerão do suporte dos desenvolvedores da plataforma para dar conta de novos protocolos e tipos de dispositivos IoT que surgem ao longo do tempo.

CAPÍTULO 3

Análise dos Gaps em Plataformas de Internet das Coisas

O objetivo deste capítulo é avaliar o cenário do IoT em relação à capacidade das plataformas IoT de lidar com os desafios emergentes dos desenvolvimentos atuais de tecnologias IoT. O escopo da análise é derivado do trabalho de Mineraud et. al [1], que buscou diversas soluções de plataformas IoT e fez uma análise compreensiva das lacunas encontradas. Uma das principais contribuições do trabalho foi definir características consideradas fundamentais para que a plataforma IoT supra as demandas dos desenvolvedores de aplicações.

A tabela 3.1 apresenta um conjunto de plataformas IoT apontando características importantes para atender as expectativas de usuários e desenvolvedores de aplicações. Um código de cor foi adicionado a algumas colunas para deixar a informação na tabela mais acurada. A cord verde indica que a característica está de acordo com as expectativas dos usuários, enquanto a cor vermelha indica o contrário. O laranja indica uma qualificação intermediária da característica. A coluna 2 indica que tipos de dispositivos são suportados pela plataforma, com ou sem Gateway. A coluna 3 descreve o tipo de plataforma, PaaS (*Platform as a service*, ou plataforma como um serviço) ou SaaS (*Software as a service* ou software como um serviço). A plataforma também pode ser M2M caso seja esse seu foco primário.

A coluna 4 refere-se à arquitetura da plataforma, que podem ser centralizadas, distribuídas ou hospedada na nuvem. A coluna 5 indica se a plataforma é proprietária ou open-source, enquanto as colunas 6, 7 e 8 indica, respectivamente, a disponibilidade de uma API REST, de controle de acesso aos dados e de mecanismos de descobrimento de serviços.

Dadas essas características de plataformas IoT, é possível identificar múltiplas lacunas nas funcionalidades implementadas destas plataformas. Dessa maneira, é apresentada uma análise das lacunas observadas, que pode ser sumarizada pela tabela 3.2. O objetivo da análise é avaliar a maturidade das soluções atuais identificando suas lacunas nos seguintes aspectos: i) suporte a dispositivos heterogêneos, ii) mecanismo de controle de acesso aos dados, iii) integração de técnicas de compartilhamento e processamento de dados e (iv) suporte aos desenvolvedores de aplicações.

3.1 Integração de Dispositivos Heterogêneos

Uma das características do IoT é a diversidade de dispositivos, sensores, atuadores, plataformas de hardware, software e protocolos de comunicação que compõem os objetos inteligentes conectados. Como não existe um padrão único de comunicação, diferentes dispositivos de diferentes fabricantes irão implementar diferentes pilhas de protocolos de comunicação. Dessa

Tabela 3.1 Plataformas IoT avaliadas e suas características [1]

Plataforma	Suporte a dispositivos heterogêneos	Tipo	Arquitetura	Open-source	REST	Controle de acesso	Descobrimento de serviços
Carriots	Sim	PaaS	Cloud	Não	Sim	Acesso seguro	Não
IoT-framework	Sim	Servidor	Centralizado	Licença Apache 2.0	Sim	Armazenamento local	Sim
IFTTT	Sim	SaaS	Centralizado	Não	Não	Sem armazenamento	Limitado
Kahvihub	Sim	Servidor	Centralizado	Licença Apache 2.0	Sim	Armazenamento local	Sim
LinkSmart	Dispositivos embarcados	P2P	Descentraliza	LGPLv3	Não	Armazenamento local	Sim
NinjaPlatform	Sim, com gateway	PaaS	Cloud	Apenas gateway	Sim	OAuth2	Não
Node-RED	Sim	Servidor	Centralizado	Licença Apache 2.0	Não	Privilégios baseados em usuário	Não
OpenRemote	Dispositivos domésticos	Servidor	Centralizado	Licença pública Affero GNU	Sim	Armazenamento local	Não
realTime.io	Sim, com gateway	PaaS	Cloud	Não	Sim	Acesso seguro	Não
SensorCloud	Não	PaaS	Cloud	Não	Sim	n.a.	Não
TempoDB	Não	PaaS	Cloud	Não	Sim	Acesso seguro	Não
ThingSpeak	Sim	Servidor	Centralizado	GNU GPLv3	Sim	2 níveis	Limitado
ThingWorx	Sim	M2M PaaS	Cloud	Não	Sim	Privilégios baseados em usuário	Sim
Xively	Sim	PaaS	Cloud	Biblioteca sim (BSD 3-clause), plataforma não	Sim	Acesso seguro	Sim

Tabela 3.2 Análise de lacunas nas plataformas IoT [1]

Categoria	Status	Expectativas	Lacunas	Problemas	Recomendações
Suporte a dispositivos heterogêneos	Plataformas exigem HTTP ou gateway	Dispositivos devem ser integrados sem gateway	Suporte a dispositivos heterogêneos	Interações heterogêneas	Utilizar protocolos padrões (MQTT, CoAP, etc.)
		Recursos unificados e usabilidade simplificada	Modelos padrões de dispositivos IoT	Padronização de protocolos	Integração de protocolos de segurança
			Autenticação segura no gerenciamento de dispositivos		
Acesso de dados	Normalmente o usuário tem propriedade dos dados mas com políticas muito simples de privacidade	Controle total ao dono do dado	Manipulação do dado nos dispositivos	Segurança no armazenamento	Mecanismos disponíveis ao proprietário dos dados que o permitam controlar totalmente o acesso
		Armazenamento local	Auto armazenamento	Limitação dos dispositivos para armazenamento e segurança	
Processamento e compartilhamento de dados	Formato de compartilhamento não uniforme	Formato de dados uniforme para múltiplas plataformas	Processamento de dados não muito bem integrados a plataformas	Acesso complexo aos dados devido a heterogeneidade	Catálogos de dados com indexação semântica
			Falta de modelos e formatos de dados eficientes	Fusão eficiente de streams de dados distintas	Modelo de dados uniforme e interoperável
	Compartilhamento via APIs REST não uniformes	Catálogos de dados	Analíticos apenas em plataformas cloud	Dispositivos IoT possuem poder computacional limitado	Integração de tecnologias de processamento de dados nas plataformas
		Analíticos de edge	Não há catálogos de dados		Fogs para analíticos de edge
Suporte ao desenvolvedor	API REST para acessar os dados dos dispositivos	Uso de uma API comum para desenvolvimento em múltiplas plataformas	APIs não uniformes	Requer padronização de interações entre aplicações IoT	Plataformas IoT devem fornecer SDKs e APIs que maximizem a reusabilidade de seus serviços
	Aplicações não são feitas para compartilhamento (exceto IFTTT)		Presença limitada de SDKs	Não há uma loja de aplicações IoT	

maneira, uma plataforma de IoT possui um valor proporcional à sua capacidade de integrar diferentes dispositivos. Uma plataforma ideal, portanto, ofereceria diversas opções de protocolos. Dispositivos com conectividade limitada necessitariam ser intermediados por um gateway, preferencialmente um gateway totalmente controlado pelo usuário.

Vários protocolos de comunicação foram propostos para a Internet das Coisas, levando em conta a limitação natural de memória e processamento de dispositivos IoT, como CoAP, MQTT, Z-Wave e LWM2M. Algumas plataformas oferecem suporte a alguns destes protocolos, como *OpenRemote* (KNX, Z-Wave, etc.), *LinkSmart* (ZigBee) e *ThingWorx* (MQTT). Entretanto, várias plataformas de IoT não oferecem suporte a estes protocolos, dando preferência ao tradicional HTTP, um protocolo que exige dispositivos mais poderosos. Este é o caso de plataformas como *SensorCloud* ou *TempoDB*. De modo geral, a interoperabilidade de dispositivos heterogêneos nas plataformas IoT atuais é garantida com o uso de um gateway extensível, proprietário ou open source, ou com o suporte a um conjunto limitado de protocolos padrões. Por exemplo, a plataforma NinjaPlatform fornece um gateway inteiramente open-source enquanto o gateway da plataforma *realTime.io* é proprietário.

Como se pode observar, uma grande lacuna nas plataformas IoT é a falta de padronização dos dispositivos IoT e seus protocolos de comunicação subjacentes. Atingir o ponto ideal de interoperabilidade irá envolver um esforço conjunto na definição de modelos de dispositivos IoT e o suporte às limitações destes dispositivos, seja na forma de protocolos otimizados como MQTT e CoAP, ou na forma de um intermediador, o Gateway.

3.2 Permissão de uso dos dados

Um importante aspecto do paradigma IoT, tendo em vista a quantidade massiva de dados decorrente de sua implementação, é o gerenciamento de dados. É esperado que o dono dos dados possua controle total sobre o acesso ao dado. Esta expectativa gera uma demanda maior por esquemas de privacidade e segurança robustos [49].

A maior parte das plataformas analisadas especifica políticas de segurança e privacidade dos dados. Entretanto, o nível de controle é bastante variável, e raramente o usuário final possui controle total. Plataformas como *ThingSpeak* fornecem uma interface web para o usuário final determinar permissões simples de escrita e leitura, enquanto outras, como o *OpenRemote* e o *LinkSmart*, deixam isso a cabo dos desenvolvedores de aplicações.

De modo geral, os dados são enviados à plataforma cloud em formato bruto, armazenados sem criptografia e com poucas preocupações em relação à segurança do dado. Em relação à privacidade dos dados, diversas plataformas permitem ao usuário final designar permissões de leitura e escrita a outras aplicações e usuários.

Para um controle total dos dados e maior segurança, é recomendado que exista suporte para armazenamento local dos dados, como fornecido pela plataforma *OpenRemote*. A arquitetura em *Edge Clouds* ou *Fogs* permite que o usuário tenha controle local sobre os dados. Sendo assim, é possível definir políticas de privacidades mais avançadas, controlando não só quem pode acessar, mas também que recursos podem ser acessados. Além disso, o usuário final pode optar por encriptar os dados localmente, adicionando uma camada de segurança.

3.3 Processamento de dados

O principal valor entregue pelo paradigma IoT são os dados. O que torna a Internet das Coisas tão relevante atualmente é precisamente a capacidade de gerar informação de ambientes e objetos do dia-a-dia. Como os dispositivos IoT necessários para gerar e compartilhar dados são bastante heterogêneos, como já discutido anteriormente, existe dois grandes obstáculos na criação de uma plataforma IoT: organização semântica dos dados, i.e., criar uma modelo de conhecimento uniforme para diferentes tipos de dados; e criação de analíticos de dados eficientes, que extraiam informação útil dos dados respeitando as limitações dos dispositivos IoT e possíveis restrições de tempo real.

Um desafio técnico bastante relevante é o compartilhamento de streams de dados de origens diferentes. O conceito de *data flows*, empregado pela plataforma *Node-RED* surge desta demanda, com a composição de dados de diferentes dispositivos [50], enriquecendo a criação de conteúdo para a *web of things*. Entretanto, poucas plataformas adotam este tipo de tecnologia. Dessa maneira, uma lacuna ainda presente de modo geral é a falta de capacidade das plataformas de lidar com diferentes modelos de dados. Preencher a lacuna irá envolver a criação de um modelo semântico que abstraia as particularidades de cada dispositivo sensor e que também seja robusto para lidar com perda de dados, dados incompletos ou não confiáveis; situações não incomuns em aplicações IoT. Encontrar as streams de dados relevantes é também uma funcionalidade importante, em que a busca pode ser feita através de indexação semântica. Apenas quatro plataformas analisadas (*IoT-Framework*, *Kahvihub*, *ThingWorx* e *Xively*) trazem suporte à busca de stream de dados, e nenhuma delas suporta busca cruzada entre diversas plataformas.

Em relação à análise e processamento de dados, ainda há um suporte limitado à integração de técnicas de processamento de dados às plataformas, e uma das razões para isso é o fato de que é necessário adaptar as técnicas de análise de dados às realidades de IoT, como por exemplo na pesquisa de Tsai et al. em mineração de dados para IoT [37]. A maior lacuna em processamento de dados para IoT é garantir eficiência na análise de dados. Eficiência significando i) processamento de dados considerando as limitações de memória, poder computacional e comunicação de dispositivos IoT; e ii) baixa latência na geração de informação, pois muitas aplicações possuem demandas de tempo real. Além disso, a característica de Big Data das aplicações IoT demandam uma alta vazão de dados.

A utilização de *Fogs* ou *Edge Clouds* no contexto de analíticos de dados em IoT é uma potencial solução para os obstáculos citados. A grande maioria das plataformas emprega apenas um serviço de Cloud, de modo que um analítico de dados é utilizado sobre uma quantidade massiva de dados de forma centralizada. Ao empregar *edge analytics* em Fogs, é possível distribuir o processamento de dados, aumentando a performance e escalabilidade do sistema como um todo. A plataforma *Kahvihub* aplica este conceito para dispositivos limitados, fornecendo execução em *sandbox* de serviços IoT. De modo que uma rede de dispositivos heterogêneos pode analisar de maneira colaborativa os dados produzidos. Este tipo de aplicação é também discutido por Kovatsch et al. [51].

3.4 Suporte ao desenvolvedor

Para criar um ecossistema de desenvolvimento eficiente, é necessário que as plataformas IoT forneçam APIs aos desenvolvedores, de preferência criando uma camada superior de abstração que dispense aos desenvolvedores conhecimento específico da arquitetura e tecnologias subjacentes à plataforma. Maior parte das plataformas de IoT fornecem uma REST API pública, com operações básicas de PUT, PUSH, GET ou DELETE, as únicas exceções sendo as plataformas *LinkSmart*^{TM} e *IFTTT*. Estas operações permitem a interação com dispositivos IoT, além de seu gerenciamento. Porém, como cada plataforma desenvolve sua própria API e modelo de dados, há um grande desafio tecnológico na interoperabilidade entre plataformas a ser superado.

Outra lacuna importante a respeito do suporte ao desenvolvedor é a presença limitada de Software Development Kits (SDKs). De modo geral, as plataformas apenas fornecem *bindings* em diferentes linguagens para as APIs, que implementam funcionalidades básicas da plataforma, e.g. *IoT-Framework* e *Xively*. Idealmente, uma plataforma IoT deve fornecer SDKs que permitam ao desenvolvedor explorar ao máximo os serviços da plataforma, como é o caso da plataforma *Carriots*®.

CAPÍTULO 4

Descrição do KNoT

O KNoT é uma plataforma open source de hardware e software para IoT. Seu principal objetivo é integrar plataformas e protocolos pré-existentes, mediando sua conexão e integração. O KNoT fornece, então, uma solução fim-a-fim, contendo um design de dispositivo IoT, semântica de dados e armazenamento e computação em nuvem.

Neste capítulo, será feita uma descrição em alto nível da meta plataforma KNoT, detalhado sua arquitetura geral e a arquitetura dos componentes básicos. Ao fim do capítulo, a meta plataforma será avaliada em relação aos critérios estabelecidos no capítulo anterior.

4.1 Arquitetura geral do KNoT

A arquitetura do KNoT, ilustrada na Figura 4.1, fornece conectividade a dispositivos com limitações de processamento e comunicação através do Gateway, que age como um intermediário entre a pilha de protocolos mais leve dos dispositivos (Things) e os protocolos Web tradicionais da Cloud.

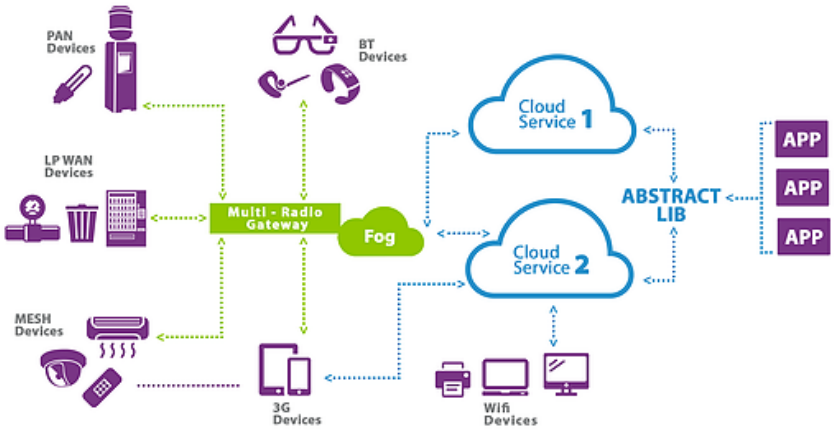


Figura 4.1 Arquitetura geral do KNoT

Nesta arquitetura, os Things são os dispositivos que irão embarcar conectividade a sensores e atuadores que irão enviar e receber dados. Dispositivos limitados irão fazer a comunicação através do Gateway, enquanto dispositivos Wi-Fi e 3G, que já possuem neles implementados protocolos Web padrão, podem se conectar diretamente ao serviço Cloud.

É possível notar que o Gateway está associado a uma Fog, i.e., uma micro instância do

serviço Cloud. Nessa Fog, os dados estão sincronizados com a Cloud e podem ser acessados localmente.

O serviço Cloud irá coletar todo os dados e será o ponto de entrada para as aplicações dos usuários finais. Através de bibliotecas e APIs, desenvolvedores podem ter acesso aos dados de maneira simplificada, com a heterogeneidade de tecnologias e protocolos sendo abstraída pelo KNoT.

Nas próximas seções, serão detalhados os componentes básicos desta arquitetura: as Things, os Gateways e a Cloud.

4.2 KNoT Things

Um KNoT Thing é qualquer dispositivo capaz de coletar e trocar dados via comunicação sem fio com o Gateway utilizando o protocolo KNoT. Um diagrama de seus principais componentes pode ser visto na Figura 4.2. O KNoT Thing é um dispositivo programável, e contém uma API que encapsula o protocolo KNoT subjacente, de modo que o desenvolvedor da aplicação IoT precisa apenas configurar o Thing e implementar os métodos de sensoriamento e atuação.

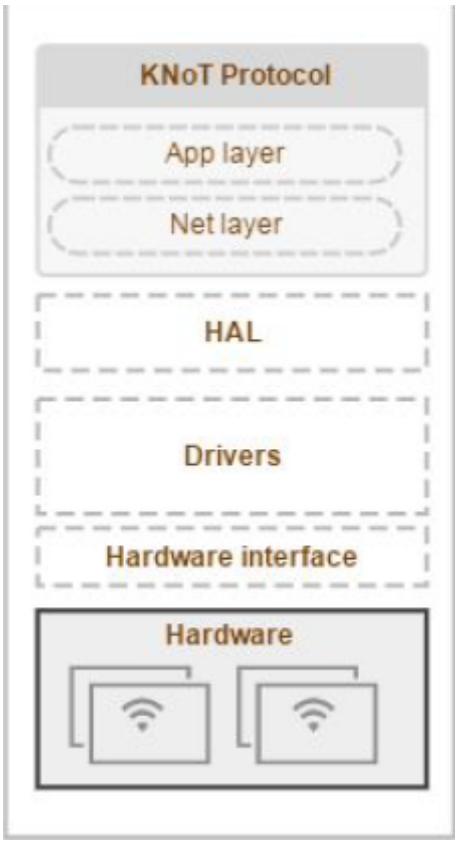


Figura 4.2 Componentes de um KNoT Thing

O hardware de um KNoT Thing possui três componentes: o microcontrolador, o módulo de comunicação e o módulo de alimentação. O módulo de alimentação é trivial, e pode ser

projetado para funcionar com bateria ou diretamente na tomada.

O microcontrolador é a unidade de processamento central do Thing, responsável por gerenciar os sensores, atuadores e faz interface com o módulo de comunicação. Atualmente, a plataforma de hardware suportada pelo KNoT é o Arduino Pro Mini [52]. É um microcontrolador de baixo custo, com portas analógicas e digitais e suporte a interfaces UART, SPI e I2C. Apesar do baixo custo e de ter especificações de processamento e memória modestas, é capaz de fazer interface com diversos sensores e módulos de comunicação sem fio.

O módulo de comunicação do Thing deve ser facilmente substituível, pois a ideia é termos dispositivos heterogêneos em que a tecnologia de comunicação se adequa à necessidade da aplicação. Como se observa na Figura 4.1, os Things podem estar associados a diferentes tipos de redes sem fio.

Um KNoT Thing pode suportar diversas tecnologias sem fio como:

- NRF24L01: É um transceptor RF *ultra low power* com taxa de transmissão de 2Mbps que opera na faixa 2.4 GHz ISM. Sua principal característica é oferecer conectividade sem fio de curto alcance (PAN, ou *Personal Area Network*) a baixos custos e consumo de energia [53].
- LoRa
- Wi-Fi
- Bluetooth Low Energy
- IEEE 802.15.4

O suporte a diversos hardwares é garantido pelo HAL, *Hardware Abstraction Layer* ou Camada de Abstração de Hardware, que fornece uma API padronizada que encapsula as implementações dos drivers de diversas interfaces de hardware, como o SPI e UART e as interfaces com os módulos de comunicação. Em relação aos módulos de comunicação, foi definido um modelo de comunicação padrão baseado em sockets. A camada física e de enlace do módulo é encapsulada neste modelo, com a implementação de fato destas camadas inferiores sendo fornecida pelo próprio módulo. As camadas superiores são definidas pelo KNoT Protocol, respeitando as especificações do HAL. As camadas superiores da aplicação do KNoT Thing, então, podem ser facilmente portáveis, bastando apenas que os drivers do hardware subjacente seja implementado de acordo com as especificações do HAL.

Um KNoT Thing típico, ilustrado na Figura 4.3, possui uma API para o desenvolvedor final bastante simples. Com toda os processos de hardware encapsulados pelo HAL e os processos de comunicação encapsulados pelo KNoT Protocol, o desenvolvedor final necessita apenas embarcar os sensores e atuadores desejados no KNoT Thing e configurá-los via software através de três métodos:

- Registrar sensor ou atuador: Neste método o desenvolvedor registra os metadados do sensor, i.e., o tipo computacional do dado do sensor (booleano, inteiro, float, etc.), o tipo semântico do dado (voltagem, corrente, temperatura, etc.), a unidade de medida do sensor (volts, millivolts, ampére, etc.), nome e ID único do sensor. É neste método também que o desenvolvedor deve registrar as funções de leitura e escrita do sensor.

- Configurar sensor ou atuador: Neste método o desenvolvedor irá definir a política de envio de dados do sensor. Ou seja, define-se que tipo de evento será o gatilho para a transmissão de dados ao Gateway. Esses eventos podem ser: intervalo de tempo, ultrapassagem de um limiar, mudança de valor, etc.
- Iniciar Thing: Após a etapa de registro e configuração de cada sensor ou atuador, este método inicializa a operação do Thing.

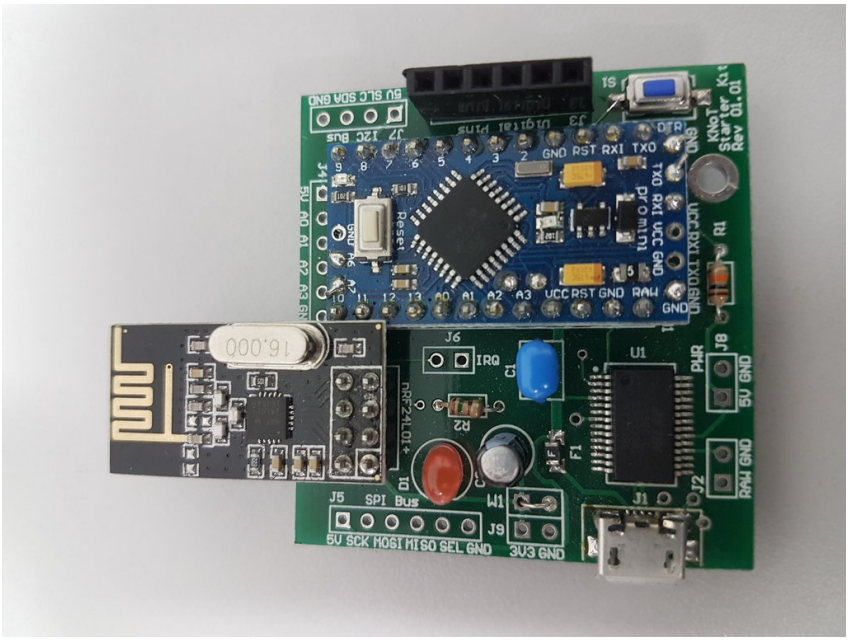


Figura 4.3 Exemplo de um KNoT Thing

Na seção seguinte, será explicado em maiores detalhes o funcionamento do KNoT Protocol.

4.3 KNoT Protocol

O KNoT Protocol é composto de duas camadas: KNoT Net Layer (Camada de rede) e a KNoT Application Layer (Camada de Aplicação). A camada de rede é responsável por gerenciar o endereçamento dos dispositivos conectados à rede e pelo encapsulamento das mensagens de aplicação. A camada de aplicação define os tipos de datagramas utilizados para a troca de mensagens entre os KNoT Things e o KNoT Gateway.

4.3.1 KNoT Net Layer

No protocolo KNoT, cada gateway pode endereçar 240 Things (endereço 1 a 240) e pode se comunicar com outros 13 gateways (endereços 241 a 254). O endereço 255 é o endereço de broadcast. Sendo assim, com o KNoT Protocol é possível ter 3360 dispositivos (gateways e

Things) conectados numa mesma rede. Nas figuras 4.4 e 4.5 estão ilustrados o processo de conexão, respectivamente, de um Thing e um Gateway à rede:

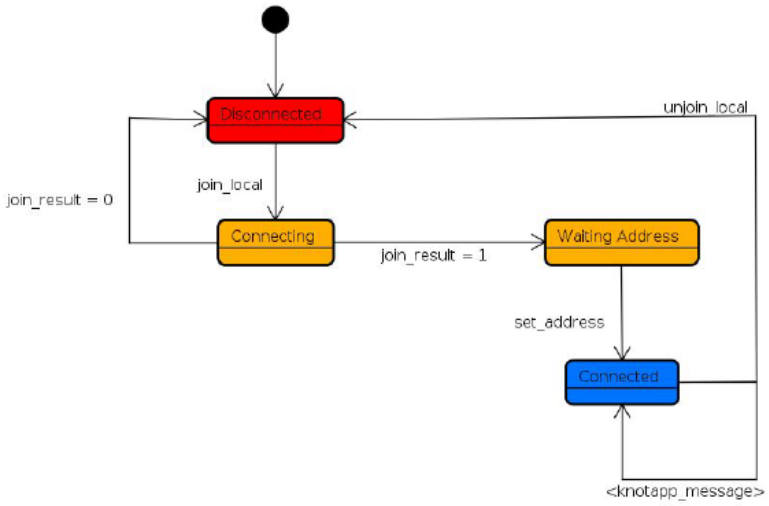


Figura 4.4 Máquina de estados do processo de conexão de um Thing à rede

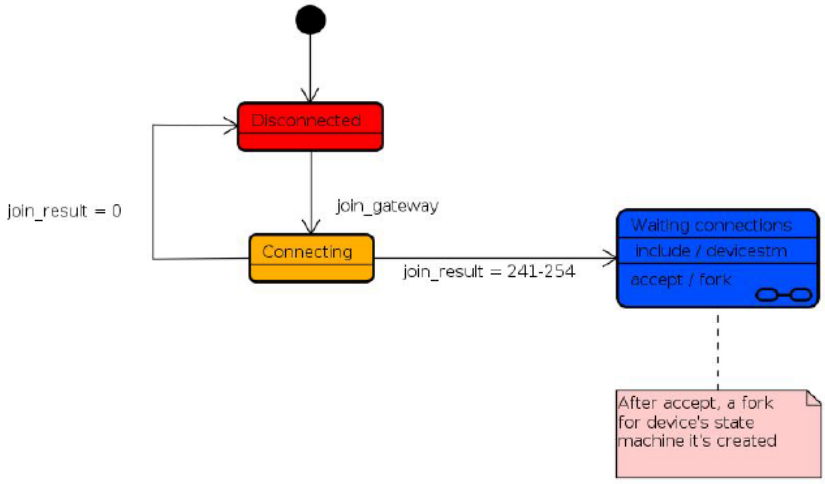


Figura 4.5 Máquina de estados do processo de conexão de um Gateway à rede

A Net Layer do protocolo KNoT possui seis tipos de mensagens, com máximo payload de 128 bytes e um header de 4 bytes. O header contém quatro informações com um byte cada:

- Tipo de mensagem;
- Endereço-fonte;
- Endereço-destino;

- Tamanho do payload.

As mensagens da Net Layer do protocolo KNoT são:

- Join local message: O Thing envia essa mensagem solicitando acesso à rede. Recebe do gateway uma resposta *join result* indicando se a conexão foi bem sucedida ou não.
- Join gateway message: O Gateway faz broadcast desta mensagem para descobrir outros gateways na rede e seus respectivos endereços. O Gateway, então, aloca o menor endereço disponível para si mesmo.
- Join result message: Mensagem enviada em resposta às mensagens de join local ou join gateway. No primeiro caso, a resposta contém apenas um byte com 1 para conexão bem sucedido e 0 para conexão mal sucedida. No segundo caso, a resposta contém o endereço do Gateway que está transmitindo a resposta.
- Unjoin local message: O Thing envia esta mensagem para notificar o gateway da saída da rede.
- Set Address message: O Gateway envia esta mensagem após o sucesso de uma join local message anterior. Nesta mensagem é enviado para o Thing o seu endereço, alocado pelo gateway.
- Application Message: Consiste do header mais a mensagem da camada de aplicação.

4.3.2 KNoT Application Layer

O KNoT Application Layer define as mensagens que são trocadas entre Things e Gateway na configuração de sensores, registro de Things e transmissão de dados. O header da camada de aplicação contém apenas dois bytes, o tipo de mensagem e o tamanho do payload. Os tipos de mensagens são:

- KNoT Register Message: O Thing faz uma solicitação de registro ao Gateway. O Thing só pode enviar dados aos Gateway caso esteja registrado e possuir credenciais (UUID e Token) válidas.
- KNoT Unregister message: O Thing solicita a retirada de cadastro do Gateway ao qual está atualmente atrelado.
- KNoT Credential Message: O Gateway ao receber uma Register Message, gera credenciais válidas e as transmite ao Thing.
- KNoT Authentication Message: O Thing transmite suas credenciais para se autenticar perante ao Gateway, para poder iniciar a transmissão de dados.
- KNoT Result Message: O Gateway informa ao Thing se a autenticação é válida ou não.

- KNoT Schema Message: O Thing envia ao Gateway o schema de um de seus itens (sensor ou atuador). O schema contém as seguintes informações: tipo computacional do dado, tipo semântico do dado, unidade do dado e um identificador único para o item.
- KNoT Configuration Message: O Thing envia ao gateway as informações de configuração do item, i.e., que tipo de evento deve gerar a transmissão do dado e valores pertinentes ao tipo de evento escolhido. E.g., caso o item seja configurado para enviar por período de tempo, deve-se informar qual o período, em segundos.
- KNoT Data Message: A mensagem contém o id do item e um payload. Pode ser enviada do Thing para o Gateway contendo informações dos sensores ou vice-versa contendo comandos aos atuadores.

A Figura 4.6 ilustra o fluxograma de operação do Thing sob o protocolo KNoT. O Thing verifica se possui credenciais, e caso possua ele irá se autenticar com o Gateway. Caso não possua, irá passar pelo processo de registro no Gateway e configuração de schema e itens. Com o Thing devidamente autenticado, ele irá entrar no seu loop de operação, trocando dados com o Gateway de acordo com o especificado nos schemas dos itens.

4.4 KNoT Gateway

As principais funções do KNoT Gateway é agir como um proxy para os KNoT Things e também hospedar uma micro instância do serviço Cloud, a Fog. Dessa maneira, as plataformas de hardware e software que compõem o gateway precisam ser mais poderosas. A modelagem básica dos componentes do KNoT Gateway pode ser vista na Figura 4.7. O principal requisito tecnológico do Gateway é que seja um sistema computacional capaz de rodar uma distribuição Linux chamada KNoT Linux [54], montada especificamente para atender as necessidades da plataforma. Naturalmente, o sistema subjacente ao Gateway deve ter suporte às interfaces de hardware necessárias para garantir a funcionalidade dos módulos de comunicação. Uma vez que o Gateway servirá como proxy para dispositivos heterogêneos, deve haver suporte a uma variedade de tecnologias de comunicação sem fio. Atualmente, a HAL dá suporte a Raspberry Pi 3 [55], um microcomputador amplamente utilizado em soluções IoT.

O KNoT Gateway possui os seguintes componentes:

4.4.1 Service

Este componente é o responsável pela troca de mensagens com os Things, e pela tradução do KNoT Protocol, um protocolo binário, para o protocolo utilizado na Fog e Cloud.

Assim como o KNoT Thing, o Gateway possui os módulos do KNoT Protocol e o HAL. A camada de abstração de hardware no Gateway é responsável por encapsular a comunicação de cada tipo de rádio no KNoT Protocol e enviar esse pacote de dados ao Manager. O Manager, então, traduz o protocolo binário para um modelo semântico padronizado, em JSON ou XML. Os dados são enviados neste modelo a Fog, através de protocolos Web padrão, como MQTT, HTTP, Web Sockets e CoAP.

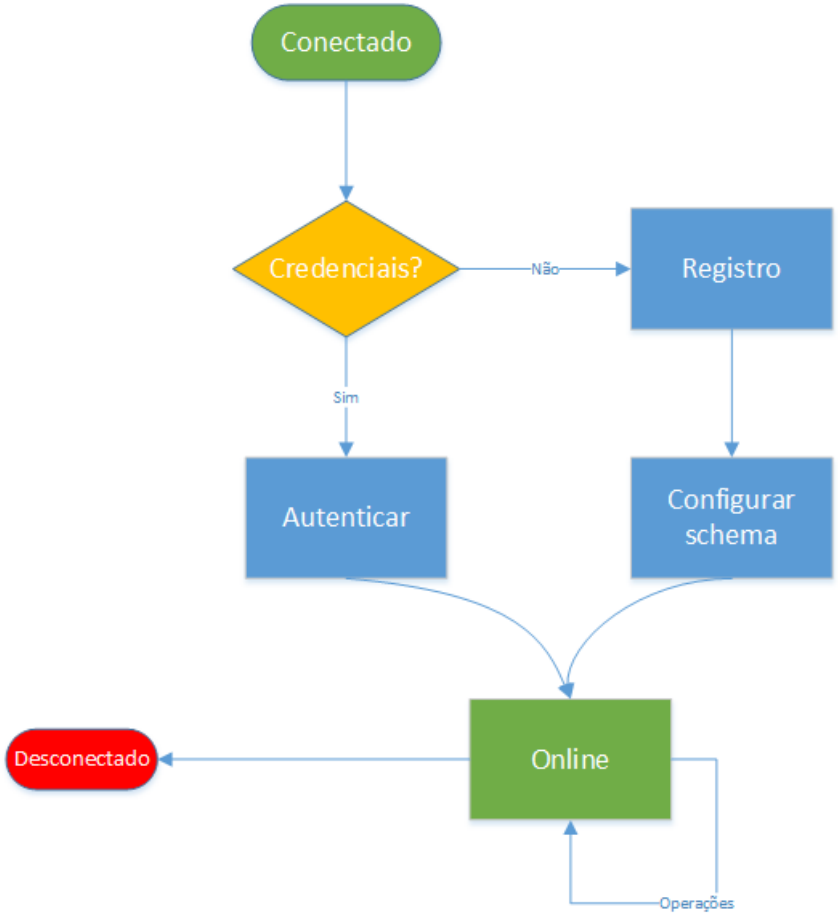


Figura 4.6 Fluxograma do KNoT Protocol em um Thing

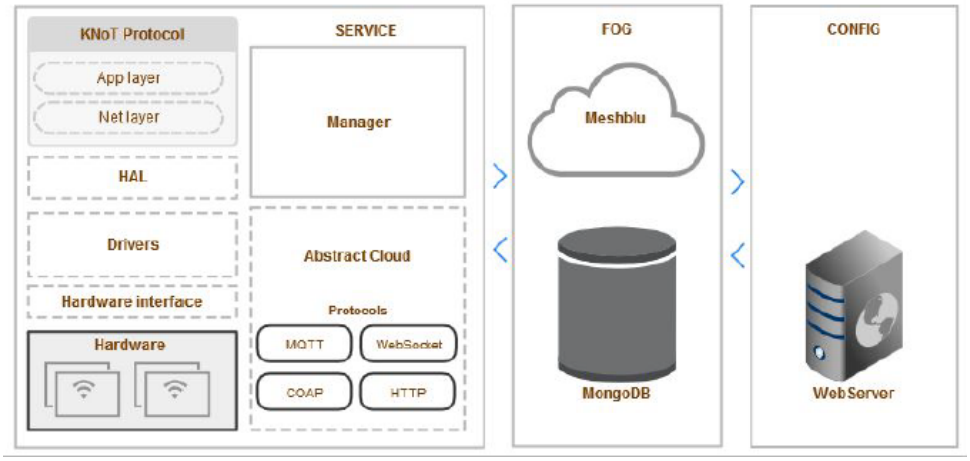


Figura 4.7 Componentes do KNoT Gateway

Em mais detalhes, cada tipo de rádio, NRF24L01, LoRa ou Bluetooth, possui uma especificação diferente das camadas física e de enlace. É papel do HAL criar uma abstração dessas

camadas, de modo que as camadas de rede e de aplicação definidas pelo KNoT Protocol se comuniquem com os rádios de maneira uniforme, em uma interface baseada em sockets. Para cada rádio, existe um processo no SO do Gateway que realiza essas operações. O payload resultante com os dados no formato KNoT Protocol é enviado ao manager. A comunicação entre esses processos se dá via Unix Sockets, uma vez que são processos no mesmo sistema hospedeiro. Entretanto, o Manager também dá suporte a sockets TCP, de modo que também é possível receber dados formatados com o KNoT Protocol de dispositivos que implementam a pilha de protocolos TCP/IP. Por fim, o Manager traduz os dados para um modelo semântico em JSON ou XML e transmite à Fog, como ilustrado pela Figura 4.8.

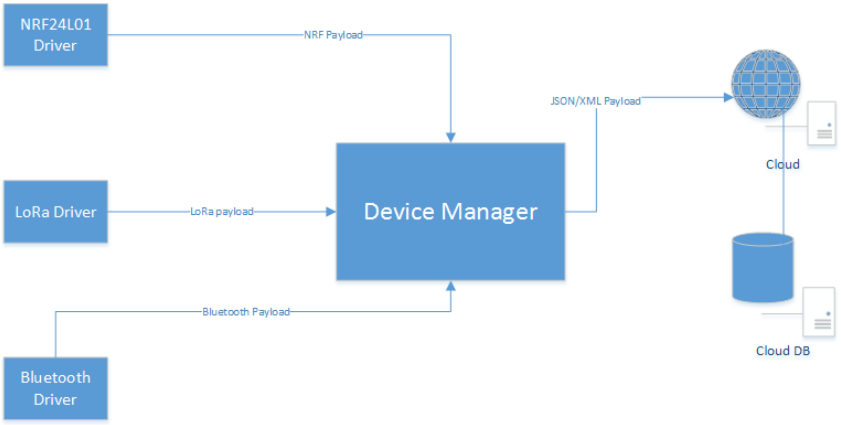


Figura 4.8 Fluxograma dos dados do momento que são captados pelo Gateway até o envio à Fog

4.4.2 Fog

Como no conceito introduzido por Chang et al. [38], a Fog atua como uma micro instância local da Cloud. Os dados de sensores e de configurações dos dispositivos são armazenados no banco de dados da Fog, permitindo acesso local e operação offline. Evidentemente, os dados são sempre sincronizados com a Cloud enquanto houver conexão.

Além de fornecer armazenamento local de dados, a Fog traz o eventual suporte a analíticos de dados locais. Avaliando na escala esperada de IoT, teremos milhares de Gateways gerenciando milhões de devices, gerando dados a um único usuário, que deseja extrair informação útil disso tudo. A Fog, neste caso, proporciona o processamento distribuído de dados, aliviando a carga no serviço de Cloud centralizado, permitindo analíticos de dados cada vez mais robustos e eficientes.

4.4.3 Configuration App

O Gateway também fornece uma aplicação web para configuração e gerenciamento da rede local. Um web server hospedado no próprio Gateway permite gerenciamento dos Things e configuração de protocolos, de parâmetros dos rádios e da rede. A aplicação só pode ser acessada por um usuário KNoT que possui conta associada a um ou mais serviços KNoT Cloud,

como mostrado na Figura 4.9. O Configuration App também permite ao usuário, após realizar o login, definir a qual serviço Cloud a Fog do Gateway está associada, como mostrado na Figura 4.10.

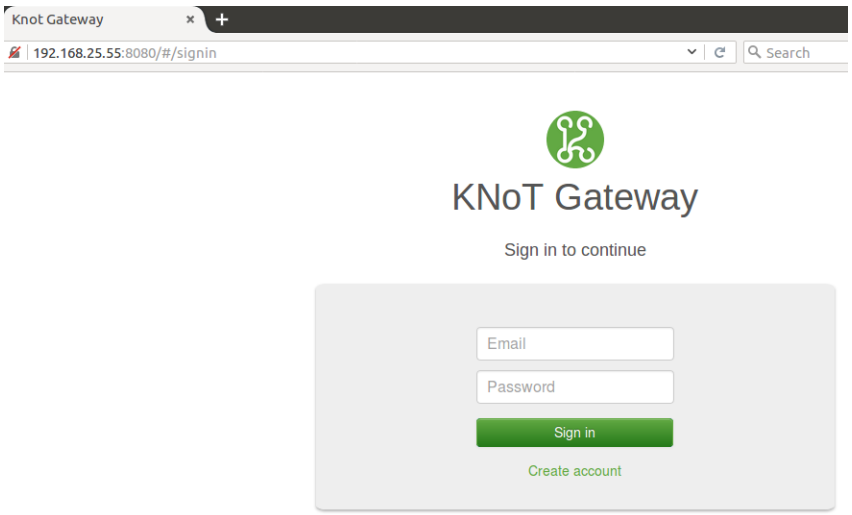


Figura 4.9 Tela de login do Configuration App

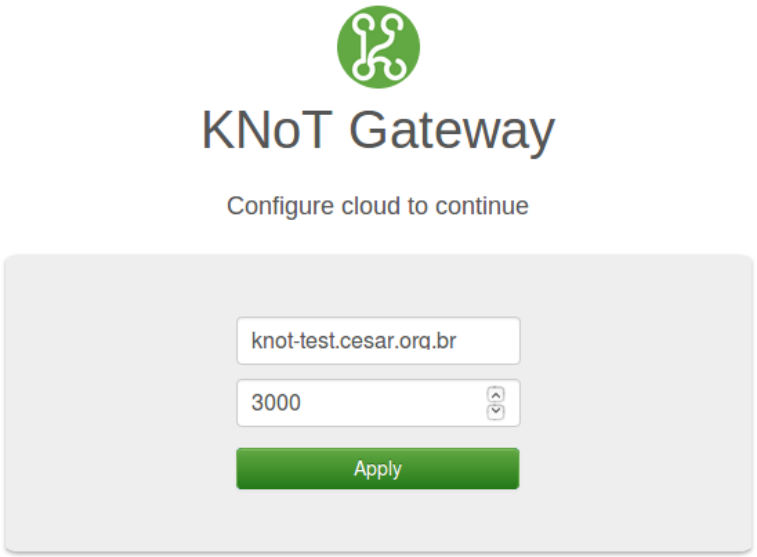


Figura 4.10 Tela de cadastro do Parent Cloud da Fog

A aplicação possui quatro funcionalidades básicas:

1. Administration: Mostra as credenciais do usuário e do Gateway
2. Network Interface: Permite ao usuário configurar a rede local

3. Radio Interface: Permite ao usuário configurar os parâmetros dos rádios. Por exemplo: configurar o canal de transmissão do rádio NRF24L01.

4. My Devices: Permite ao usuário visualizar os Things que estão solicitando acesso à rede e escolher quais Things terão acesso à rede.

A Figura 4.11 mostra a tela de Administration, enquanto a a Figura 4.12 mostra a tela de gerenciamento de Things. Quando o usuário dá permissão a um Thing, o Gateway enfim dá acesso ao Thing à rede local, dando início ao processo ilustrado na Figura 4.6.

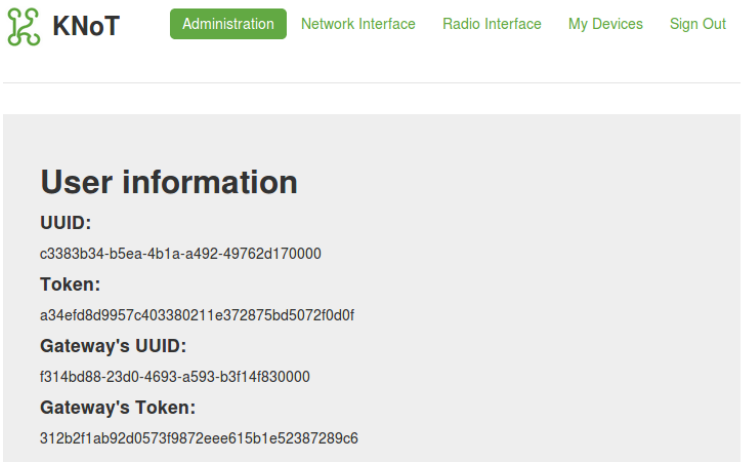


Figura 4.11 Tela inicial do Configuration App

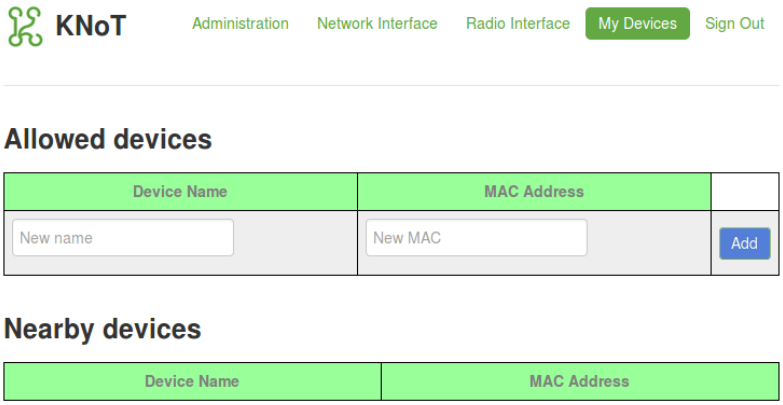


Figura 4.12 Tela de gerenciamento de Things

4.5 KNoT Cloud

O KNoT Cloud é o ambiente que centraliza todos os dados dos Things e os torna acessíveis para as aplicações externas. Utilizando a API padronizada do KNoT Lib, as aplicações podem

descobrir e consultar Things, gerenciar as Fogs, enviar dados aos Things e acessar serviços de analíticos de dados sendo rodados na Cloud e nas Fogs. A Figura 4.13 mostra a estrutura básica do KNoT Cloud e sua comunicação com os Fogs e as aplicações externas.

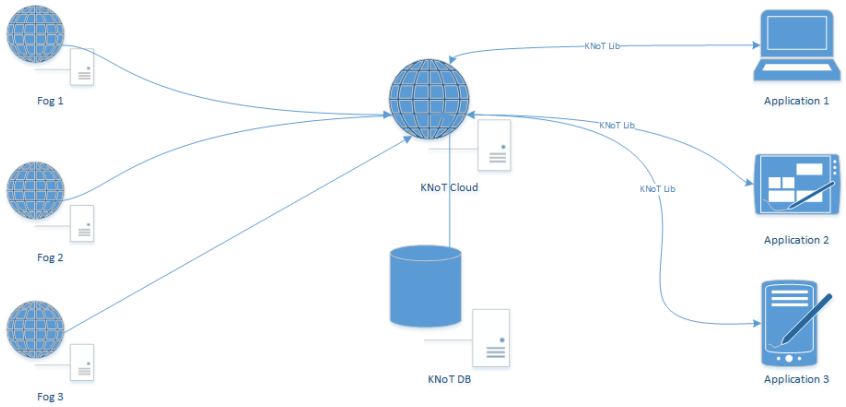


Figura 4.13 Estrutura básica do KNoT Cloud

Atualmente, o KNoT Cloud está hospedado na plataforma Meshblu [10], um sistema de comunicação em nuvem multi-protocolo que fornece uma integração entre diversos dispositivos. Os protocolos suportados pelo Meshblu incluem CoAP, HTTP, AMQP, XMPP, MQTT e Web Sockets. A Figura 4.14 ilustra a característica multi-protocolo da plataforma Meshblu. Com a integração desta plataforma e o KNoT Cloud, é possível utilizar uma API única para entregar os dados dos Things para algum analítico de dados, receber algum resultado e repassá-lo para uma aplicação móvel para visualização do usuário. Do mesmo modo, pode-se receber os dados dos sensores e enviá-los a algum atuador que irá tomar a decisão apropriada.

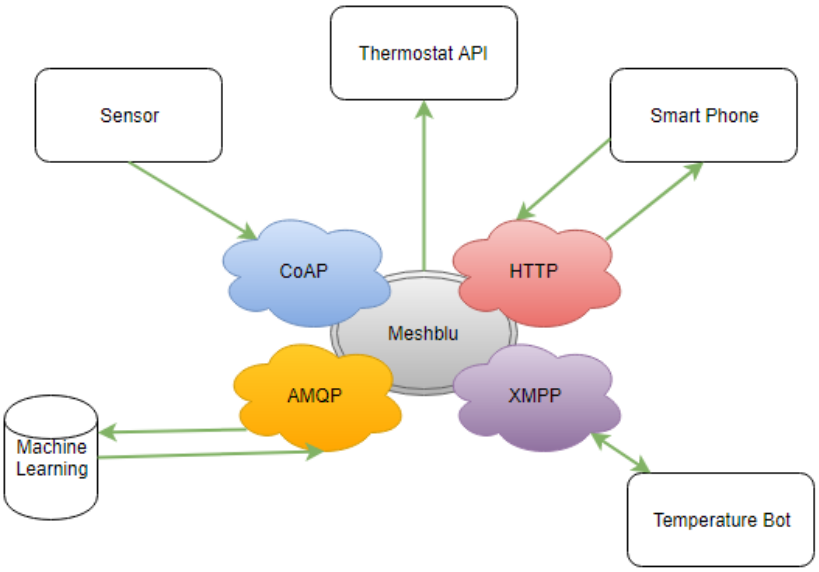


Figura 4.14 Exemplo de funcionamento do Meshblu [10]

Levando em conta a natureza Big Data do IoT, o banco de dados utilizado pelo KNoT Cloud é o MongoDB. O MongoDB, utilizando uma abordagem NoSQL, foi projetado para lidar com a quantidade massiva de dados gerada pelo Big Data, além de possuir suporte nativo a bancos de dados distribuídos e na nuvem, se encaixando no conceito de Fogs e Cloud propagado pelo KNoT. É importante salientar que o KNoT Cloud está em um estágio de desenvolvimento menos avançado que o KNoT Thing e o KNoT Gateway. Dessa maneira, espera-se no futuro que uma plataforma de analítico de dados seja adicionada ao KNoT Cloud, além do suporte a mais de uma plataforma cloud além do Meshblu, para maior flexibilidade do usuário.

CAPÍTULO 5

Análise de gaps do KNoT

Nesta seção, a meta-plataforma KNoT será avaliada em relação aos critérios estabelecidos no Capítulo 3. A Tabela 5.1 sumariza as características principais da plataforma KNoT, em conformidade com a análise de Mineraud et al [1].

5.1 Suporte a dispositivos heterogêneos

Uma das maneiras em que o KNoT suporta dispositivos heterogêneos é na definição de uma camada de abstração de hardware para a comunicação baseada em sockets que ocorre entre dispositivo e Gateway. Dessa maneira, tem-se uma API de comunicação dispositivo-gateway padrão que garante a interoperabilidade de dispositivos, dependendo apenas da criação de drivers apropriados. Embora no presente momento o KNoT suporte de maneira estável apenas o rádio NRF24L01, está em desenvolvimento o suporte a LoRaWAN, Bluetooth e IEEE 802.15.4. Este suporte a diferentes de tecnologias de comunicação sem fio garante o desenvolvimento estável de aplicações mais complexas, que podem combinar diferentes tipos de redes, como LPWAN e PAN.A mesma camada de abstração de hardware encapsula o acesso ao hardware do microcontrolador do KNoT Thing e do hardware do Gateway. Embora as plataformas atuais sejam, respectivamente, Arduino Pro Mini e Raspberry Pi 3, pode-se adicionar novas plataformas com a criação de drivers. Para o desenvolvedor da aplicação, a API mantém-se uniforme.

A arquitetura do KNoT, ilustrada na Figura 4.1, dá grande importância às Fogs e ao Gateway. Uma característica importante do Gateway é de fornecer conectividade à internet a dispositivos que não a possuem naturalmente. Como o Gateway também dá acesso à Fog, todos os dispositivos locais se comunicam com ele. O KNoT Protocol é empregado em toda comunicação dispositivo-gateway. É um protocolo flexível, podendo ser utilizado até mesmo em dispositivos mais limitados, por ter um baixo overhead de memória e processamento. Na seção 4.4 é detalhado que pode-se comunicar com o Gateway via sockets numa rede local de dispositivos limitados, ou através do KNoT Protocol por cima da pilha TCP/IP.

Do Gateway para a Fog e Cloud existe também um suporte para diversos protocolos, que podem ser escolhidos de acordo o desenvolvedor da aplicação. O Meshblu, a plataforma Cloud atualmente suportada pelo KNoT, aceita uma série de protocolos, como mostra a Figura 4.14.

Tabela 5.1 Características da plataforma KNoT, conforme análise de Mineraud et al. [1]

Suporte a dispositivos heterogêneos	Tipo	Arquitetura	Open Source	REST	Controle de acesso de dados	Descoberta de serviços
Sim, com Gateways	PaaS	Cloud + Fog	Sim	Sim	Acesso autenticado	Sim

Atualmente, existe a integração com o KNoT apenas de HTTP e Web Sockets, embora protocolos mais apropriados para IoT como MQTT e CoAP estejam no planejamento para futuros desenvolvimentos.

De modo geral, pode-se observar que embora o suporte do KNoT ainda esteja limitado a algumas plataformas e protocolos, existe uma infra-estrutura que permitem a padronização de todo o caminho do dado, do Thing à Cloud. De modo que uma única aplicação possa servir para qualquer dispositivo, com uma configuração mínima de drivers.

5.2 Permissão de uso dos dados

Como discutido anteriormente, existe um processo de autenticação dos dispositivos perante ao Gateway, baseado na autenticação OAuth2. Esta autenticação é gerenciada pelo KNoT Cloud, e existe da mesma maneira para associar um usuário a um Gateway específico e a um serviço da Cloud. Dessa maneira, apenas usuários autenticados podem ter acesso a determinado dado, fornecendo uma camada de segurança e privacidade de dados ao sistema. Como o KNoT permite o acesso direto ao Gateway de usuários autenticados, existe um controle e armazenamento local de dados.

Entretanto, atualmente não há nenhuma configuração de controle de permissão para outros usuários no KNoT. Apenas um usuário pode se associar a um serviço Cloud e associar Fogs e Gateways a esta Cloud, e este usuário tem controle total dos dados ao longo de todo o processo. No entanto, espera-se no futuro que um superusuário, o dono da aplicação, possa conceder permissões customizadas a outros usuários.

Em relação a criptografia na transmissão dos dados, ainda não há nenhum suporte no KNoT para esta funcionalidade. Este é um desafio tecnológico não-trivial para os desenvolvedores de protocolos para IoT, pois adicionar uma camada de encriptação geralmente adiciona overheads indesejados que podem inviabilizar algumas aplicações.

5.3 Processamento de dados

Como explicitado na seção 3.3, há dois principais desafios em relação ao processamento de dados em plataformas IoT: 1) a criação de modelos semânticos uniformes e 2) analíticos de dados eficientes para o ambiente IoT. Um outro aspecto importante, derivado destes dois desafios, é a capacidade de busca e composição de fontes de dados diferentes.

Em relação a um modelo semântico uniforme, a principal proposta do KNoT é justamente a interoperabilidade. Além do KNoT Protocol, que padroniza a comunicação entre dispositivos e gateways, o KNoT fornece um modelo semântico padronizado em JSON. Todos os dados dos Things enviados ao Gateway passam por um processo de tradução, do protocolo binário para o modelo semântico, antes de serem enviados à fog ou à nuvem.

Entretanto, ainda não há suporte no KNoT Cloud para serviços de analíticos de dados. O uso de Fogs em sua arquitetura básica é um passo na direção das recomendações citadas na seção 3.3. Mas para fazer uso total do potencial das Fogs em IoT, é necessário o emprego de analíticos de dados distribuídos atrelados a uma plataforma robusta de analíticos de dados na

nuvem.

Outra lacuna encontrada no KNoT é a falta de suporte à busca e composição de streams de dados. O uso de um modelo semântico padrão, no entanto, oferece um bom ponto de partida para o preenchimento desta lacuna.

5.4 Suporte ao desenvolvedor

A meta-plataforma KNoT é inteiramente open source [56], permitindo aos desenvolvedores o acesso à toda infra-estrutura do KNoT que permite a interoperabilidade de dispositivos e plataformas, e.g. o HAL, as especificações de hardware e software do Gateway, a especificação do KNoT Protocol, a API do KNoT Thing, a API do KNoT Cloud e etc. O KNoT ganha muito em flexibilidade dessa maneira, pois a existência de uma comunidade ativa de colaboradores e usuários da plataforma irá mantê-la em constante evolução e atualização. Como novas tecnologias surgem diariamente, esta flexibilidade é importante para manter a proposta do KNoT relevante, i.e., garantir a interoperabilidade entre tecnologias IoT. Os esforços da equipe do KNoT e de colaboradores poderão garantir que:

- Novas tecnologias de comunicação sem fio sejam utilizadas no KNoT, através da implementação de bindings e drivers que permitam a utilização do KNoT Protocol.
- Novos microcontroladores e microcomputadores sejam utilizados para os Things e Gateways, respectivamente, com a implementação dos drivers apropriados.
- O Gateway suporte novos protocolos de comunicação IoT em sua interface com a Cloud.
- Sejam integradas novas plataformas Cloud ao KNoT Cloud, para entregar maior flexibilidade ao usuário.
- Novas funcionalidades de modo geral sejam adicionadas para acrescentar robustez à plataforma, como por exemplo as lacunas citadas durante ao longo deste capítulo.

O KNoT também fornece o KNoT Lib, uma abstração sobre o KNoT Cloud que permite o desenvolvimento de aplicações IoT, e.g. monitoramento de perímetro com sensores de barreira, monitoramento de temperatura em ambientes de interesse, controle remoto de atuadores, etc. Atualmente, há o suporte do KNoT Lib apenas para Android, mas espera-se no futuro suporte para iOS e aplicações Web.

5.5 Considerações finais

A tabela 5.2 sumariza a análise do KNoT conforme critérios propostos por Mineraud et. al [1], avaliando o status atual da plataforma, as lacunas encontradas, os desafios que produzem as lacunas e recomendações. Em relação ao suporte a dispositivos heterogêneos, fica evidente que o maior desafio a ser superado é a limitação computacional dos dispositivos, o que obriga o KNoT a adotar estratégias menos otimizadas para garantir o suporte a dispositivos heterogêneos, como

a presença de um Gateway na rede e dispensar protocolos de segurança. Recomendações para o KNoT incluem a elaboração de uma arquitetura sem Gateway, focando em dar suporte aos Things a protocolos de comunicação IoT como o MQTT e CoAP. A plataforma *Xively*, por exemplo, fornece uma biblioteca em C para habilitar comunicação MQTT em dispositivos embarcados. Uma abordagem semelhante é recomendada, dando múltiplas opções de protocolos de comunicação, tanto para os dispositivos quanto para a plataforma na nuvem. Embora ainda seja pouco implementado na indústria, é importante também a presença de protocolos de segurança na comunicação entre dispositivos IoT numa rede. O trabalho de Asokan et al. [57], por exemplo, fornece segurança para enxames de dispositivos IoT levando em conta as limitações tecnológicas (e.g. memória, processamento, latência, energia). Outro trabalho faz uma análise de diversos protocolos propostos para segurança em cenários IoT centralizados e distribuídos [58]. A recomendação à equipe do KNoT, portanto, é tomar ciência de trabalhos como esses e do estado da arte em segurança para IoT para construir uma plataforma IoT segura.

A principal lacuna do KNoT quanto à sua política de acesso de dados é a falta de perfis de usuários com permissões customizadas. É uma política de dados ainda primitiva, em que apenas o usuário com as credenciais válidas possui acesso aos dados. Espera-se que o KNoT, no futuro, permita que o proprietário dos dados defina perfis de usuários. Estes perfis devem conter não só permissões de leitura e/ou escrita, mas também restrições de acesso a determinados recursos da plataforma. Por exemplo, o proprietário pode definir um tipo de usuário chamado "operador", que deve acionar atuadores seguindo um determinado procedimento. Portanto, a este usuário será concedida permissões de escrita para apenas alguns dos dispositivos, no caso atuadores de uma região específica. A criação de perfis mais sofisticados fornece maior robustez à plataforma, e dá mais controle e liberdade ao proprietário dos dados na construção de aplicações.

Quanto ao processamento e compartilhamento de dados, o KNoT ainda é bastante primitivo. Embora adote um formato padronizado de dados, a interoperabilidade que tal formato permite não é explorada. Dessa maneira, espera-se que o KNoT desenvolva um mecanismo de busca de dados via indexação semântica. Com um modelo semântico padrão, será possível utilizar dados públicos de diversas fontes em uma mesma aplicação de maneira simples. Isso entra na questão de privacidade de dados, pois deve ser prerrogativa do proprietário dos dados definir quais dados podem ser indexados publicamente. Outra lacuna é a falta de processamento de dados. Visualizar e extrair informação dos dados brutos de milhões de sensores é uma tarefa impossível para o usuário final, de modo que uma ferramenta de analíticos de dados torna-se necessária. Para suprir essa lacuna, o KNoT deve fornecer suporte a técnicas de mineração de dados para IoT [37], além de mecanismos de visualização de dados capazes de entregar ao usuário final informação útil e legível. Espera-se também que o KNoT integre técnicas de *edge analytics*, aproveitando-se de sua arquitetura em Fog. O uso de *edge analytics* na indústria ainda é escasso, mas há extensos estudos acadêmicos apontando sua importância para a IoT [59] [60].

A principal expectativa para o KNoT em termos de suporte ao desenvolvedor é a expansão do SDK, estendendo suporte a iOS e Web. Naturalmente, o SDK deve ser expandido de acordo com a implementação de novas funcionalidades. À parte disto, o KNoT mantém a plataforma inteiramente open-source, de modo que a criação de novas funcionalidades é mais flexível.

Tabela 5.2 Análise do KNoT

Categoria	Status	Lacunas	Problemas	Recomendações
Suporte a dispositivos heterogêneos	Existe suporte mas depende de gateway open-source	Depende de gateway	Dispositivos com limitação computacional	Arquitetura sem dependência de gateway
		Não há integração de protocolos de segurança		Integração de protocolos de segurança
Acesso de dados	Armazenamento local com configurações limitadas de acesso aos dados	Não há perfis de usuários	Ainda em desenvolvimento	Proprietário dos dados deve conceder permissões customizadas a diferentes usuários
Processamento e compartilhamento de dados	Formato uniforme de dados, independente de plataforma Cloud	Não há integração de analíticos de dados	Ainda em desenvolvimento	Inserir processamento de dados em um contexto de edge analytics
	Pouca integração de analíticos de dados, mas há infra-estrutura para Edge Analytics	Não há suporte a busca data streams		Permitir busca por catálogo de dados
Suporte ao desenvolvedor	Plataforma totalmente open-source	n.a.	n.a.	Expandir SDK
	Fornece SDK para Android			

Espera-se apenas uma documentação mais robusta de toda a arquitetura e componentes.

CAPÍTULO 6

Conclusão

A escala massiva da Internet das Coisas e a diversidade de tecnologias subjacentes ao paradigma fazem com que cada vez mais seja necessária a presença de um agente mediador. As plataformas IoT surgem para suprir essa demanda, oferecendo recursos para conectividade e gerenciamento de dispositivos IoT e para análise, visualização e armazenamento e dados. No cenário atual das plataformas IoT, diversas lacunas oriundas de desafios tecnológicos impedem o desenvolvimento pleno de aplicações IoT com alta capacidade de integração de dispositivos e com analíticos de dados robustos. Neste cenário, a meta-plataforma KNoT foi proposta com o intuito de integrar plataformas de hardware e software.

Ao longo da análise do KNoT com respeito às lacunas observadas no cenário IoT, foi possível fazer as seguintes observações:

- O KNoT possui uma infra-estrutura robusta para integração de dispositivos heterogêneos, com suporte atual Arduino e Raspberry Pi, duas plataformas amplamente utilizadas. Com o HAL (Hardware Abstraction Layer, ou Camada de Abstração de Hardware), torna-se possível estender o suporte para outras plataformas implementando drivers. A aplicação final poderia, então, ser utilizada em diferentes tipos de KNoT Things.

- O KNoT propôs um protocolo de baixo overhead de memória para integrar diferentes tipos de tecnologia de comunicação sem fio. Com uma interface de comunicação padronizada baseada em sockets, o KNoT Protocol fornece as camadas de rede e de aplicação. Para as camadas inferiores, há o suporte atual para NRF24L01 [61], e em desenvolvimento para Bluetooth, LoRaWAN e IEEE 802.15.4. O suporte é extensível.

- O KNoT fornece autenticação para dispositivos e usuários, mas não há nenhum mecanismo atualmente para concessão de permissões customizadas para outros usuários.

- A arquitetura com Fogs do KNoT a põe à frente da maioria das plataformas atuais, que normalmente fornecem apenas um serviço Cloud. Entretanto, a falta de suporte atual a analíticos de dados impede que se faça uso pleno dessa arquitetura. Ainda assim, o armazenamento local de dados é uma funcionalidade importante para dar mais controle do dado ao usuário final.

- A natureza open source do KNoT permite uma extensibilidade mais elevada, fazendo com que novas tecnologias possam ser integradas à plataforma mais rapidamente.

- O fornecimento de uma API pública com modelo semântico padronizado para desenvolvedores da aplicação fim garante uma maior interoperabilidade de dados de diferentes fontes.

Como fica evidente, o KNoT ainda é uma plataforma em desenvolvimento. Portanto, trabalhos futuros devem focar na implementação de novas funcionalidades para o preenchimento das lacunas observadas, e na integração de novas tecnologias relevantes no cenário IoT.

Referências Bibliográficas

[1] J. Mineraud, O. Mazhelis, X. Su, and S. Tarkoma. A gap analysis of internet of things platforms. *Computer Communications*, 89-90:5–17, 2016.

[2] IHS Markit. Iot platforms - enabling the internet of things. `https://www.ihs.com/Info/0416/internet-of-things.html`. Acessado em: 2017-04-03.

[3] J. Manyika et al. Disruptive technologies: Advances that will transform life, business and the global economy. Technical report, McKinsey Global Institution, 2013.

[4] P. Pongpaibool. A study on performance of uhf rfid tags in a package for animal traceability application. In *ECTI-CON*. ECTI, 2008.

[5] Estimote. `https://estimote.com`.

[6] J. A. Gutierrez, M. Naeve, E. Callaway, M. Bourgeois, V. Mitter, and B. Heile. Ieee 802.15.4: a developing standard for low-power low-cost wireless personal area networks. *IEEE Network*, 15(5):12–19, 2001.

[7] A technical overview of lora and lorawan. Technical report, LoRa Alliance, 2015.

[8] Xiang Su, Jukka Riekki, Jukka K. Nurminen, Johanna Nieminen, and Markus Koskimies. Adding semantics to internet of things. *Concurrency and Computation: Practice and Experience*, 27(8):1844–1860, 2015.

[9] A. Al-Fuqaha, M. Guizane, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols and applications. *IEEE Communications Surveys and Tutorials*, 17(4):2347–2376, 2015.

[10] Meshblu. Welcome to meshblu. `https://meshblu.readme.io/`.

[11] K. J. Singh and D. S. Kapoor. Create your own internet of things: A survey of iot platforms. *IEEE Consumer Electronics Magazine*, 6(2):57–68, 2017.

[12] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787–2805, 2010.

[13] Aeris. Internet of things faq. `http://www.internetofthingsfaq.com/`. Acessado em: 2017-05-17.

[14] L. Srivastava. Pervasive, ambient, ubiquitous: the magic of radio. In *From RFID to the Internet of Things*. European Comission Conference, Belgium, 2006.

[15] Disruptive Civil Technologies. Six technologies with potential impacts on us interests out to 2025. Technical report, National Intelligence Council, 2008.

[16] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang. A first look at cellular machine-to-machine traffic: large scale measurement and characterization. *Proc. ACM Sigmetrics Perform. Eval. Rev.*, pages 65–76, 2012.

[17] C.E.S.A.R. Knot network of things. `http://knot.cesar.org.br/`. Acessado em: 2017-04-03.

[18] D. Donsez K. Gama, L. Touseau. Combining heterogeneus service technologies for building an internet of things middleware. *Computer Communications*, 35(4):405–417, 2012.

[19] W. Wang, S. de, R. Toenjes, E. Reetz, and K. Moessner. A compreehensive ontology for knowledge representation in the internet of things. *Proc. TrustCom*, pages 1793–1798, 2012.

[20] M Ayoub Khan, Manoj Sharma, and Ha Prabhu R. A survey of rfid tags.

[21] RFID Journal. Boeing tags shipment to the dod. `http://rfidjournal.com/article/articleview/1587/1/1`.

[22] Apple pay. `https://www.apple.com/apple-pay/`.

[23] Tech Crunch. 6.1b smartphone users globally by 2020 overtaking basic fixed phone subscriptions. `https://techcrunch.com/2015/06/02/6-1b-smartphone-users-globally-by-2020-overtaking-basic-fixed-phone-s`

[24] Bluetooth Special Interest Group. Specification of the bluetooth system, covered core package, version: 4.0, 2010.

[25] Bluetooth Special Interest Group. Bluetooth sig annual report. `https://www.bluetooth.org/en-us/Documents/Annual_Report_2014.pdf`, 2014.

[26] ibeacon. `https://developer.apple.com/ibeacon/`.

[27] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393 – 422, 2002.

[28] Internet usage statistics. http://www.internetworldstats.com/stats.htm, 2017.

[29] Espressif. Esp32 overview. `https://espressif.com/en/products/hardware/esp32/overview`.

[30] L. Li, H. Xiaoguang, C. Ke, and H. Ketai. The applications of wifi-based wireless sensor network in internet of things and smart grid. In *2011 6th IEEE Conference on Industrial Electronics and Applications*, pages 789–793, 2011.

[31] J. P. Bardyn, T. Melly, O. Seller, and N. Sornin. Iot: The era of lpwan is starting now. In *ESSCIRC Conference 2016: 42nd European Solid-State Circuits Conference*, pages 25–30, Sept 2016.

[32] N. Abramson. The alohanet - surfing for wireless data [history of communications]. *IEEE Communications Magazine*, 47(12):21–25, Dec 2009.

[33] T Berners-Lee, J Hendler, and Lassila O. The semantic web. *Scientific American*, 2001.

[34] Big data computing and clouds: Trends and future directions. *Journal of Parallel and Distributed Computing*, 79–80:3 – 15, 2015.

[35] G. Bell, T. Hey, and A. Szalay. Computer science: Beyond the data deluge. *Science*, 323(5919):1297–1298, 2009.

[36] B. B. P. Rao, P. Saluia, N. Sharma, A. Mittal, and S. V. Sharma. Cloud computing for internet of things amp; sensing based applications. In *2012 Sixth International Conference on Sensing Technology (ICST)*, pages 374–380, 2012.

[37] C. Tsai, C. Lai, M. Chiang, and L. T. Yang. Data mining for internet of things: a survey. *IEEE Communications Surveys & Tutorials*, 16(1):77–97, 2014.

[38] H. Chang, A. Hari, S. Mukherjee, and T. V. Lakshman. Bringing the cloud to the edge. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 346–351, 2014.

[39] Oleksiy Mazhelis and Pasi Tyrvainen. A framework for evaluating internet of things platforms: Application provider viewpoint. In *IEEE World Forum on Internet of Things*, 2014.

[40] James F. Moore. *The Death of Competition - Leadership & Strategy in the Age of Business Ecosystems*. Harper Paperbacks, 1997.

[41] D. et. al Guinard. Towards physical mashups in the web of things. In *International Conference on Networked Sensing Systems*. IEEE Computer Society, 2009.

[42] M. W. Maier and Rechtin E. *The Art of Systems Architecting*. CRC Press, 2nd edition, 2000.

[43] S. et al. Bandyopadhyay. Role of middleware for internet of things: a study. *International Journal of Computer Science & Engineering Survey*, 2(3):94–105, 2011.

[44] T. et al. Teixeira. Service-oriented middleware for the internet of things: a perspective. In *Proceedings of the 4th European Conference on Towards a Service-Based Internet*, pages 220–229, 2011.

[45] E.C.G.F Silva, M.I.S. Oliveira, E.A. Oliveira, K. Gama, and B.F. Loscio. Um survey sobre plataformas de mediação de dados para internet das coisas. In *42º SEMISH - Seminário Integrado de Software e Hardware*. XXXV Congresso da Sociedade Brasileira de Computação(CSBC), 2015.

[46] J. Pasley. How bpel and soa are changing web services development. *IEEE Internet Computing*, 9(3):60–67, 2005.

[47] Z. Yang et al. Study and application on the architecture and key technologies for iot. *Proc. ICMT*, pages 747–751, 2011.

[48] Tech Target. Using an iot gateway to connect the things to the cloud. `http://internetofthingsagenda.techtarget.com/feature/Using-an-IoT-gateway-to-connect-the-Things-to-the-cloud`.

[49] R. Roman, P. Najera, and J. Lopez. Securing the internet of things. *Computer*, 44(9):51–58, Sept 2011.

[50] Michael Blackstock and Rodger Lea. Toward a distributed data flow platform for the web of things (distributed node-red). In *Proceedings of the 5th International Workshop on Web of Things*, WoT '14, pages 34–39, New York, NY, USA, 2014. ACM.

[51] M. Kovatsch, M. Lanter, and S. Duquennoy. Actinium: A restful runtime container for scriptable internet of things applications. In *2012 3rd IEEE International Conference on the Internet of Things*, pages 135–142, Oct 2012.

[52] Arduino pro mini. `https://www.arduino.cc/en/Main/ArduinoBoardProMini`.

[53] Nordic Semiconductor. nrf24l01+. `http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01P`.

[54] CESAR. Knot gateway buildroot. `https://github.com/CESARBR/knot-gateway-buildroot`.

[55] Raspberry. Raspberry pi 3 model b. `https://www.raspberrypi.org/products/raspberry-pi-3-model-b/`.

[56] CESAR. `https://github.com/CESARBR`.

[57] N. Asokan, Ferdinand Brasser, Ahmad Ibrahim, Ahmad-Reza Sadeghi, Matthias Schunter, Gene Tsudik, and Christian Wachsmann. Seda: Scalable embedded device attestation. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 964–975, New York, NY, USA, 2015. ACM.

[58] Rodrigo Roman, Jianying Zhou, and Javier Lopez. On the features and challenges of security and privacy in distributed internet of things. *Computer Networks*, 57(10):2266 – 2279, 2013. Towards a Science of Cyber Security Security and Identity Architecture for the Future Internet.

[59] A. V. Dastjerdi and R. Buyya. Fog computing: Helping the internet of things realize its potential. *Computer*, 49(8):112–116, Aug 2016.

[60] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos. Edge analytics in the internet of things. *IEEE Pervasive Computing*, 14(2):24–31, Apr 2015.

[61] H. Araujo. Implementação e análise do nrf24l01+ como beacon bluetooth low energy.

Este volume foi tipografado em LaTeX na classe UFPEThesis (www.cin.ufpe.br/~paguso/ufpethesis).