UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
CURSO DE BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

Esdras Barbosa Lima da Silva Júnior

Node-RED KNoT: Um módulo de integração da ferramenta Node-RED com a meta plataforma KNoT

RECIFE
2020

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
CURSO DE BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

Esdras Barbosa Lima da Silva Júnior

Node-RED KNoT: Um módulo de integração da ferramenta Node-RED com a meta plataforma KNoT

Monografia apresentada ao Centro de Informática (CIN) da Universidade Federal de Pernambuco (UFPE), como um requisito parcial para conclusão do Curso de Engenharia da Computação, orientada pelo professor Kiev Santos da Gama.

RECIFE
2020

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
CURSO DE BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

Esdras Barbosa Lima da Silva Júnior

Node-RED KNoT: Um módulo de integração da ferramenta Node-RED com a meta plataforma KNoT

Monografia submetida ao corpo docente do Centro de Informática (CIN) da Universidade Federal de Pernambuco (UFPE).

Recife, _____ de _____ de 2020.

BANCA EXAMINADORA

_____

Prof. Dr. Kiev Santos da Gama
(Orientador)
Universidade Federal de Pernambuco

_____

Prof. Dr. Adriano Augusto de Moraes Sarmento
(Examinador)
Universidade Federal de Pernambuco

RECIFE
2020

AGRADECIMENTOS

Agradeço primeiramente a Deus pela Vida, raíz de toda força, que me ajuda a superar os desafios dessa jornada, me fazendo crescer cada vez mais; de toda sabedoria, que me auxilia a tomar as decisões mais assertivas e corretas; de todo amor, que me permite olhar os meus amigos e familiares com empatia e humildade; e de toda a felicidade da qual pude desfrutar durante esse meu caminhar.

Agradeço também a meus amados pais, que para além das oportunidades, experiências e aprendizados que já me proporcionaram, me nutriram daquilo que é mais importante: muito amor, compreensão, incentivo e liberdade. Meu muito obrigado a essas duas pessoinhas, que são verdadeiros gigantes, dos quais me orgulho e admiro tanto.

Agradeço profundamente também a meu irmão, minha namorada, minha sobrinha e minha cunhada, que me apoiaram incondicionalmente, com muito amor e compreensão durante minha trajetória. Espero também poder ajudar cada um a realizar seus objetivos e sonhos!

Aos meus queridos amigos, muito obrigado pelos momentos alegres, noites viradas fazendo projetos e/ou estudando, confraternizações pós provas, almoços e lanches no RU/área 2, enfim, todas essas inestimáveis lembraças que vivemos. Obrigado pelo companheirismo e lealdade, que tornaram essa jornada tão leve e divertida. Cada um tem e sempre terá um lugar especial em meu coração.

Agradeço também a UFPE, em especial ao meu querido CIn. Não apenas a instituição física, na qual tive a oportunidade de estar presente durante esse momento tão especial de minha vida, mas principalmente às pessoas que constituem esse centro de excelência. Obrigado aos servidores, que trabalham com zelo por esse centro que tanto prezamos, pelas recepções alegres, conversas descontraídas e auxílio nas questões mais burocráticas.

De um modo especial agradeço aos meus estimados professores, que mais do que conhecimentos, transmitiram suas experiências com muito amor e empenho a cada um de nós alunos. Muito obrigado por tudo! Mais do que profissionais, vocês formam verdadeiros seres humanos, capazes de trabalhar para o bem uns dos outros.

Finalizo agradecendo a todos aqueles e aquelas que participaram dessa minha caminhada de alguma forma, em seu respectivo tempo e local, os quais eventualmente não citei previamente aqui. Muito obrigado a todos!

“Jovens, sonhem! Cresçam e subam bem alto!”

“Você é um ser grandioso dotado de possibilidades infinitas.”

“Avance corajosamente em direção ao seu ideal.”

“Assim, alcançará infalivelmente a vitória.”

- Masaharu Taniguchi (O Livro dos Jovens)

RESUMO

A internet das coisas vem crescendo significativamente nos últimos anos, e o número de aplicações envolvendo dispositivos inteligentes, conectados, e que interagem com meio físico através de sensores e atuadores é cada vez maior. Para atender aos diversos requisitos necessários em um projeto de IoT, muitos protocolos, ferramentas e tecnologias vêm surgindo ao longo desses anos. O KNoT é uma meta plataforma de código aberto para IoT que tem o propósito de simplificar a criação de soluções em IoT, reduzindo o esforço de desenvolvimento e abstraindo boa parte da conectividade necessária em um projeto do tipo, agindo como uma espécie de tradutor de protocolos, onde o desenvolvedor pode configurar quais tecnologias e protocolos ele deseja usar em seu projeto. O Node-RED por sua vez é uma outra ferramenta de código aberto utilizada na construção de aplicações orientadas a eventos. Ela é programada por meio de blocos visuais, chamados nodes, que possibilitam a criação de fluxos de execução sem o uso de linguagens de programação, tornando-a acessível a um vasto número de usuários, tanto por sua didática simples quanto pela facilidade na construção de fluxos. Dito isso, este trabalho tem o objetivo de criar uma biblioteca de nodes para o Node-RED com o propósito de facilitar o desenvolvimento e prototipação de projetos com a plataforma KNoT, bem como a configuração da mesma.

Palavras-chave: Internet das coisas, sistemas embarcados, computação em nuvem, Node-RED, KNoT.

ABSTRACT

The internet of things has been growing significantly in the last few years and with it the number of applications involving connected smart devices, which make use of sensors and actuators to interact with the real world. To meet all IoT solution requirements, many tools, protocols, and technologies have been surging as well. The KNoT project, for example, is an open-source meta platform for IoT that aims to simplify the IoT solutions development, by reducing the complexity and abstracting part of the connectivity in such projects. It acts like a protocol translator, in which a developer can set up the technologies and protocols that he wants in his project. The Node-RED, in turn, is another open-source tool used to develop event-driven applications. It makes use of visual blocks, called nodes, which allow execution flows to be created without the use of programming languages. It makes Node-RED very accessible to a big number of users, both for its simplicity and ease of flow creation. That said, this work aims to create a Node-RED library that makes the prototyping, development, and setting of KNoT projects easier.

Keywords: Internet of things, embedded systems, cloud computing, Node-RED, KNoT.

LISTA DE ILUSTRAÇÕES

Figura 1. Arquitetura de computação em nuvem 13
Figura 2. Arquitetura de computação em névoa 14
Figura 3. Arquitetura da meta plataforma KNoT 15
Figura 4. Diagrama de serviços que se comunicam com a KNoT Cloud 16
Figura 5. Tela de edição de fluxos do Node-RED 18
Figura 6. Tela de seleção da cloud na KNoT WebUI 19
Figura 7. Tela de configuração da cloud na KNoT WebUI 20
Figura 8. Tela de validação de usuário na KNoT WebUI 20
Figura 9. Tela de dashboard da KNoT WebUI 21
Figura 10. Tela de adição do VirtualThing na KNoT WebUI 21
Figura 11. Fluxo de interação com pino 13 do arduino 22
Figura 12. Interação do Node-RED KNoT SDK com arquitetura KNoT 23
Figura 13. Tela de configuração com metadados do thing selecionado 27
Figura 14. Interação do node virtual thing com a arquitetura KNoT 28
Figura 15. Interação do node knot cloud com arquitetura KNoT 29
Figura 16. Fluxo exemplo com nodes KNoT SDK 32
Figura 17. Fluxo exemplo com o node cloud thing 34

LISTA DE TABELAS

Tabela 1. Nodes do KNoT SDK 24

LISTA DE SIGLAS

Sigla	Significado
IoT	Internet of Things
KNoT	KNoT Network of Things
API	Application Program Interface
SDK	Software Development Kit
AMQP	Advanced Message Queuing Protocol
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
UI	User Interface

SUMÁRIO

1. Introdução 11
1.1. Motivação 11
1.2. Objetivos 12
1.3. Estrutura do trabalho 12
2. Conceitos básicos 13
2.1. Computação em nuvem 13
2.2. Computação em névoa 14
2.3. KNoT 15
2.3.1. KNoT Cloud 16
2.3.2. KNoT SDK 17
2.3.3. KNoT VirtualThing 17
2.4. Programação por fluxo de dados 18
2.4.1. Node-RED 18
3. Trabalhos relacionados 19
3.1. KNoT WebUI 19
3.2. Node-RED Arduino 22
4. Biblioteca Node-RED/KNoT 24
4.1. Node-RED KNoT SDK 24
4.2. Node-RED KNoT Things 27
4.3. Node-RED KNoT Settings 29
5. Casos de uso 30
5.1. Preparação do ambiente 30
5.2. Caso de uso com os nodes KNoT SDK 31
5.3. Caso de uso com o node cloud thing 32
6. Conclusão 35
6.1. Dificuldades encontradas 35
6.2. Trabalhos futuros 36
REFERÊNCIAS 38
APÊNDICES 41
Apêndice A – Imagens dos nodes por categorias 41
Apêndice B – Imagens das telas de configuração dos nodes 42

1. Introdução

1.1 Motivação

A internet das coisas é uma tecnologia emergente de grande potencial disruptivo. A ideia básica por trás desse conceito é a presença difusa em nosso cotidiano de diversos objetos físicos, como dispositivos, instrumentos, veículos, construções e outras “coisas”, habilitadas com sensores e/ou atuadores, todos conectados a internet, comunicando-se, interagindo e cooperando na realização de tarefas comuns [1]. As aplicações de IoT são diversas e podem aparecer em diferentes cenários de nossa vida, como por exemplo, desde um ar condicionado inteligente controlado pelo smartphone, como na indústria, onde existam sensores integrados a máquinas de uma fábrica que monitoram e analisam dados para realização de manutenção preditiva.

Contudo, conectar um dispositivo inteligente à internet não é uma tarefa simples, pois envolve projetar e desenvolver um hardware com sensores, atuadores, conexão sem fio, e embarcá-lo fisicamente nessa “coisa”. Dessa forma, existem diversos protocolos de comunicação que buscam atender a alguns requisitos relacionados à conectividade como: taxa de transmissão, consumo de energia, alcance, confiabilidade e custo. Após conectada, essa “coisa” precisa se comunicar com um servidor, transmitindo seus dados, para então uma aplicação realizar a leitura dessas informações e poder controlar essa “coisa”. Como vimos, uma solução de IoT envolve diversas áreas da computação, como sistemas embarcados, protocolos de comunicação, computação em nuvem, aplicações móveis e web, inclusive ciência de dados e inteligência artificial. Essa complexidade no desenvolvimento de soluções IoT e a interoperabilidade entre essas soluções são barreiras relevantes que dificultam a maior difusão da IoT nos dias atuais [2].

Diante deste cenário que o KNoT [3] foi criado, como uma meta plataforma para internet das coisas que visa preencher o espaço existente entre as plataformas de hardware e software. Seu objetivo é facilitar o desenvolvimento de soluções, provendo uma infraestrutura habilitadora de IoT fim a fim. O KNoT é open source e conecta as principais plataformas open source existentes para prover uma solução completa de IoT. Entretanto, para se iniciar um projeto utilizando o KNoT, é necessário um conhecimento básico em programação embarcada, redes de computadores e tecnologias web/mobile, tanto para configuração como desenvolvimento das aplicações.

O Node-RED [4], por outro lado, é uma ferramenta de programação por fluxo de dados open source, que permite conectar dispositivos, APIs e serviços de uma maneira interativa e intuitiva. Ele provê um editor no navegador com uma ampla variedade de blocos lógicos, sendo possível conectá-los para a criação de fluxos de execução, sem a necessidade de conhecimentos específicos [5].

1.2 Objetivos

Este trabalho visa realizar um estudo acerca da ferramenta Node-RED, além de verificar a viabilidade de utilizar o Node-RED como um serviço na meta plataforma KNoT e desenvolver uma integração entre essas duas tecnologias, buscando levar a facilidade e simplicidade da criação dos fluxos Node-RED para o KNoT. Como objetivos específicos este trabalho se propõe a criar uma biblioteca de nodes que crie aplicações e configurem componentes do KNoT. Esta biblioteca é definida em três conjuntos de nodes:

- **Node-RED/KNoT SDK**: nodes que executem as operações do KNoT SDK AMQP, para auxiliar na criação de fluxos que interajam com a infraestrutura do KNoT;

- **Node-RED/KNoT Things**: nodes que façam abstração de um thing KNoT, real ou simulado, para auxiliar na criação de aplicações KNoT mais complexas;

- **Node-RED/KNoT Settings**: nodes que configuram componentes do KNoT, simplificando o setup inicial da meta plataforma.

1.3 Estrutura do trabalho

A elaboração deste trabalho foi dividida em 7 capítulos. O primeiro capítulo apresenta uma contextualização e motivação do trabalho, bem como seus objetivos gerais e específicos. No segundo capítulo é dada uma introdução teórica aos principais temas abordados ao longo do trabalho e no terceiro capítulo são introduzidos trabalhos semelhantes, que serviram de inspiração para o projeto.

O quarto capítulo apresenta o desenvolvimento da biblioteca proposta, onde são detalhados cada um dos nodes implementados. No quinto capítulo é realizada uma análise de casos de uso da biblioteca, discutindo os benefícios que cada conjunto de nodes entrega ao KNoT. Por fim, no sexto capítulo são colocadas as conclusões a respeito do trabalho, onde são mostradas as dificuldades encontradas ao longo do mesmo, e também possíveis trabalhos futuros.

2. Conceitos básicos

2.1 Computação em nuvem

Computação em nuvem foi a denominação dada inicialmente a uma categoria de serviços computacionais sofisticados e sob demanda, que surgiram sendo oferecidos por grandes empresas de tecnologia, e desde então esse modelo vem se popularizando bastante ao longo dos anos. A sua principal infraestrutura computacional é chamada nuvem, e é nela onde negócios e/ou indivíduos conseguem acessar aplicações de seu interesse sempre que necessário e a partir de qualquer lugar com acesso a internet. O mais importante princípio por trás desse modelo, é a disponibilização de serviços como capacidade de computação, armazenamento de dados e softwares [6]. A figura 1 apresenta um modelo de arquitetura em nuvem.

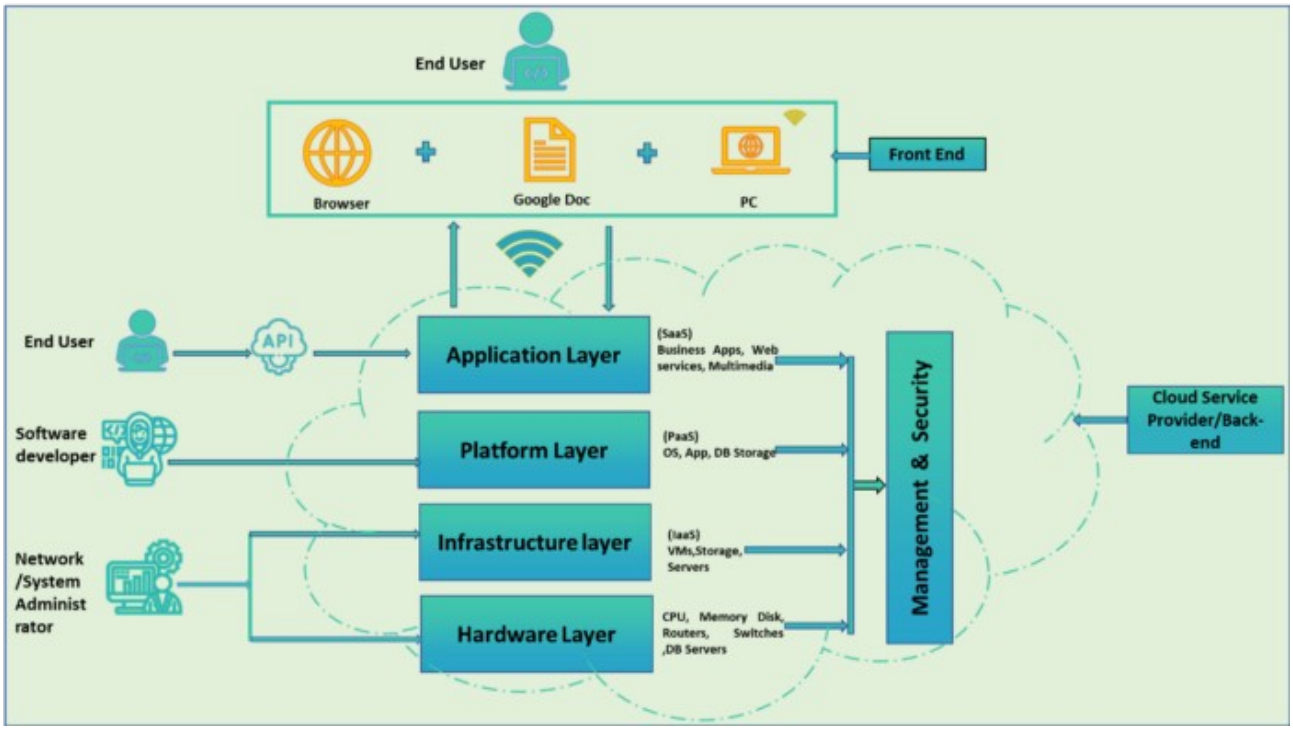

Figura 1. Arquitetura de computação em nuvem
Fonte: https://medium.com/@click2cloud.contact/cloud-computing-architecture-a-blueprint-of-cloud-infrastructure-b932aa9461b9

No contexto de internet das coisas, à medida que mais coisas forem sendo acopladas com eletrônica embarcada e conectividade, a quantidade de informação gerada por essas coisas inteligentes será consideravelmente grande, tornando o armazenamento e processamento desses dados inviáveis de serem realizados por essas coisas, que possuem limitações como o gerenciamento de energia e o armazenamento de dados. Neste cenário, a computação em nuvem entra com um importante papel, pois se torna o centro de interconectividade entre as coisas, sendo

responsável por gerir os dados recebidos por cada uma. Contudo, a integração entre computação em nuvem e internet das coisas não é tão simples e possui diversos desafios [7]. O maior deles é o aumento expressivo de tráfego na infraestrutura de rede, decorrente da imensa quantidade de coisas enviando e recebendo dados na internet. O elevado volume de tráfego pode levar essas redes, que conectam os centros de computação em nuvem aos dispositivos, a se tornarem gargalos se elas não forem adequadamente projetadas para isso. Uma solução para reduzir esse tráfego gerado pela integração entre IoT e computação em nuvem é o modelo de computação em névoa, que vem se tornando bastante popular neste contexto [8].

2.2 Computação em névoa

O termo computação em névoa (ou computação de borda) significa que as tarefas não mais serão executadas em uma nuvem centralizada, pois sistemas em névoa operam na borda da infraestrutura de rede. Esta prática visa trazer processos e recursos para a ponta da rede, deixando-a o mais próximo possível dos dispositivos e desobstruindo a comunicação com o sistema em nuvem. A computação em névoa ajuda a reduzir a necessidade de largura de banda, pois ela diminui os dados enviados à nuvem, processando-os e realizando uma triagem do que realmente precisa ser encaminhado. Esse tipo de estratégia distribuída ajuda a reduzir custos, aumentando a eficiência do sistema, e estende o paradigma da computação em nuvem para a borda da rede [9]. A figura 2 demonstra a arquitetura de computação em névoa.

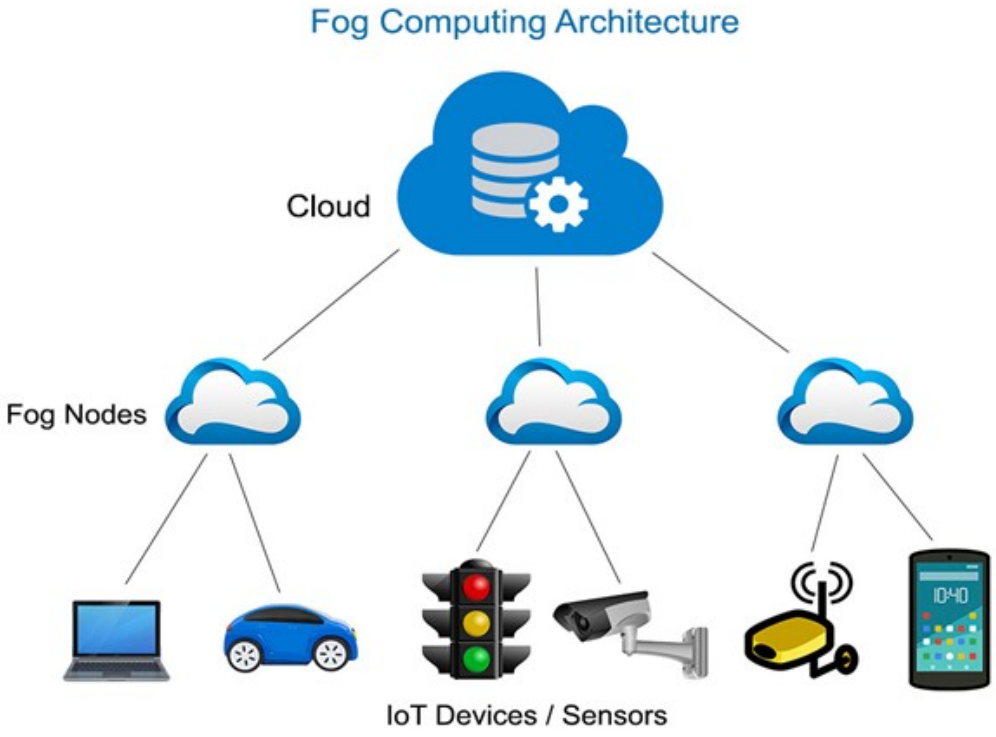

Figura 2. Arquitetura de computação em névoa
Fonte: https://www.terminalworks.com/blog/post/2017/05/13/fog-computing

2.3 KNoT

Desenvolvido pelo grupo de pesquisa em IoT do CESAR, a meta plataforma KNoT tem o objetivo de integrar as mais diversas plataformas de hardware e software para IoT, com foco em hardware de baixo custo e software de fácil usabilidade, a fim de prover uma solução de IoT fim a fim [10]. A figura 3 apresenta a arquitetura básica do KNoT. Abaixo estão descritas algumas das principais características do KNoT:

- **Faz uso de plataformas já consolidadas**: o KNoT é uma meta plataforma que foca na implementação e integração entre plataformas já existentes de hardware e software para IoT.
- **Código aberto**: todos os softwares e hardwares usados no KNoT possuem licença de código aberto permissiva, o que dá liberdade ao desenvolvedor para utilizá-lo em seu próprio projeto comercial, e também para contribuir com o projeto KNoT.
- **Arquitetura de nuvem distribuída**: o KNoT implementa o conceito de computação em névoa, onde instâncias locais e menores da nuvem recebem diretamente os dados dos dispositivos que estão fisicamente próximos, sincronizando-os hierarquicamente de uma maneira que permita a conexão de milhões desses dispositivos.
- **Modelo de dados semânticos**: o KNoT define um modelo de dados semânticos na camada de aplicação, fazendo com que todas as aplicações na plataforma conheçam o formato de dados umas das outras, o que facilita o compartilhamento de dados entre elas.

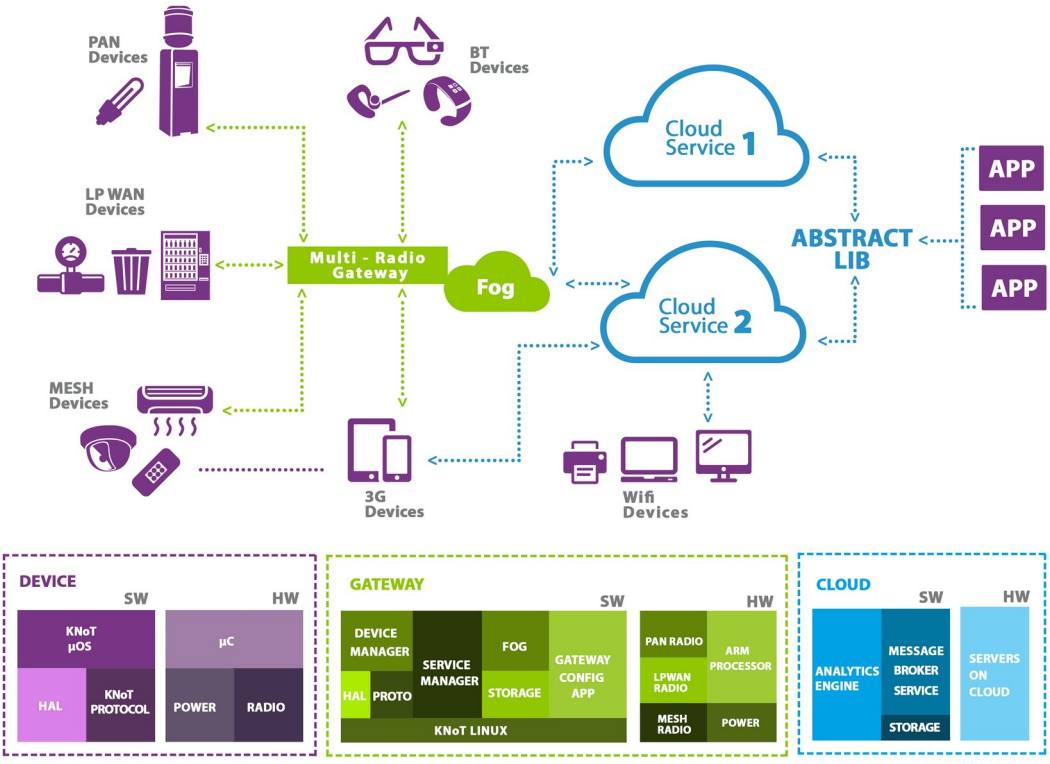

Figura 3. Arquitetura da meta plataforma KNoT
Fonte: https://knot.cesar.org.br/#architecture

Como apresentado na figura 3, o KNoT possui três principais componentes: o **KNoT Thing**, o **KNoT Gateway** e a **KNoT Cloud**. Segue abaixo uma descrição acerca de cada um:

- **KNoT Thing**: é a designação dada a cada um dos dispositivos, providos de alimentação, microcontrolador e um módulo RF, onde o desenvolvedor pode conectar os sensores e atuadores que lhe forem mais convenientes.
- **KNoT Gateway**: atua como um proxy, traduzindo o protocolo KNoT entre a tecnologia usada pelo KNoT Thing para JSON ou XML, usados na plataforma da nuvem. É composto por uma placa ARM, 3 a 4 diferentes módulos RF (bluetooth, low PAN, mesh, etc) para se conectar com diferentes dispositivos, conexão ethernet e uma distribuição linux embarcada personalizada, chamada KNoT Linux. O gateway possui também uma instância reduzida da KNoT Cloud, onde dados de dispositivos locais podem ser acessados diretamente [11].
- **KNoT Cloud**: é responsável por coletar todos dados publicados no KNoT Gateway e atuar como um ponto de conexão, interagindo com as aplicações. Também é responsável pelo roteamento de mensagens para aplicações e dispositivos. Mais detalhes sobre a KNoT Cloud são apresentados na próxima seção.

2.3.1 KNoT Cloud

A KNoT Cloud [12] é uma solução em nuvem também desenvolvida pelo time de pesquisa em IoT do CESAR, com o propósito de conectar dispositivos IoT, com facilidade e segurança, a infraestruturas mínimas de gerenciamento de mensagens.

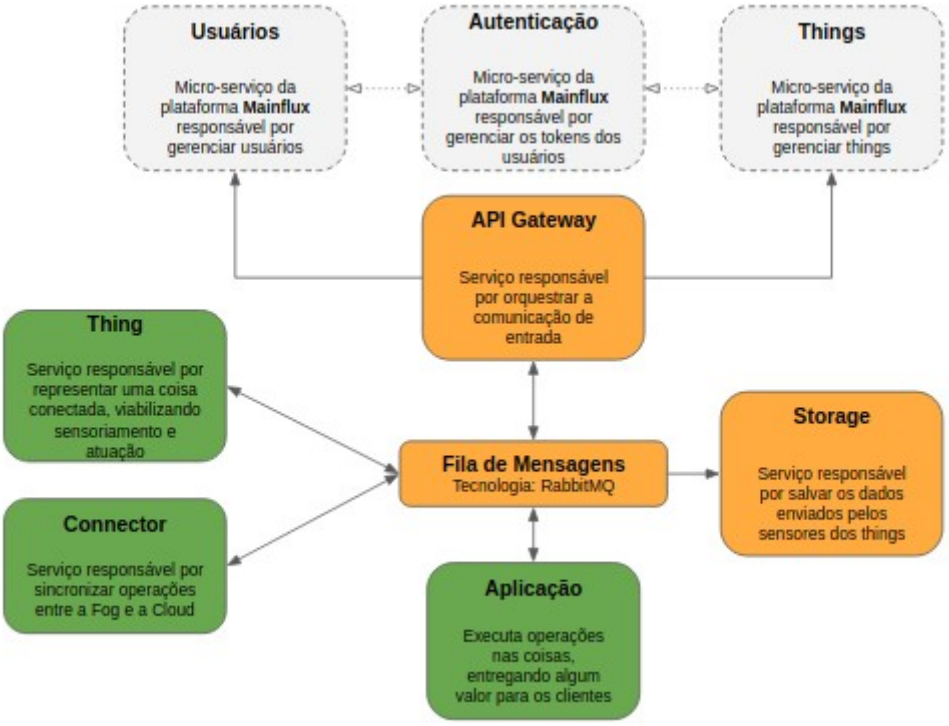

Figura 4. Diagrama de serviços que se comunicam com a KNoT Cloud

A KNoT Cloud possui interfaces com protocolos AMQP e HTTP, utilizados na comunicação com aplicações e outros serviços do KNoT. A figura 4 apresenta os serviços presentes na KNoT Cloud (laranja e cinza), e também os serviços com os quais ela se comunica (verde).

Além de se comunicar diretamente com outros serviços do KNoT, esta define a API do KNoT SDK, o que permite a criação de aplicações que gerenciam usuários e things na infraestrutura da cloud. Essa solução torna fácil a criação de aplicações IoT que coletam, processam, analisam e atuam nos dados gerados pelos dispositivos conectados [13].

2.3.2 KNoT SDK

O KNoT SDK é a principal ferramenta utilizada na criação de aplicações IoT com o KNoT. É uma biblioteca implementada do lado cliente, para Node.js e navegadores, que objetiva ajudar desenvolvedores a criarem soluções com auxílio da KNoT Cloud [14]. Atualmente possui três interfaces, cada uma com seu propósito específico, descritos abaixo:

- **AMQP SDK** [15]: reponsável por se comunicar com o broker AMQP (RabbitMQ), publicando mensagens e se inscrevendo em tópicos, a fim de realizar operações de controle e interagir com os things KNoT.
- **Authenticator SDK** [16]: responsável por se comunicar com o KNoT BabelTower [17], gerenciando a criação de usuários e tokens na KNoT Cloud.
- **Storage SDK** [18]: responsável por interagir com o KNoT Storage [19], acessando dados registrados pelos things.

2.3.3 KNoT VirtualThing

O KNoT VirtualThing [20] é um serviço responsável por criar uma simulação robusta de um thing KNoT, simulando inclusive a máquina de estados do thing. Este cria uma abstração que permite a interação com a KNoT Cloud, de um thing virtualizado, através do protocolo industrial Modbus. Ele foi criado com o objetivo inicial de possibilitar a inclusão de PLC’s, dispositivos amplamente utilizados na indústria, na infraestrutura do KNoT.

2.4 Programação por fluxo de dados

A programação por fluxo de dados é um paradigma computacional que consiste em modelar programas como grafos direcionados, onde são conectados diversos blocos lógicos (ou nodes) que trocam informações ao longo desse fluxo. Isso agrega mais flexibilidade a este tipo de programa, mantendo a facilidade de criação dele [5].

No contexto de internet das coisas, esse paradigma permite a criação de soluções em tempo real para IoT de forma simples, sem a exigência do programador aprender novas tecnologias e protocolos, implementar códigos para processamento de dados e então integrar todos os serviços criados. Por este motivo, muitas linguagens de programação visual por fluxo de dados vem se popularizando, dentre as quais o Node-RED vem adquirindo um certo destaque. Além disso, uma evolução natural para o cenário de IoT seria embarcar esses programas em gateways e outros dispositivos de borda, tornando as soluções mais distribuídas e eficientes [21].

2.4.1 Node-RED

O Node-RED é uma linguagem de programação visual por fluxo de dados, e possui um editor web gráfico que auxilia a conexão entre dispositivos físicos e API's. É escrito em Node.js e os programas criados pela sua interface são chamados de fluxos, consistindo em vários nodes (blocos lógicos previamente definidos) conectados por fios. O editor gráfico possui por uma lista de modelos de nodes, que podem ser arrastados e soltado na tela de fluxo. A figura 5 mostra a tela de edição do Node-RED.

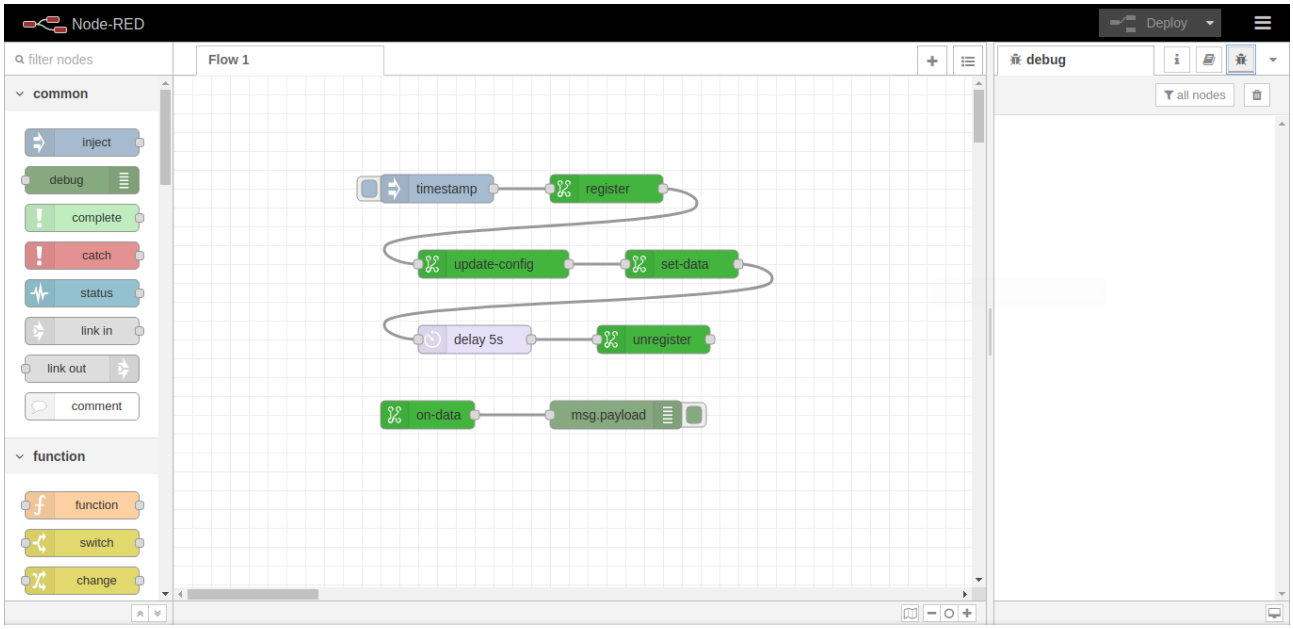

Figura 5. Tela de edição de fluxos do Node-RED

3. Trabalhos relacionados

Neste capítulo serão apresentados os trabalhos semelhantes que serviram como inspiração para a elaboração deste trabalho. São eles o KNoT WebUI, serviço integrante do componente KNoT Gateway, e o Node-RED Arduino, biblioteca de nodes do Node-RED para interação com placas arduino. Nas seções seguintes esses trabalhos serão brevemente descritos, e no final de cada seção serão discutidos os pontos de semelhança entre cada um e o trabalho proposto.

3.1 KNoT WebUI

Desenvolvido pelo grupo de pesquisa KNoT do CESAR, o KNoT WebUI [22] é um serviço pertencente ao componente KNoT Gateway, da plataforma KNoT. É uma aplicação web, e tem como principais objetivos realizar a configuração inicial do KNoT Gateway [11] e gerenciar os things KNoT registrados no mesmo. É desenvolvido tendo como base a MEAN stack (MongoDB, Express.js, AngularJS e Node.js), que são soluções de código aberto baseados em JavaScript. Sua comunicação com os outros serviços do KNoT Gateway é realizada pela interface Modbus, e o rastreamento dos things registrados se dá por meio do protocolo AMQP, através de troca de mensagens com o broker RabbitMQ do gateway.

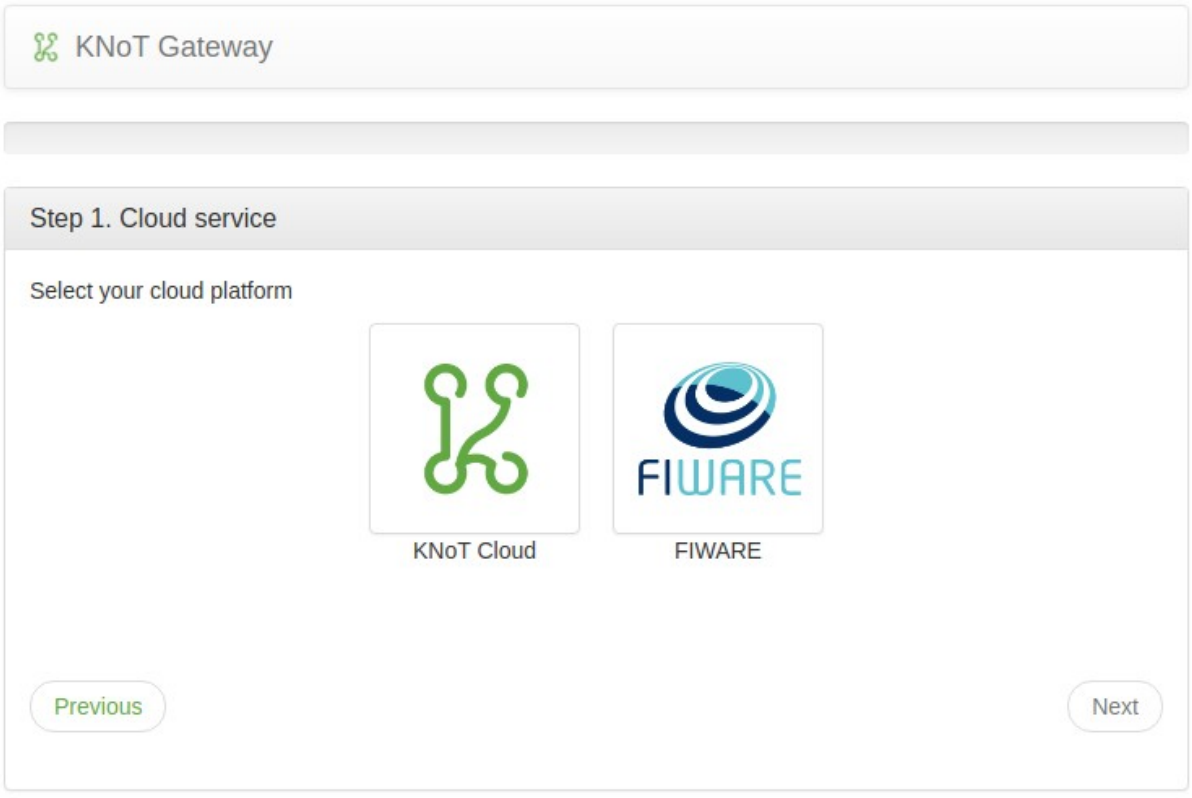

Figura 6. Tela de seleção da cloud na KNoT WebUI

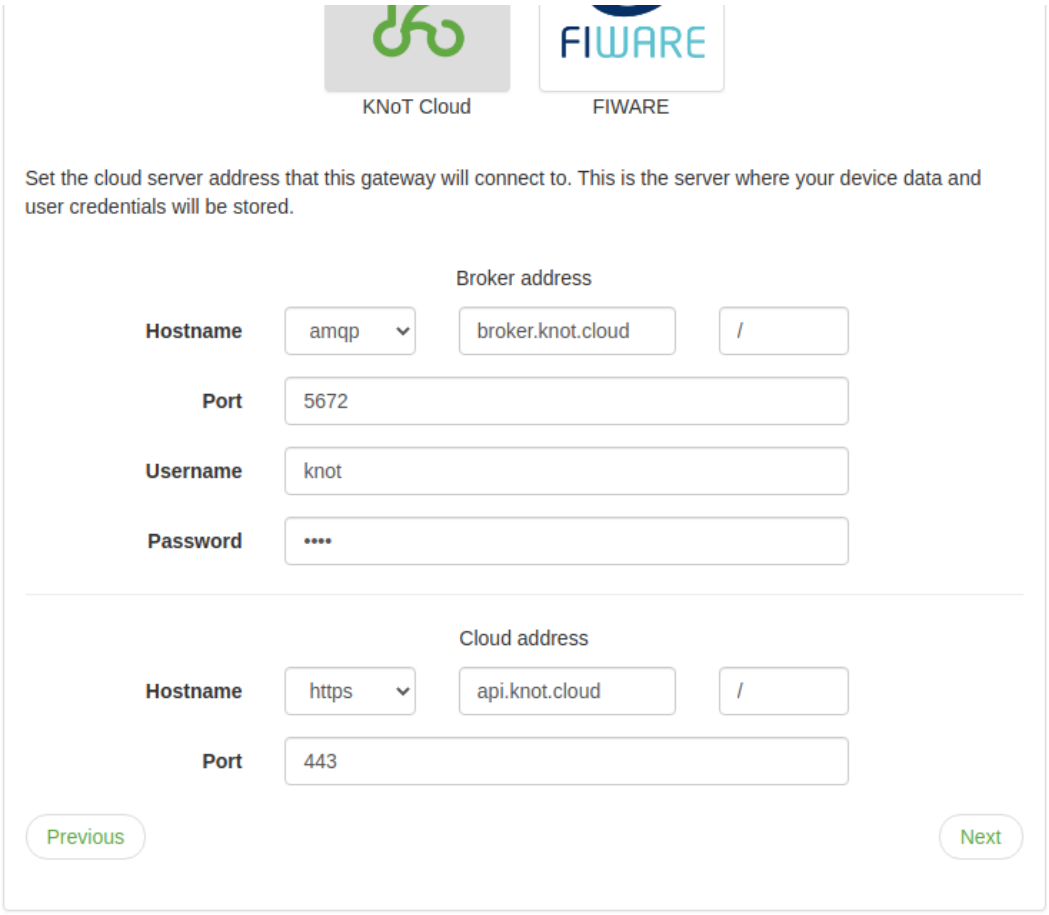

Figura 7. Tela de configuração da cloud na KNoT WebUI

A configuração do gateway se dá inicialmente com a seleção da plataforma de cloud e configuração dos endpoints da mesma, como pode ser visto nas figuras 6 e 7 respectivamente, seguido pela criação/autenticação de um usuário na plataforma KNoT, vide figura 8.

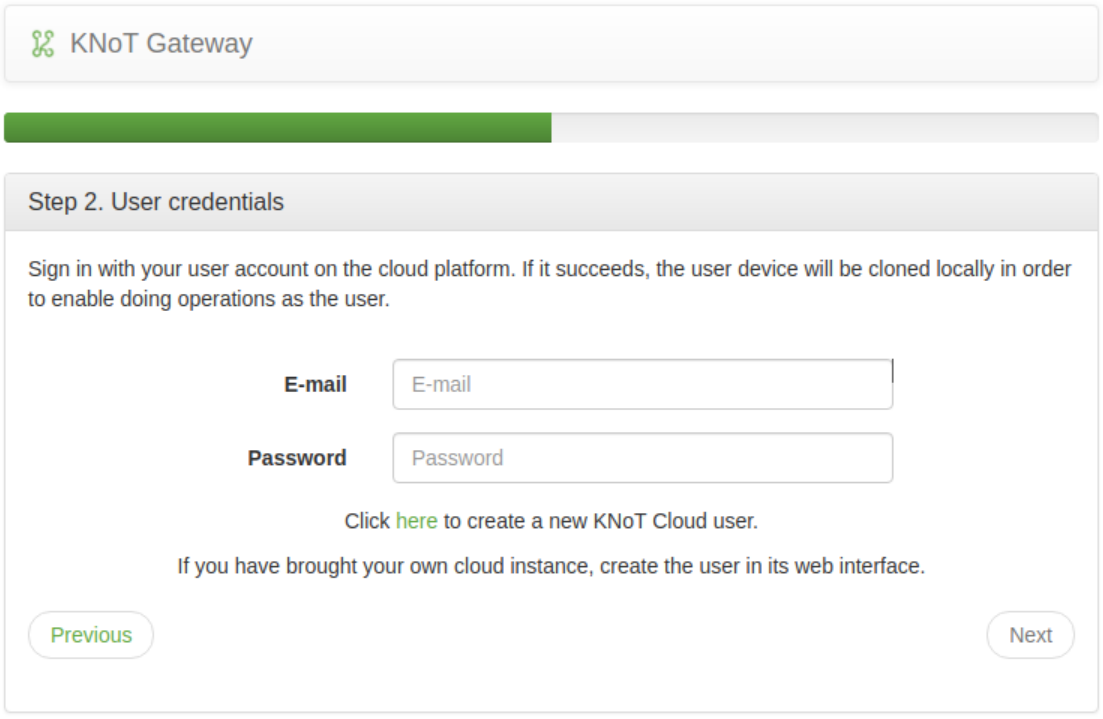

Figura 8. Tela de validação de usuário na KNoT WebUI

O gerenciamento de things é realizado em uma dashboard, onde é possível visualizar o schema e o último valor publicado por cada um dos things registrados. Além de configurar os things virtualizados pelo serviço KNoT VirtualThing. A figura 9 apresenta a tela de monitoramento de things, e a figura 10 mostra como se configura um thing virtualizado.

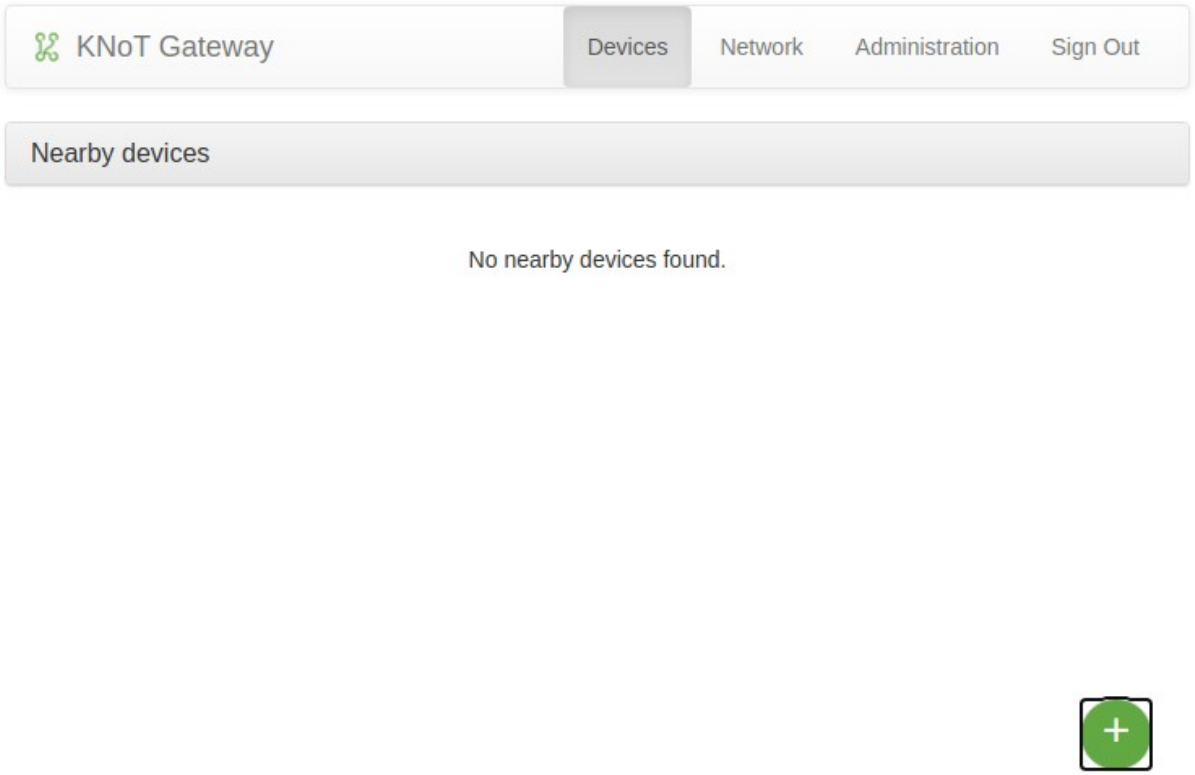

Figura 9. Tela de dashboard da KNoT WebUI

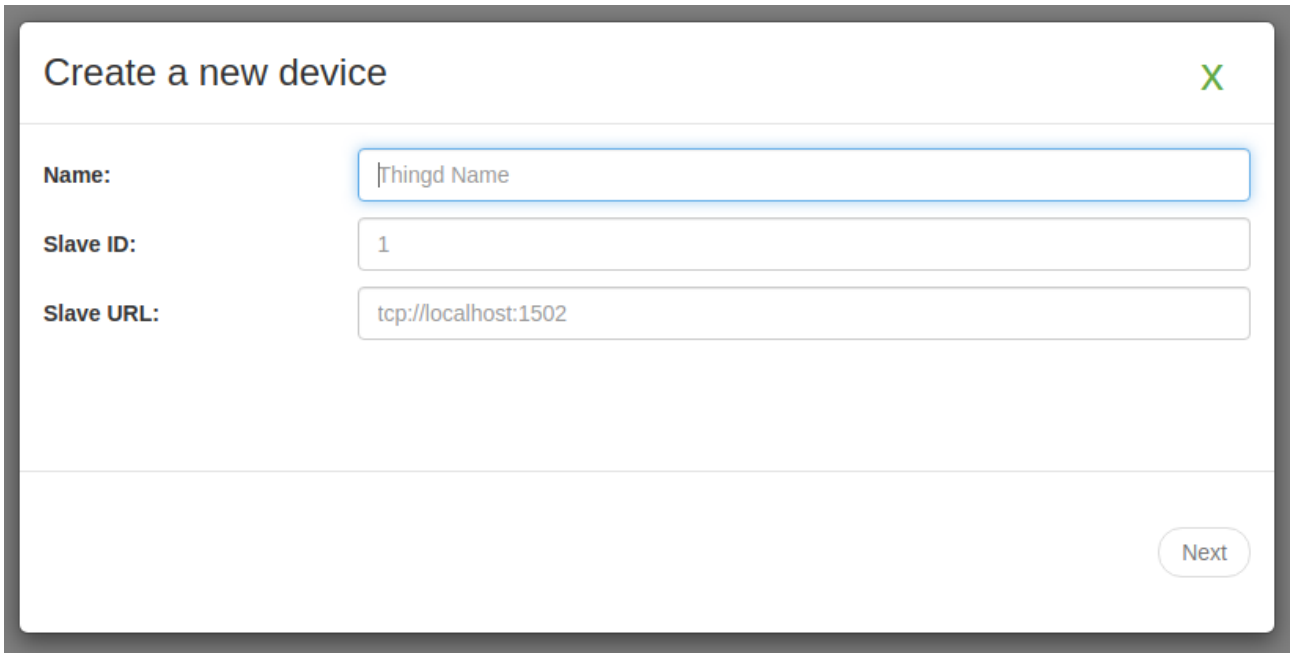

Figura 10. Tela de adição do VirtualThing na KNoT WebUI

A principal inspiração que o KNoT WebUI deu ao trabalho proposto foi a capacidade de visualização dos valores publicados por things KNoT registrados no gateway e de criação de things virtuais de uma maneira prática ao usuário, utilizando o KNoT VirtualThing. Consequentemente surgiu a ideia de implementar o node que representa things reais e os nodes que simulam things, seja um simples thing registrado no serviço de cloud, seja esse thing uma instância do KNoT VirtualThing. A parte de configuração do web-UI também serviu de inspiração para a criação de nodes de configuração, tanto do próprio gateway, quanto da stack da cloud, juntamente com seus respectivos serviços.

Apesar de cumprir bem os seus papéis, o KNoT WebUI é um serviço que possui elevada complexidade de código, o que dificulta ainda mais a sua manutenção. Uma solução viável seria distribuir suas funcionalidades em outros serviços mais simples. Por este motivo, o trabalho pode também contribuir para a plataforma KNoT como um possível substituto do KNoT WebUI no futuro pois consegue descentralizar essas operações em diferentes nodes.

3.2 Node-RED Arduino

O Node-RED Arduino por sua vez é um node criado e oficialmente mantido pelo projeto Node-RED [23], e tem como principal objetivo interagir com placas arduino conectadas via porta serial ao dispositivo que está rodando a instância do Node-RED. Essa comunicação se dá em via dupla, tanto o arduino enviando informações, que são percebidas pelo node de leitura, quanto ele recebendo também informações, do node de escrita. Tudo isso utilizando o protocolo da camada de aplicação firmata.

As figuras 12 e 11 abaixo mostram respectivamente a tela de configuração do node de escrita no arduino e um exemplo de fluxo onde o usuário consegue, através da interface web do Node-RED, ligar e desligar um LED conectado no pino 13 do arduino [24].

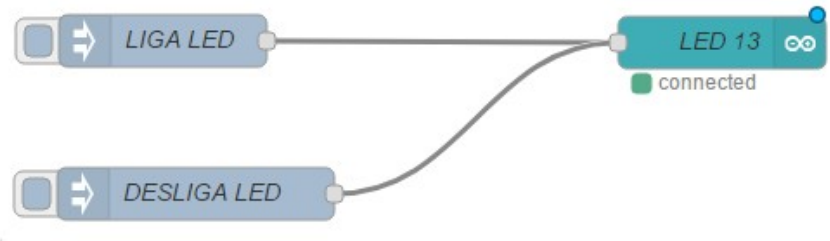

Figura 11. Fluxo de interação com pino 13 do arduino
Fonte: https://www.filipeflop.com/blog/primeiros-passos-node-red-arduino-uno/

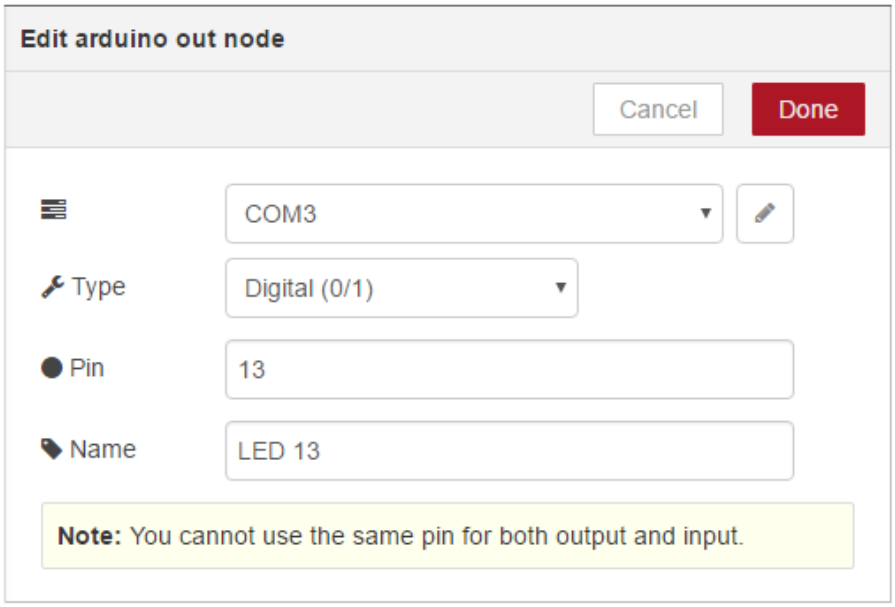

Figura 12. Tela de configuração do node arduino out
Fonte: https://www.filipeflop.com/blog/primeiros-passos-node-red-arduino-uno/

Esse tipo de aplicação com o node arduino, onde o usuário consegue ler e escrever dados nos things de maneira simples e prática, serviu de inspiração para criação de nodes que executassem as operações do KNoT SDK, o qual possibilita a interação direta com things, em cima da infraestrutura do KNoT. Para permitir a criação de aplicações que se comunicam diretamente com os things KNoT através de fluxos do Node-RED, foi decidida a implementação desses nodes do KNoT SDK.

4. Biblioteca Node-RED/KNoT

Este capítulo apresenta a biblioteca de nodes do Node-RED para a meta plataforma KNoT. Esta foi desenvolvida com o auxílio de tecnologias como Node.js, Docker e Git. Nas seções seguintes serão apresentados cada um dos grupos de nodes implementados, e as respectivas discussões. Todo o código está disponível em um repositório público do GitHub, para utilização e colaboração de qualquer pessoa interessada.

4.1 Node-RED KNoT SDK

O desenvolvimento da biblioteca foi iniciado pelos nodes que realizam as operações do KNoT SDK. Como ambos são implementados em Node.js, a complexidade inicial maior foi o aprendizado da ferramenta Node-RED e a criação de nodes para a mesma. Após o período inicial de estudos, foi implementado um node de configuração **amqp broker**, que serviu de base para os nodes listados na tabela 1 abaixo:

Nome	**Descrição**	**Parâmetros**
register	Registra um novo thing	• broker AMQP • ID do thing • nome do thing
unregister	Remove um thing já registrado	• broker AMQP • ID do thing
update config	Atualiza o metadados de configuração de um thing	• broker AMQP • ID do thing • lista de definições de configuração
publish data	Publica dados no KNoT Cloud Storage	• broker AMQP • ID do thing • lista de dados a serem publicados
set data	Envia dados para um thing registrado	• broker AMQP • ID do thing • lista de dados a serem publicados
get data	Requisita dados de um thing	• broker AMQP • ID do thing • lista de sensores do thing
get devices	Requisita a lista de things registrados	• broker AMQP
on data	Escuta a eventos de dados	• broker AMQP • evento [publish, set, get] • ID do thing (apenas para set e get) • opção de executar apenas uma vez

Tabela 1. Nodes do KNoT SDK

Esses nodes foram desenvolvidos em Node.js/JavaScript, HTML e CSS, pois são as tecnologias utilizadas na implementação do Node-RED. Cada node possui dois arquivos: um .**html** e outro **.js**. O arquivo HTML é responsável por definir o estilo e comportamento da tela de configuração na interface web gráfica do Node-RED, e por isso possui tags de estilo em CSS e de script em JavaScript. O arquivo JS por outro lado, é todo escrito em Node.js, e descreve o comportamento do node durante a execução do fluxo Node-RED, montado na interface gráfica.

Para todos os nodes, com exceção do amqp broker, os parâmetros das operações são passados via configuração ou então pela mensagem de entrada. As imagens dos nodes e as telas de configuração, estão respectivamente nos apêndices A e B. A seguir temos uma descrição mais detalhada acerca de cada um:

- **amqp broker**: é um node de configuração. Utilizado pelos outros nodes para viabilizar a troca de mensagens AMQP com o broker da KNoT Cloud.

- **register**: ao receber um sinal de entrada, registra um thing novo com ID e nome na KNoT Cloud. Emite um evento ao concluir a operação, passando os metadados do thing.

- **unregister**: ao receber um sinal de entrada, remove um thing registrado na KNoT Cloud com o ID especificado. Emite um evento ao concluir a operação.

- **update config**: ao receber um sinal de entrada, atualiza os dados de configuração de um thing registrado na KNoT Cloud. Emite um evento ao concluir a operação, passando os metadados do thing.

- **publish data**: ao receber um sinal de entrada, publica um dado de sensor, de um thing já registrado, na KNoT Cloud. Emite um evento ao concluir, passando os metadados do thing associado e o próprio dado publicado.

- **set data**: ao receber um sinal de entrada, atualiza o dado do sensor de um thing na KNoT Cloud. Emite um evento ao concluir, passando os metadados do thing e o dado atualizado.

- **get data**: ao receber um sinal de entrada, requisita a publicação do dado de um sensor do thing especificado. Emite um evento ao concluir, passando os metadados do thing e o sensor cujo dado foi pedido.

- **get devices**: ao receber um sinal de entrada, faz uma requisição à KNoT Cloud para listar todos os things registrados, juntamente com os metadados de cada um. Emite um evento ao concluir, passando a lista de things com os respectivos metadados.

- **on data**: não receber nenhum evento de entrada. Ele configura e ativa um listener, emitindo um evento toda vez que o listener receber uma mensagem. Envia os dados recebidos junto com a mensagem do evento.

Esses nodes funcionam como um embrulho ao redor do KNoT SDK, que se comunica diretamente com o broker AMQP da KNoT Cloud, enviando mensagens já previamente definidas pela documentação do KNoT. Como podemos observar na figura 13, o conjunto de nodes Node-RED KNoT SDK interage com a infraestrutura do KNoT por meio de operações com o KNoT SDK, que por sua vez consegue se comunicar com os things através do broker de mesagens e API Gateway presente na KNoT Cloud.

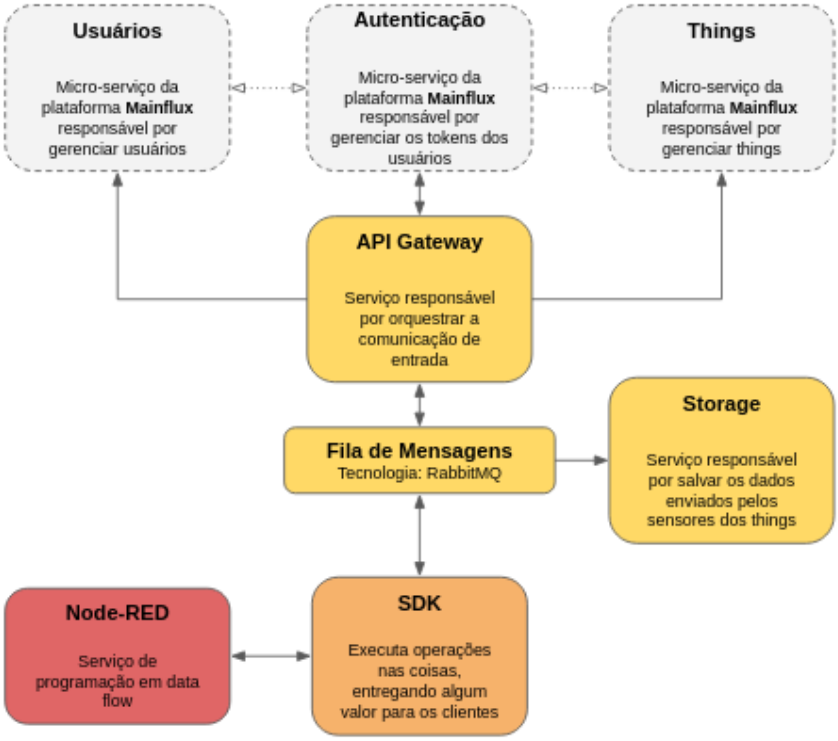

Figura 12. Interação do Node-RED KNoT SDK com arquitetura KNoT

A elaboração desses nodes já permite a criação de aplicações KNoT com fluxos Node-RED. Isso significa que qualquer outro node, seja um padrão do Node-RED ou de alguma outra

tecnologia (AWS, Google Cloud, etc), interaja e se comunique com things KNoT sem grandes dificuldades.

Até então, para criar uma aplicação desse tipo seriam necessários conhecimentos em Node.js e nas tecnologias/bibliotecas a serem utilizadas. Com esse conjunto de nodes é possível criar aplicações que façam a interação dessas diferentes plataformas com o KNoT, sem a necessidade de conhecer o funcionamento detalhado de uma tecnologia específica ou de escrever linhas de código.

4.2 Node-RED KNoT Things

Este conjunto de nodes tem como principal objetivo simular e representar things KNoT, possibilitando a interação deles com outros nodes através de fluxos Node-RED. Cada node aqui tem um propósito específico, descritos a seguir:

- **thing**: este node faz a representação de um thing KNoT real, selecionado a partir de uma lista que contém todos os thing registrados na KNoT Cloud. Os metadados do thing escolhido ficam na tela de configuração do node, como mostrado na figura 14. Ao receber um sinal de entrada, atualiza os dados recebidos no thing. Emite eventos sempre que o thing publica algum dado, enviando esse dado na mensagem.

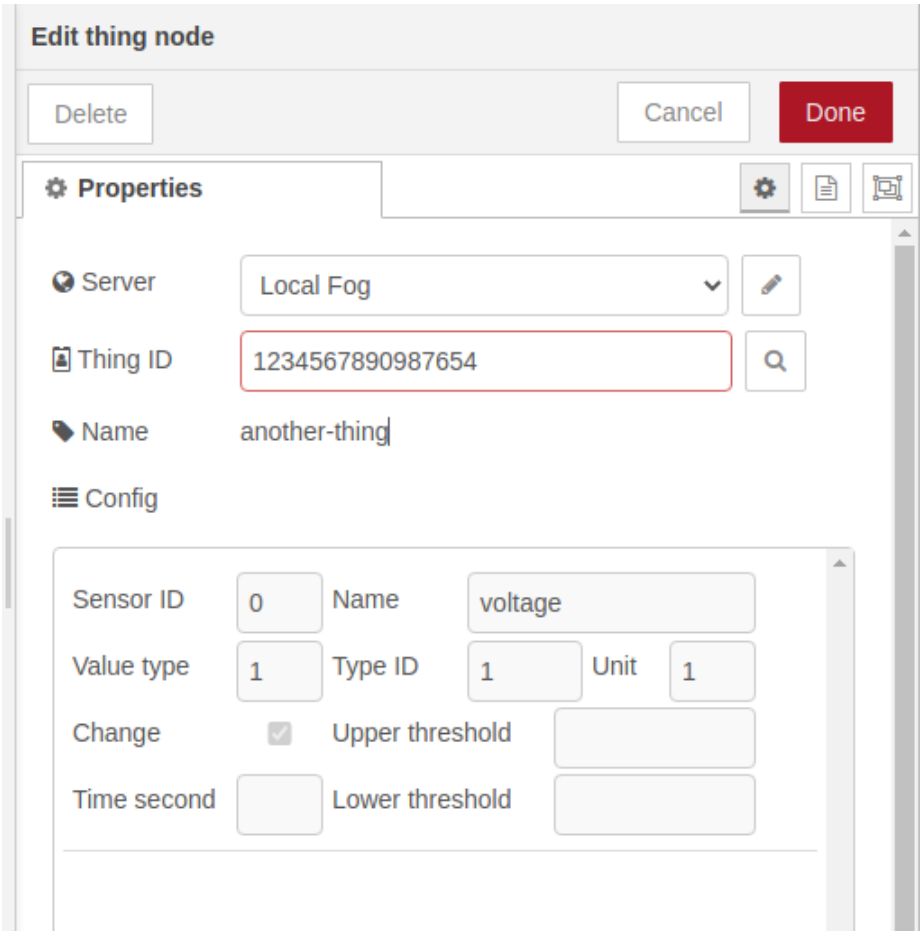

Figura 13. Tela de configuração com os metadados do thing selecionado

- **cloud thing**: registra um thing simulado na KNoT Cloud, e o remove ao finalizar a execução do fluxo. Recebe dados pelo sinal de entrada, e os publica na KNoT Cloud ao recebê-los. Emite eventos referentes a atualização e requisição de dados, fazendo com que quem for desenvolver o fluxo tenha controle também sobre esses eventos relacionados ao thing.

- **virtual thing**: configura e interage com um KNoT VirtualThing. Atualiza os dados do thing ao receber um sinal de entrada e emite eventos quando o mesmo publica algum dado. Este node interage diretamente com o serviço do VirtualThing, o configurando e executando, por meio de um script Docker. Depois de configurar o thing, interage com ele por meio do KNoT SDK. A figura 14 mostra onde esse node atua, que diferentemente dos outros nodes de things, que interagem com a infraestrutura KNoT unicamente através do KNoT SDK, este também interage diretamente com o KNoT VirtualThing.

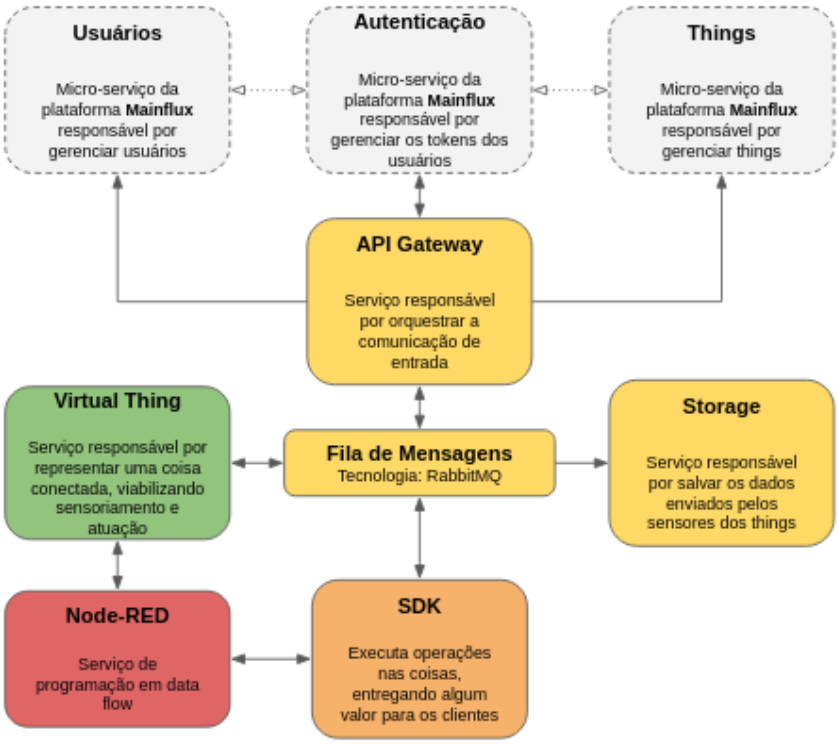

Figura 14. Interação do node virtual thing com a arquitetura KNoT

Com esse conjunto de nodes, que simulam e representam things KNoT, é possível realizar uma abstração maior de interação nos fluxos do Node-RED. Com o **cloud thing** por exemplo, é possível fazer uma simulação de um thing KNoT de forma mais simples e compacta, do que conectar vários nodes SDK interligados.

Com o **thing** consegue-se ter um monitoramento e interação direta com um thing KNoT real, sem a necessidade de criar um código com o KNoT SDK unicamente para se verificar o estado e as mudanças em um thing específico. E finalmente, com o **virtual thing** é possível configurar e executar o serviço KNoT VirtualThing sem a necessidade de configurar os diversos arquivos necessários ou executar nenhum comando, e também monitorá-lo e interagir com seus dados de forma simples e prática.

4.3 Node-RED KNoT Settings

Esse conjunto possui apenas um único node, com o propósito de configurar e executar os serviços presentes na KNoT Cloud. A configuração da stack KNoT Cloud se dá por meio de variáveis de ambiente, descritas em arquivos .env, interpretados pelo software Docker, que constrói e executa imagens relativas a cada serviço da stack.

Esse processo pode ser trabalhaso, e levar a confusões e erros do usuário, dado o grande número de serviços e parâmetros de configuração da stack. Com esse node, nomeado **knot cloud**, é possível realizar essas tarefas de forma simples e rápida, por meio do Node-RED. A figura 15 abaixo demonstra como esse node interage com a arquitetura KNoT:

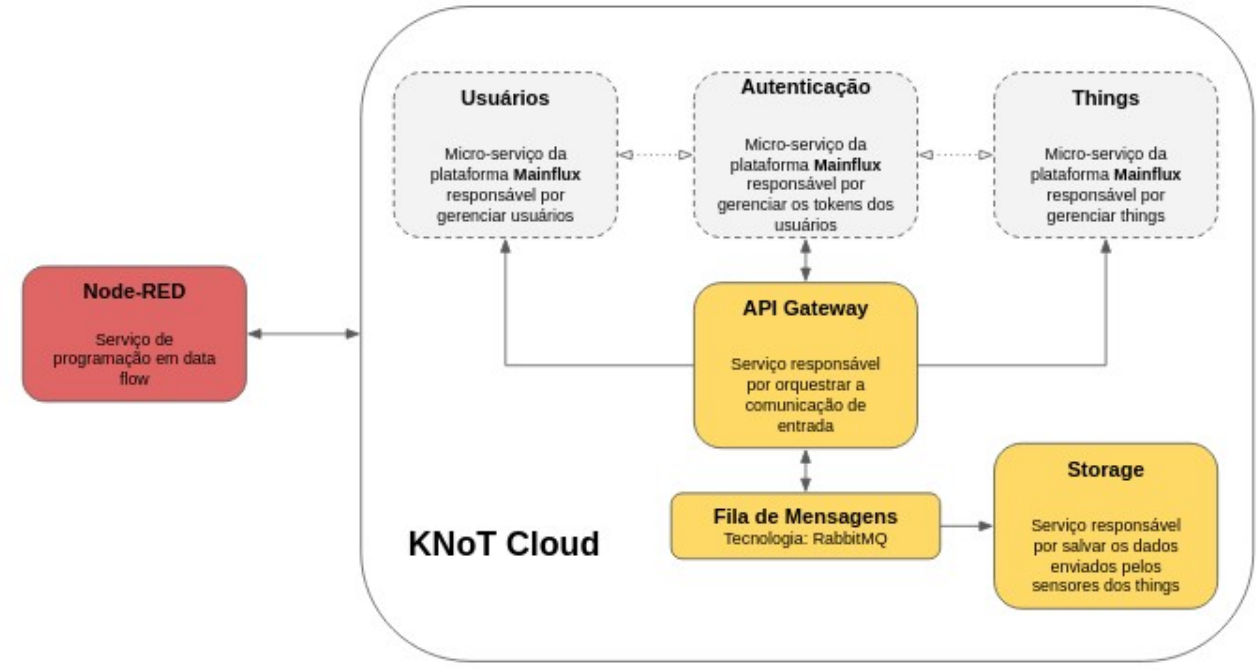

Figura 15. Interação do node knot cloud com arquitetura KNoT

5. Casos de uso

Neste capítulo são apresentados os casos de uso, onde são apresentados exemplos de aplicações práticas para os nodes apresentados no capítulo anterior. As seções a seguir descrevem o passo a passo para executar cada caso e como cada conjunto de nodes colabora com a meta plataforma KNoT.

5.1 Preparação do ambiente

Para execução dos casos de teste, foi necessária a instalação prévia das seguintes plataformas/bibliotecas:

- Node.js
- Node-RED
- Node-RED/KNoT
- KNoT Cloud CLI
- Docker

Instaladas as dependências acima, juntamente com a biblioteca de nodes do KNoT, disponível no link https://github.com/esdrasjnr/node-red-contrib-knot, para a execução dos nodes também foi necessário ter uma stack da KNoT Cloud rodando, pois ela é com ela que a maioria dos nodes implementados vão se comunicar através de suas operações. Para isso existem duas alternativas: configurar e executar cada serviço da stack individualmente ou utilizar a imagem docker da KNoT Cloud. Para ambas alternativas, a documentação necessária para subir a stack está presente no link https://github.com/CESARBR/knot-cloud#knot-cloud. Para este trabalho foi utilizada a imagem docker, por sua simplicidade de configuração e o fácil gerenciamento de recursos. Atualmente existem duas versão de stack da KNoT Cloud, esse trabalho foi realizado tomando como base a implementação mais recente, nomeada KNoT Cloud core.

Após subir a stack da KNoT Cloud, foi configurado o node **amqp broker**, abrindo um node qualquer e adicionando um novo server com as informações do broker da stack. Como a stack foi configurada com os valores padrões, o endereço do broker local foi definido como **amqp://knot:knot@localhost:5672.** O token de usuário usado aqui foi o mesmo gerado durante a configuração da stack. Aqui vale ressaltar que o token gerado tem um tempo padrão de expiração de 8 horas, sendo necessário gerar um novo após esse período, o que pode ser feito facilmente com auxílio da KNoT Cloud CLI.

5.2 Caso de uso com os nodes KNoT SDK

O objetivo deste primeiro exemplo é demonstrar como utilizar os nodes do KNoT SDK para criar interações simples com things KNoT. Este caso de uso é um fluxo onde um thing, contendo um sensor de tensão, é registrado na stack, em seguida um dado é enviado para atualizar o valor desse sensor e no final o thing é removido da stack. Para implementação desse fluxo, foram usados os seguintes nodes:

- **inject**: biblioteca padrão do Node-RED
- **debug**: biblioteca padrão do Node-RED
- **register**: biblioteca Node-RED/KNoT
- **update config**: biblioteca Node-RED/KNoT
- **set data**: biblioteca Node-RED/KNoT
- **on data**: biblioteca Node-RED/KNoT
- **unregister**: biblioteca Node-RED/KNoT

Os nodes **inject** e **debug** servem apenas para interação do usuário com o fluxo, em tempo de execução, podendo ser substituídos por quaisquer outros nodes a depender da finalidade. O **inject** envia sinais para disparar os outros nodes, e o **debug** mostra no console os dados recebidos pelo node on data. Os nodes do KNoT SDK são configurados com os seguintes parâmetros:

- AMQP hostname: **localhost**
- AMQP port: **5672**
- AMQP username: **knot**
- AMQP password: **knot**
- Thing ID: **1234567890987654**
- Thing name: **my-thing**
- Config:
  - Sensor ID: **0**
  - Name: **voltage-sensor**
  - Value Type: **1** (tipo inteiro)
  - Type ID: **1** (valor de tensão)
  - Unit: **1** (valor em volts)
  - Change: **true**

- Data:
  - Sensor ID: **0**
  - Value: **10** (10 volts)

O fluxo é bem simples: primeiro o thing é registrado na stack da KNoT Cloud, com nome e ID. Em seguida seus valores de config (ID e definições do sensor) são atualizados como metadados na stack. Logo após, o valor 10 é enviado pelo node **set data** para ser atualizado no thing. Enquanto isso o **on data** escuta se a mensagem chegou ao thing após a validação. Por fim, o thing é desregistrado da stack. A figura 29 mostra o diagrama de fluxo desse caso de uso.

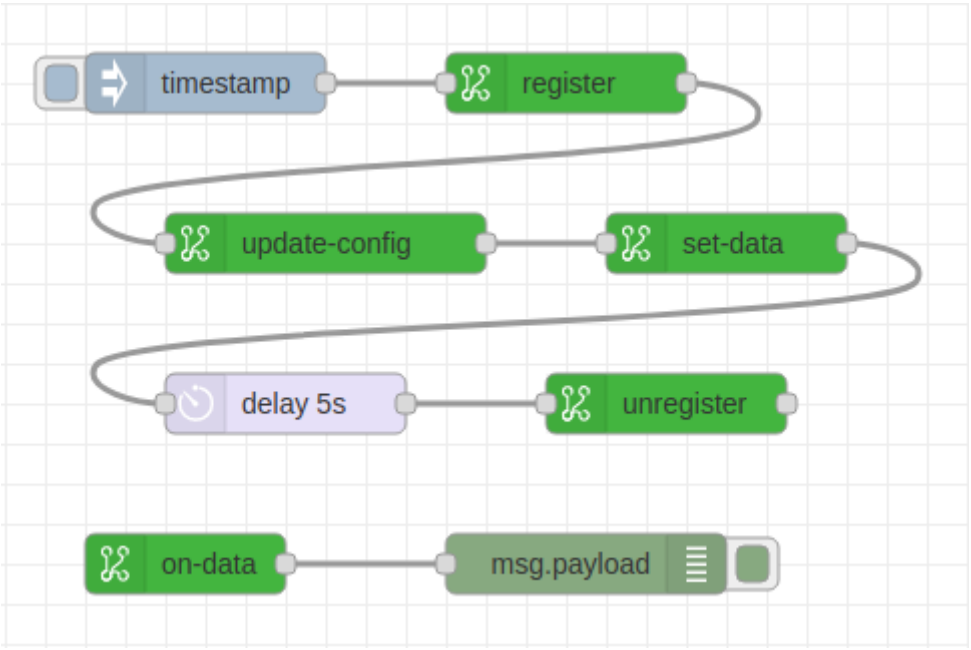

Figura 16. Fluxo exemplo com nodes KNoT SDK

Esse fluxo demonstra bem como é possível utilizar os nodes KNoT SDK em uma aplicação Node-RED. Os nodes poderiam ter sido manipulados de variadas formas, a fim de criar uma interação específica com a stack, como por exemplo enviar um sinal para um thing acender uma lâmpada, ou então realizar a leitura do valor de um sensor de temperatura. Isso amplia o conjunto de ferramentas que possibilitam a criação de aplicações compatíveis com a infraestrutura do KNoT.

5.3 Caso de uso com o node cloud thing

Este segundo exemplo mostra como interagir com um thing KNoT com um maior nível de abstração, sem se preocupar em definir diversos nodes para trocar mensagens com um único thing. Aqui é apresentado como fazer requisições e escutar os eventos vindos de um thing específico, e

são utilizadas apenas as operações relacionadas a dados no thing (publish, set e get). Neste fluxo foram usados os nodes a seguir:

- **inject**: biblioteca padrão do Node-RED
- **debug**: biblioteca padrão do Node-RED
- **cloud thing**: biblioteca Node-RED/KNoT
- **set data**: biblioteca Node-RED/KNoT
- **get data**: biblioteca Node-RED/KNoT

Novamente, os nodes **inject** e **debug** aqui servem apenas para interação com o usuário, o **inject** disparando outros nodes e enviando dados pro **cloud thing**, e o **debug** mostrando a saída do **cloud thing** no console. O **cloud thing** por sua vez é responsável pela criação do thing na stack, por converter os sinais de entrada em operações de publish, e também de escutar requisições de update e request para o thing, enviando respectivos sinais de saída. Para este fluxo, foram usados os seguintes parâmetros:

- Thing ID: **0987654321234567**
- Thing name: **another-thing**
- Config:
  - Sensor ID: **0**
  - Name: **temperature-sensor**
  - Value Type: **1** (tipo inteiro)
  - Type ID: **5** (valor de temperatura)
  - Unit: **1** (valor em graus celsius)
  - Change: **true**
- Data (inject):
  - Sensor ID: **0**
  - Value: **30** (30ºC)
- Data (set data):
  - Sensor ID: **0**
  - Value: **35** (35ºC)

Nesse caso, ao iniciar a execução do fluxo, o **cloud thing** já registra e configura o thing e os devidos eventos. Então é possível fazer publicação de dados injetando valores em sua entrada, ou

ouvir requisições de set ou get em seus sinais de saída. O **set data** e o **get data** realizam as respectivas requisições, a fim de mostrar os eventos na saída do **cloud thing** no console.

Esse exemplo apresenta uma forma de interagir com o thing simulado de uma forma bem mais abstraída, sem a necessidade de utilizar vários nodes para realizar as mesmas operações. O node **cloud thing** poderia inclusive ser substituído por um **thing**, o que possibilitaria a mesma interação com um thing KNoT real, já registrado. Dessa forma, esses nodes agregam bastante valor ao facilitar interações mais complexas com os things, tornando mais simples a criação de regras de mais alto nível na aplicação. A figura 30 mostra a montagem final do fluxo.

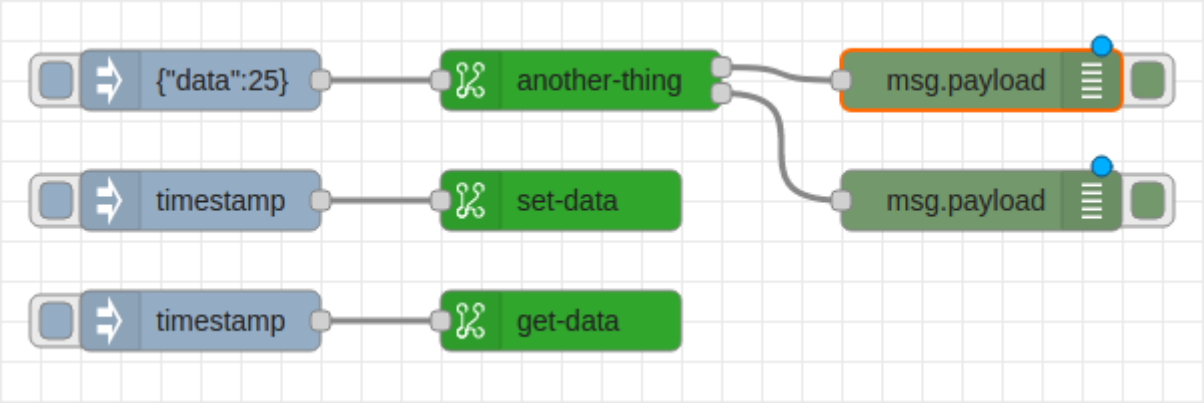

Figura 17. Fluxo exemplo com o node cloud thing

6. Conclusão

A tecnologia de IoT vem crescendo cada vez mais nos últimos anos, e com ela a necessidade de se desenvolver soluções, que possuem um elevado nível de complexidade, de forma cada vez mais simples e acessível.

A meta plataforma KNoT se propõe a reduzir parte dessa complexidade, utilizando uma infraestrutura que engloba desde os things conectados que interagem com o mundo real, até a aplicação final que vai lidar diretamente com o usuário final da solução. Apesar disso, a complexidade de criar um projeto KNoT ainda é elevada para um desenvolvedor sem tanta experiência na área, e pode acabar se tornando uma barreira na difusão da própria plataforma.

Este projeto mostrou que é possível realizar a integração da ferramenta Node-RED, o qual é bastante popular no ecossistema de IoT, com o KNoT. Essa biblioteca de nodes além ser uma extensão do KNoT SDK, que é o componente responsável por viabilizar a criação de aplicações do KNoT, facilita não só a construção de soluções específicas para o KNoT, como permite também a integração com uma enorme variedade de outras tecnologias e protocolos já presentes no Node-RED. Além disso, a execução da stack KNoT Cloud por meio de um node Node-RED mostrou também que possível simplificar essas configurações de nível de abstração mais elevado do KNoT, ajudando ainda mais os usuários menos experientes.

6.1 Dificuldades encontradas

Ao longo do trabalho algumas dificuldades foram aparecendo, servindo não só como oportunidade para pôr em prática as habilidades aprendidas e desenvolvidas ao longo do curso, como também para adquirir novos conhecimentos.

A primeira dificuldade encontrada foi entender como criar um node personalizado para o Node-RED. Foi preciso estudar a ferramenta, entender seu funcionamento e procurar exemplos práticos de nodes já implementados. Apesar do projeto ser de código aberto e possuir uma documentação intuitiva, a má organização de parte do código e a falta de informações específicas na documentação dificultaram um pouco essa etapa inicial.

Outra dificuldade que apareceu durante o desenvolvimento do trabalho foi uma mudança relativamente grande na plataforma KNoT, que deixou de dar suporte à operação **update schema** e criou uma nova, **update config**, para enviar dados referentes ao schema dos sensores/atuadores juntamente com as configurações de eventos, que definem quando um dado deve ser publicado na

KNoT Cloud. Essa mudança teve impacto direto nos nodes já implementados, que tiveram de ser adaptados para a nova interface.

Por fim, outra dificuldade presente no trabalho foi a implementação do node **virtual thing**, pois foi necessário estudar esse novo componente do KNoT, que simula um thing criando uma interface com o protocolo Modbus, e entender seu funcionamento para viabilizar uma interação com o código em Node.js.

6.2 Trabalhos futuros

A medida que o desenvolvimento foi avançando, foi-se percebendo também algumas outras oportunidades de trabalho, que são descritas a seguir:

- **Implementação de melhorias de usabilidade da biblioteca Node-RED/KNoT**: durante a implementação, percebeu-se que alguns nodes poderiam ter uma melhoria relativa a usabilidade na interface gráfica. Como por exemplo na hora de definir o schema dos sensores do thing KNoT, em vez de se passar o valor numérico do protocolo KNoT, a interface já poderia disponibilizar as opções predefinidas para o usuário selecionar os valores mais facilmente.

- **Criação de um node para configuração do KNoT Gateway**: consistiria em adicionar um node (ou mais) para o configuração dos serviços presentes na imagem linux do KNoT Gateway.

- **Criação de um serviço Node-RED para o componente KNoT Gateway**: outro trabalho interessante seria a embarcar uma instância do Node-RED com nodes Node-RED/KNoT, para permitir a interação mais direta com os things/serviços presentes no KNoT Gateway de forma nativa.

- **Criação de node para comunicação entre uma ESP32 e o KNoT VirtualThing**: dada a popularidade módulo embarcado ESP32, esse trabalho consistiria em criar uma ponte entre código embarcado da ESP32 e o protocolo Modbus, utilizado pelo KNoT VirtualThing, e serveria de exemplo para a criação de outros módulos semelhantes.

- **Elaboração de testes automatizados para a biblioteca Node-RED/KNoT**: a criação de testes unitários e de integração são fundamentais para elevar a qualidade de código da biblioteca. Caso mais nodes sejam criados futuramente, a adição desses testes seria fundamental na prevenção à introdução de erros.

- **Criação de aplicações utilizando a biblioteca Node-RED/KNoT**: visto que é possível criar aplicações KNoT utilizando o Node-RED, seria interessante o estudo de mais casos que utilizem essa biblioteca, com a finalidade de coletar mais feedbacks para o desenvolvimento gradual dela.

REFERÊNCIAS

[1] D. Giusto, A. Iera, G. Morabito, L. Atzori, “The Internet of Things”. Springer, 2010.

[2] T. Barros, “O que falta na internet para as coisas?”. Disponível em: https://www.cesar.org.br/index.php/2016/09/14/o-que-falta-na-internet-para-as-coisas/, acessado 18/11/2020.

[3] P. Tarso, R. Alves, F. Ferraz, T. Barros, “Towards a meta platform for the Internet of Things”. Disponível em: https://knot.cesar.org.br/, acessado 18/11/2020.

[4] Node-RED, disponível em: https://nodered.org/, acessado 18/11/2020.

[5] M. Blackstock, R. Lea, “Toward a Distributed Data Flow Platform for the Web of Things (Distributed Node-RED)”, 5th International Workshop on Web of Things. ACM, 2014, pp. 34–39.

[6] W. Voorsluys, J. Broberg, R. Buyya, “Cloud Computing: Principles and Paradigms”. Wiley Online Library, 2011.

[7] M. Aazam, E. N. Huh, “Fog Computing and Smart Gateway Based Communication for Cloud of Things”. The 2nd International Conference on Future Internet of Things and Cloud, 2014.

[8] B. Oliveira, “Introdução a computação em névoa”. Disponível em: https://medium.com/internet-das-coisas/fog-01-introdu%C3%A7%C3%A3o-a-computa%C3%A7%C3%A3o-em-n%C3%A9voa-d4303d5f90b9, acessado 18/11/2020.

[9] A. Banafa, “What is fog computing?”. Disponível em: https://www.ibm.com/blogs/cloud-computing/2014/08/25/fog-computing/, acessado 18/11/2020.

[10] Grupo de pesquisas em IoT do CESAR, “KNoT Introduction”. Disponível em: https://knot-devel.cesar.org.br/doc/general/introduction.html, acessado 18/11/2020.

[11] T. Barros, et al. "A Multi-Radio Gateway Architecture and Implementation for Consumer Electronics." 2019 IEEE International Conference on Consumer Electronics (ICCE).

[12] KNoT Cloud, disponível em: https://github.com/CESARBR/knot-cloud, acessado 18/11/2020.

[13] Grupo de pesquisas em IoT do CESAR, “KNoT Cloud Introduction”. Disponível em: https://knot-devel.cesar.org.br/doc/cloud/cloud-introduction.html, acessado 18/11/2020.

[14] KNoT Cloud SDK, disponível em: https://github.com/CESARBR/knot-cloud-sdk-js, acessado 18/11/2020.

[15] KNoT SDK AMQP, disponível em: https://github.com/CESARBR/knot-cloud-sdk-js-amqp, acessado 18/11/2020.

[16] KNoT SDK Authenticator, disponível em: https://github.com/CESARBR/knot-cloud-sdk-js-authenticator, acessado 18/11/2020.

[17] KNoT BabelTower, disponível em: https://github.com/CESARBR/knot-babeltower, acessado 18/11/2020.

[18] KNoT SDK Storage, disponível em: https://github.com/CESARBR/knot-cloud-sdk-js-storage, acessado 18/11/2020.

[19] KNoT Storage, disponível em: https://github.com/CESARBR/knot-cloud-storage, acessado 18/11/2020.

[20] KNoT VirtualThing, disponível em: https://github.com/CESARBR/knot-virtualthing, acessado 18/11/2020.

[21] T. Szydlo, R. Brzoza-Woch, J. Sendorek, M. Windak, C. Gniady, “Flow-Based Programming for IoT Leveraging Fog Computing”, 2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE). vol. 00, pp. 74–79, junho de 2017.

[22] KNoT WebUI, disponível em: https://github.com/CESARBR/knot-gateway-webui, acessado 18/11/2020.

[23] Node-RED Arduino, disponível em: https://github.com/node-red/node-red-nodes/tree/master/hardware/Arduino, acessado 18/11/2020.

[24] D. Madeira, “Primeiros passos com o Node-RED e Arduino UNO”. Disponível em: https://www.filipeflop.com/blog/primeiros-passos-node-red-arduino-uno/, acessado 18/11/2020.

APÊNDICES

Apêndice A – Imagens dos nodes por categorias

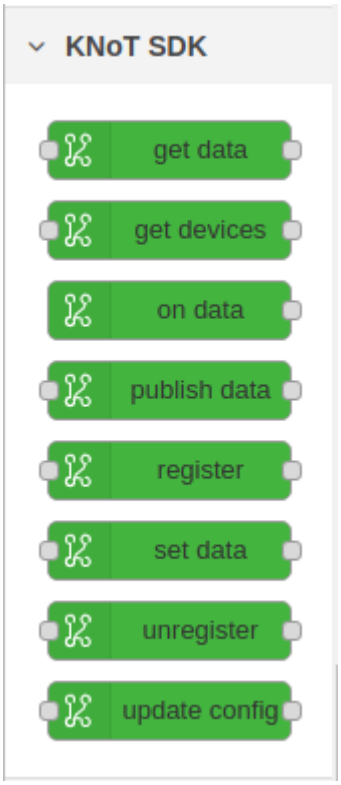

Figura A-1. Lista de nodes KNoT SDK

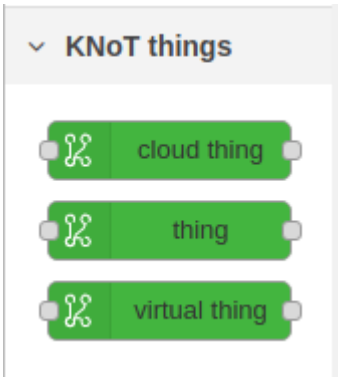

Figura A-2. Lista de nodes KNoT Things

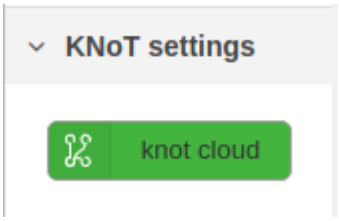

Figura A-3. Lista de nodes KNoT Settings

Apêndice B – Imagens das telas de configuração dos nodes

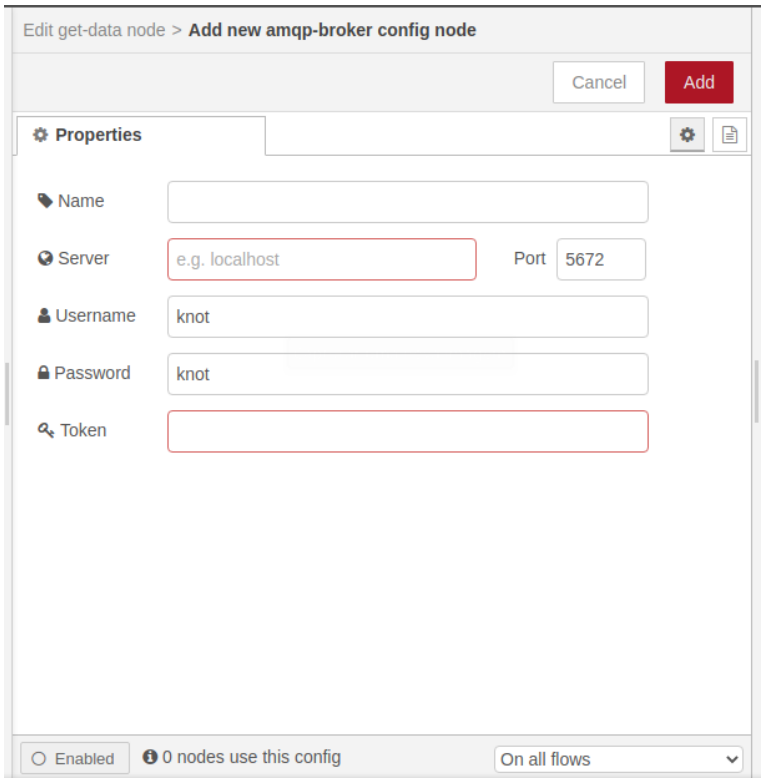

Figura B-1. Tela de configuração do node amqp broker

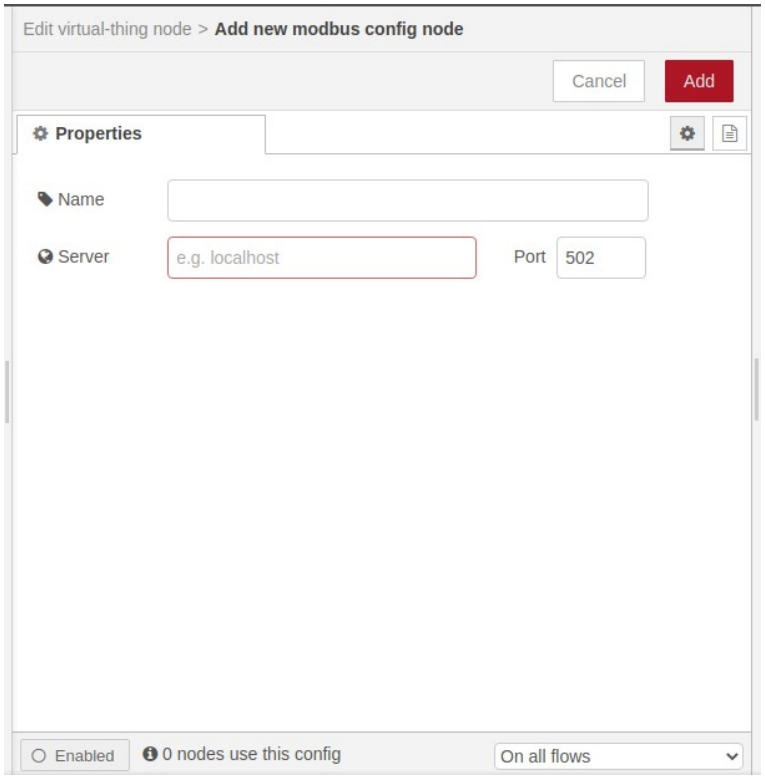

Figura B-2. Tela de configuração do node modbus

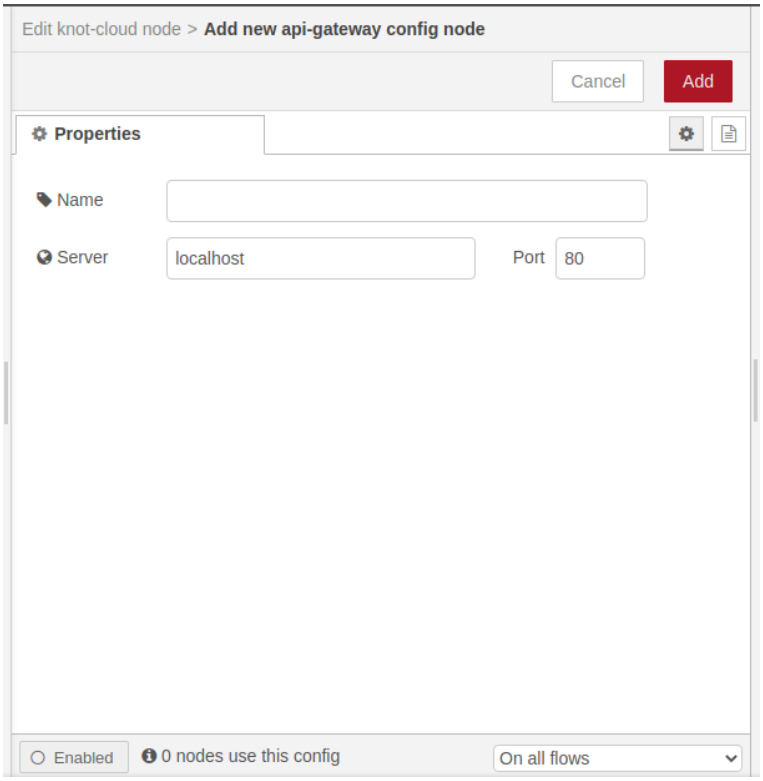

Figura B-3. Tela de configuração do node api gateway

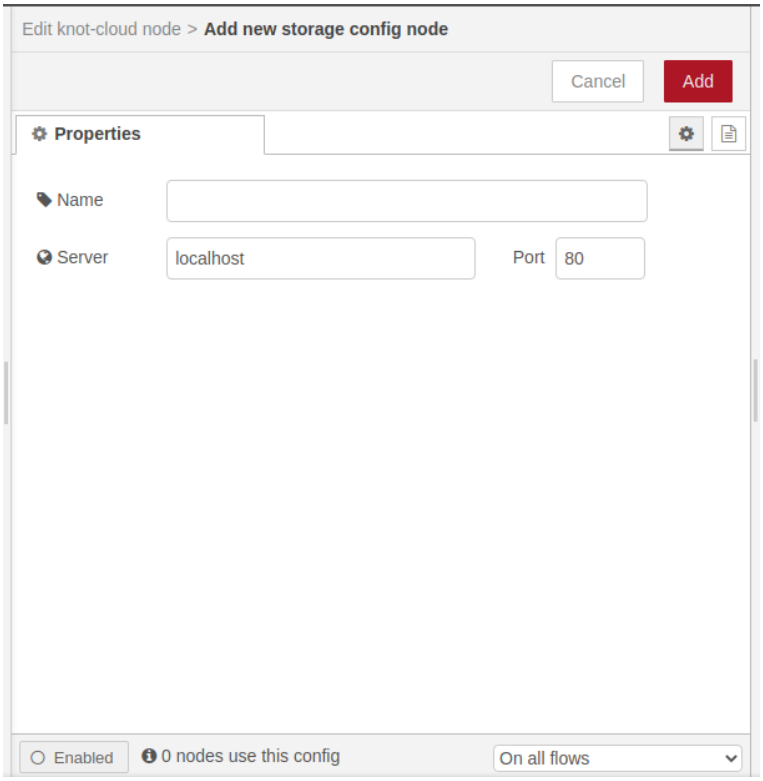

Figura B-4. Tela de configuração do node storage

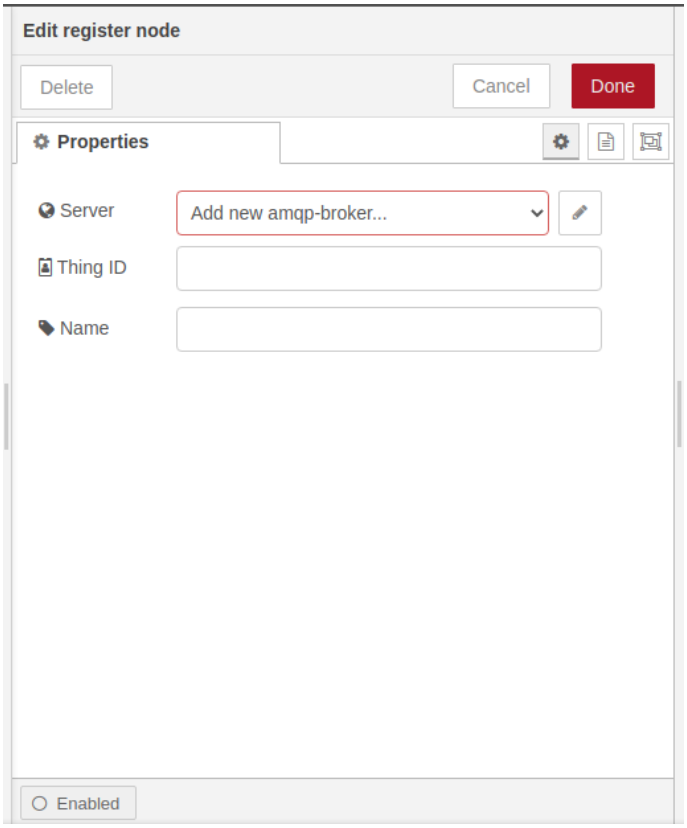

Figura B-5. Tela de configuração do node register

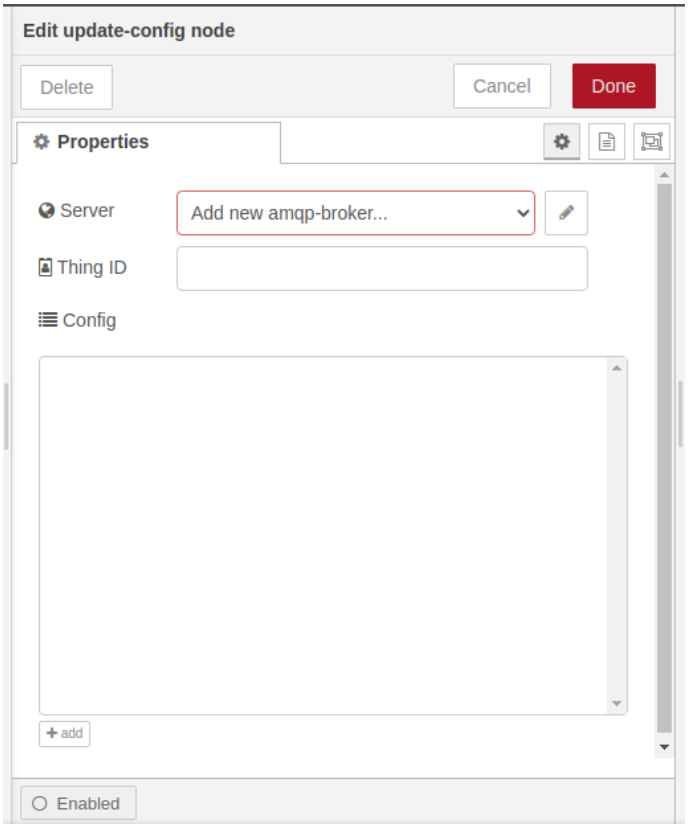

Figura B-6. Tela de configuração do node update config

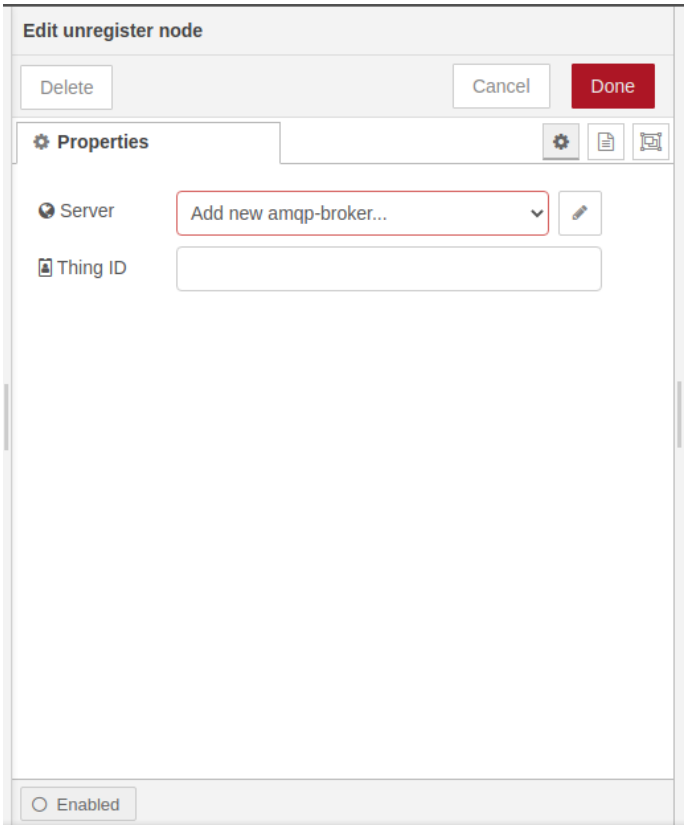

Figura B-7. Tela de configuração do node unregister

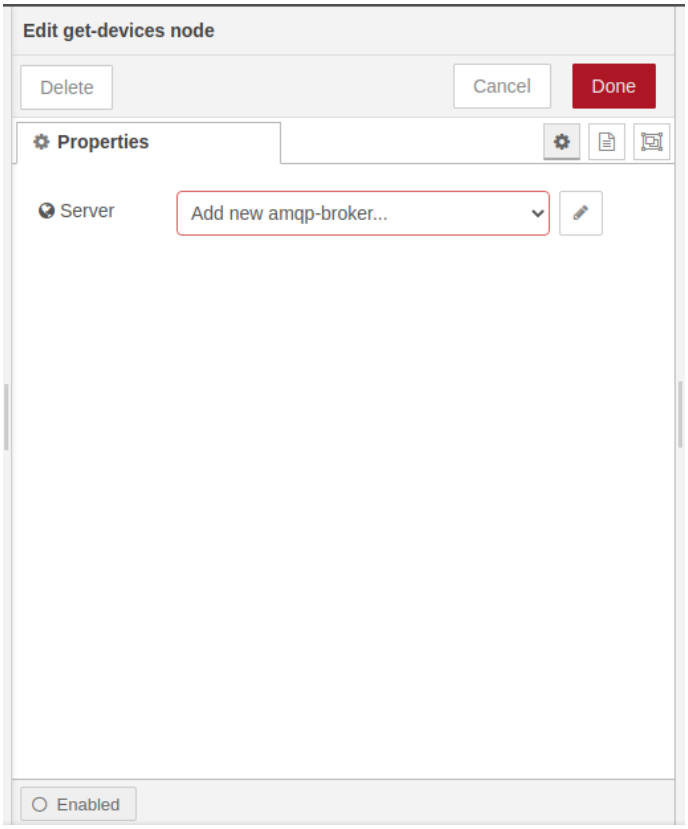

Figura B-8. Tela de configuração do node get devices

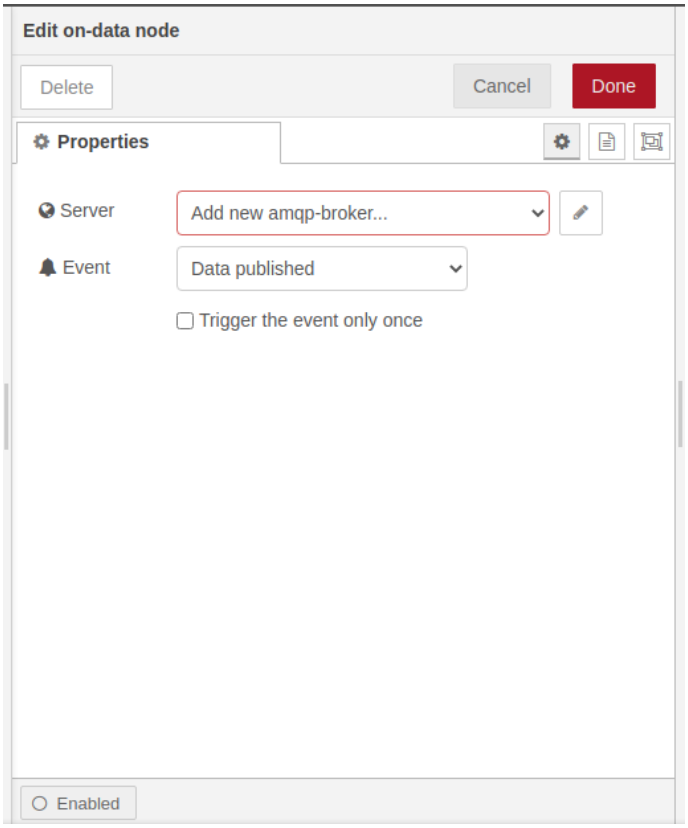

Figura B-9. Tela de configuração do node on data

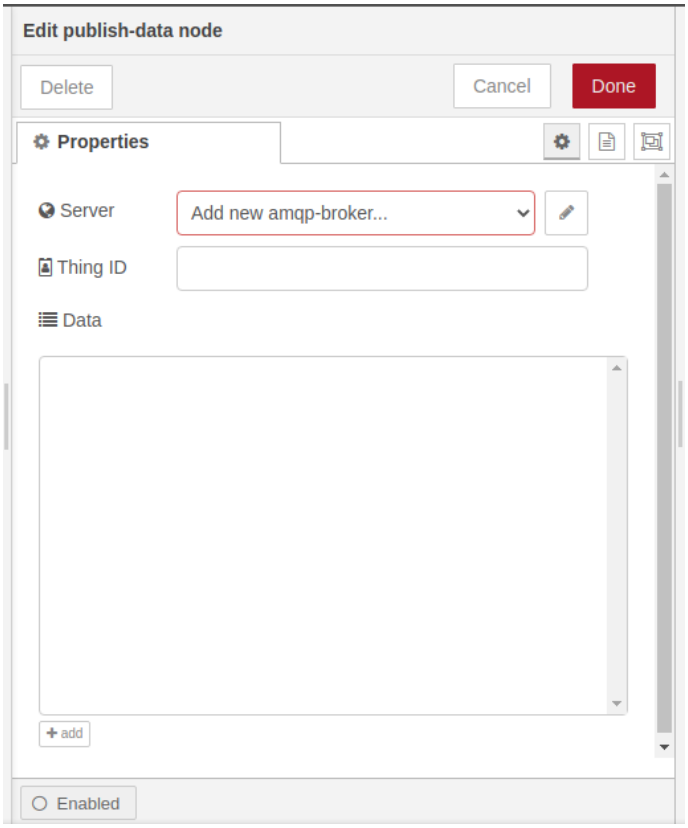

Figura B-10. Tela de configuração do node publish data

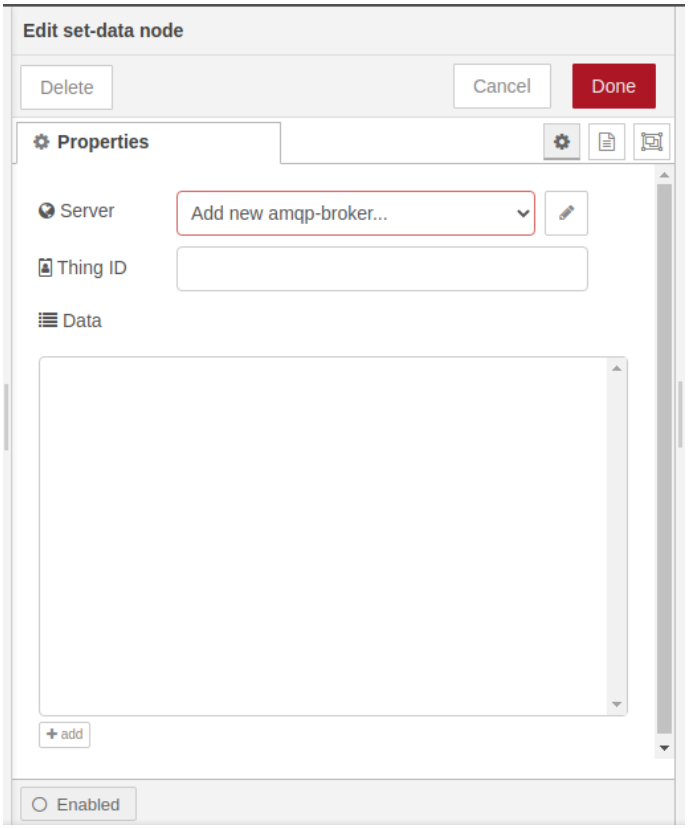

Figura B-11. Tela de configuração do node set data

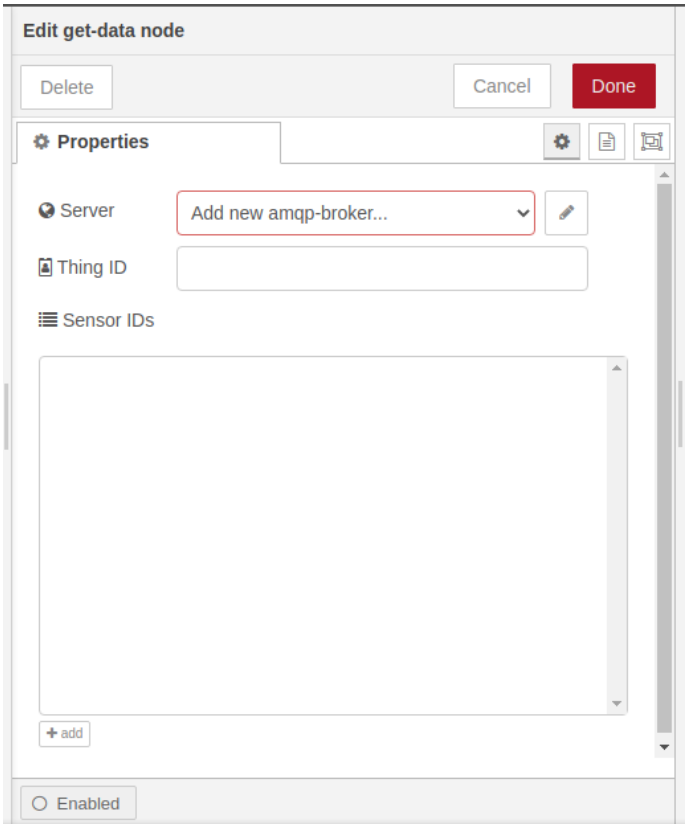

Figura B-12. Tela de configuração do node get data

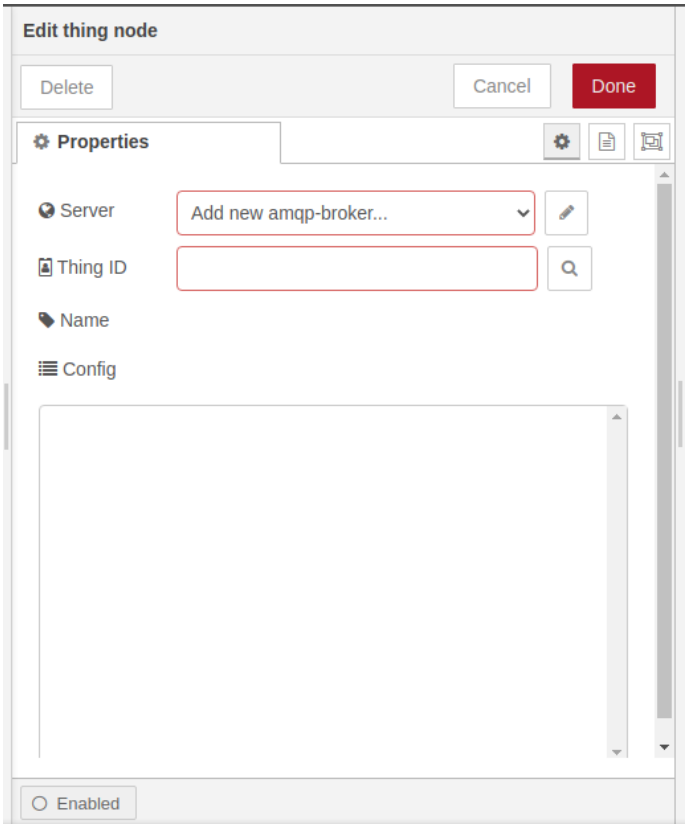

Figura B-13. Tela de configuração do node thing

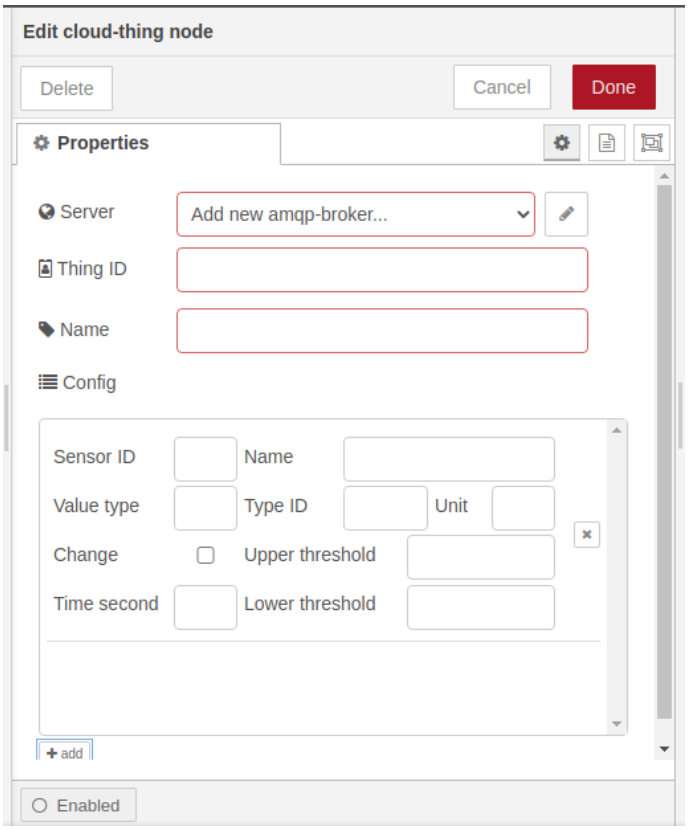

Figura B-14. Tela de configuração do node cloud thing

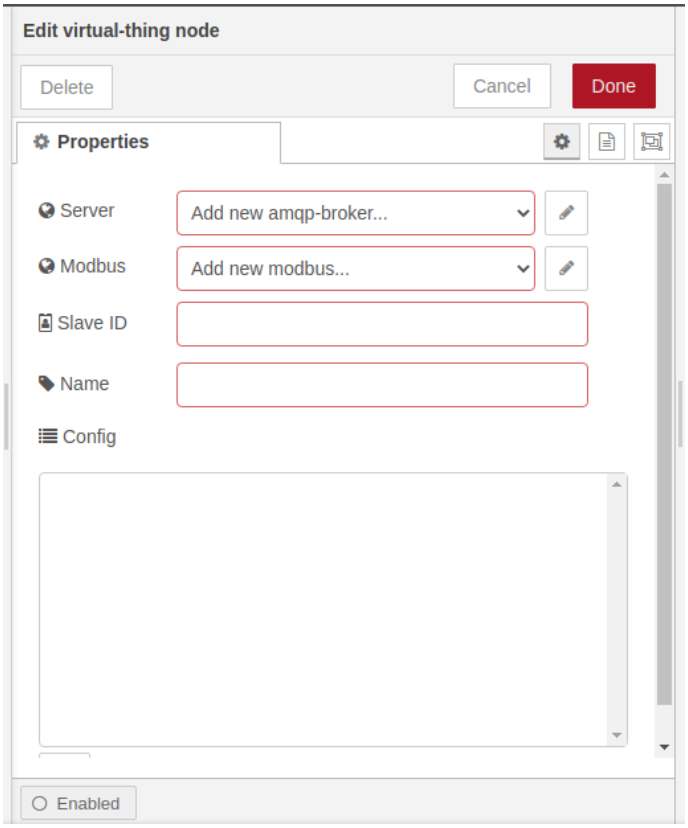

Figura B-15. Tela de configuração do node virtual thing

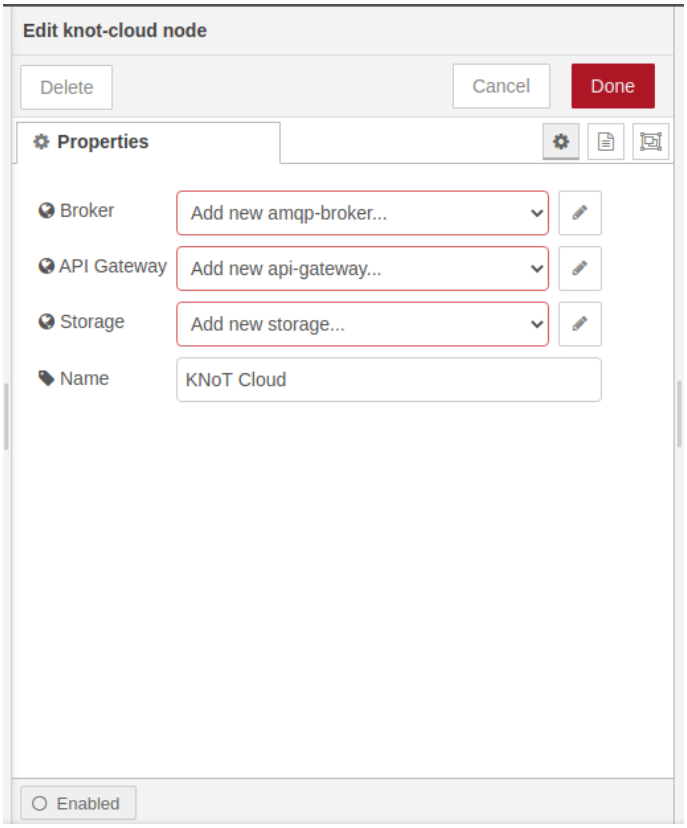

Figura B-16. Tela de configuração do node knot cloud