Uso de um testbed de rede de sensores para teste e avaliação de aplicações de cidades inteligentes

Rafael R. Silva, Fábio M. Costa (Orientador), Bruno Silvestre (Coorientador)

[1]Instituto de Informática
Universidade Federal de Goiás (UFG)

silva_rafael@discente.ufg.br, fmc@inf.ufg.br, brunoos@inf.ufg.br

Resumo

As plataformas de Cidades Inteligentes surgiram como base para o desenvolvimento e integração de aplicações distribuídas para as cidades, fornecendo serviços comuns e agilizando a construção de soluções. Contudo, um ponto crítico durante o desenvolvimento de plataformas de cidades inteligentes é a integração com sensores. Devido à heterogeneidade encontrada neste meio, a tarefa de realizar a comunicação entre dispositivos de diversos protocolos se torna uma tarefa complexa. Tal fato prejudica a implementação de *testbeds* de redes de sensores, extremamente importantes para a validação de simulações em ambientes realísticos. Desenvolvido no contexto da plataforma InterSCity, este trabalho propõe um modelo de um *gateway* responsável pelo interfaceamento de diversos tipos de sensores com a plataforma, possibilitando também a integração da plataforma com um *testbed* de rede de sensores. Portanto, o trabalho propõe uma infraestrutura adequada para testes e avaliação de aplicações de cidades inteligentes.

1. Apresentação

Tendo em vista os avanços tecnológicos obtidos nas últimas décadas, diversas tecnologias foram desenvolvidas no contexto da Internet das Coisas, como sensores e microcontroladores. Aliado ao desenvolvimento tecnológico, diversas aplicações foram criadas para solucionar problemas de uma cidade moderna. Inicialmente as soluções focavam em usos específicos, com infraestruturas próprias. Toda aplicação era responsável pela coleta, tratamento e disponibilização dos dados ao usuário através de uma aplicação.

Atualmente, devido à grande demanda de aplicações em uma cidade moderna, o cenário de desenvolvimento para aplicações em cidades inteligentes passa por uma modificação. O desenvolvimento de plataformas de cidades inteligentes, que centralizam o acesso aos dados dos sensores e facilitam o processo de criação de novas aplicações, aparece como uma alternativa para solucionar um dos principais problemas de aplicações em cidades inteligentes, que é a escalabilidade.

Outra estrutura utilizada em conjunto com as plataformas são ambientes de testes, para aproximar os resultados obtidos em testes aos resultados em cenários realísticos, além de automatizar os processos de testagem. Esses ambientes são chamados de *testbeds*.

Neste trabalho propomos um *gateway* para acesso à dispositivos de IoT, denominado *IoTGateway*, para a plataforma de Cidade Inteligente InterSCity [Esposte et al. 2017], o qual facilita o envio de dados de sensores à plataforma. Nas próximas subseções, apresentamos os conceitos de plataformas de Cidades Inteligentes

(Seção 1.1) em geral e da plataforma InterSCity em particular (Seção 1.2), bem como o conceito de *testbed* de redes de sensores (Seção 1.3).

1.1. Plataformas de Cidades Inteligentes

As grandes cidades enfrentam diversos problemas, como por exemplo gerenciamento de resíduos sólidos, desperdício de energia elétrica, congestionamentos de trânsito, poluição do ar e segurança pública. Porém, muitos problemas podem ser solucionados com o auxílio da tecnologia. Para isso, pesquisas na área de cidades inteligentes buscam proporcionar ferramentas que utilizam a internet das coisas. Durante os últimos anos, diversas aplicações surgiram por meio de pesquisas na área acadêmica e iniciativas do mercado para solucionar problemas do dia a dia dos habitantes de uma cidade, como Google Maps[1] para geolocalização e rotas, e o Olho na Bomba[2], que foi um aplicativo desenvolvido pelo Instituto de Informática da UFG para permitir que o consumidor realize a consulta dos preços de postos de combustível de uma região pelo celular.

Embora as aplicações citadas abordem problemas de cidades, elas são desenvolvidas apenas para uma finalidade específica, utilizando recursos também específicos. Para uma Cidade Inteligente seria interessante que essas aplicações pudessem interoperar entre si, compartilhando dados e a mesma infraestrutura de recursos, assim como fornecendo soluções mais integradas. Por outro lado, com uma gama maior de aplicações interagindo entre si, problemas relacionados à escalabilidade dessas aplicações surgirão inevitavelmente. Tratar estas questões é importante, pois Cidades Inteligentes costumam ser retratadas como diversos instrumentos conectados através de várias redes distintas fornecendo dados contínuos sobre a cidade, e que uma cidade só pode ser considerada como "inteligente" se existirem funções capazes de integrar e sintetizar os dados para algum propósito como melhorar a eficiência em algum processo, a sustentabilidade ou até mesmo a qualidade de vida na cidade [Batty et al. 2012].

A Internet das Coisas (IoT) possibilita conectar os objetos da cidade com as aplicações, provendo dados que servem de entrada. Além disso, por meio de atuadores da IoT, essas aplicações podem interagir de volta com a cidade (por exemplo, semáforos inteligentes) e proporcionar transformações na vida do cidadão.

No entanto, a interligação de Cidades Inteligentes com a IoT não é uma tarefa fácil. As tecnologias de IoT podem ser desde *smartphones* até minúsculos controladores, variando em capacidade computacional, armazenamento e tecnologia de comunicação. Um sistema de Cidades Inteligentes deve ser versátil para poder acomodar a interligação dessas diversidades de forma transparente para que a construção das aplicações não seja afetada por esses detalhes. Diversas plataformas de cidades inteligentes foram propostas com base nesse princípio, dentre elas: City of Things, dojot, FIWARE e InterSCity, que são descritas a seguir.

City of Things (CoT) é uma plataforma para cidades inteligentes implementada na cidade de Antuérpia (Bélgica) e que abrange várias tecnologias de comunicação sem fio [Latre et al. 2016]. CoT também é um *testbed* que permite realizar e validar testes em cenários realísticos. Sua infraestrutura possui alguns componentes principais, como pode ser visto na Figura 1. Ela possui sensores conectados à plataforma, uma rede privada

[1]https://www.google.com/maps
[2]https://ww2.inf.ufg.br/node/1349

baseada em LoraWAN e *gateways* que dão suporte a diversas tecnologias de comunicação sem fio. As principais tecnologias utilizadas pela CoT são: DYAMAND, para coleta de dados de sensores; Tengu, para processamento e armazenamento dos dados; e LimeDS, para acesso aos dados.

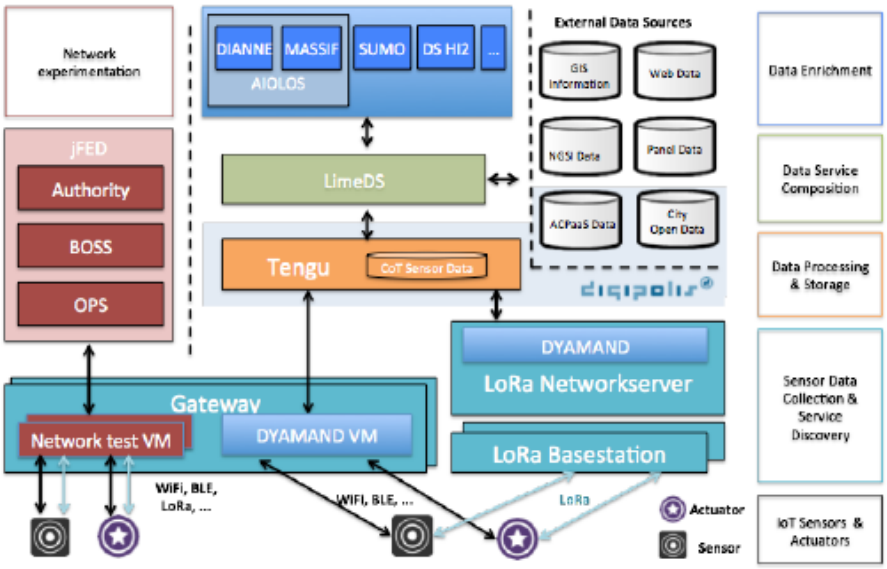

Figura 1. Arquitetura da Plataforma City Of Things [Latre et al. 2016]

Focando no processo de coleta de dados, o DYAMAND atua como uma camada de *middleware* entre a plataforma e os sensores [Nelis et al. 2012]. Ele lida com a coleta dos dados de sensores distintos e tradução desses dados para que seja possível seu acesso dentro da CoT de forma heterogênea. A plataforma define um padrão para os dados e, através do DYAMAND, toda mensagem recebida é transformada para este padrão.

A plataforma dojot [dojot 2020] é uma plataforma de código aberto baseada em microsserviços. Ela foi desenvolvida pelo CPqD, com foco nas áreas de segurança pública, mobilidade urbana e saúde, com o objetivo de prover a base para a construção de um ecossistema multidisciplinar nessas áreas. A plataforma utiliza diversos componentes, como observado na Figura 2, que ilustra sua arquitetura.

O componente utilizado para a comunicação com os sensores na plataforma dojot é o IoT Agent. Existem diversas implementações desse componente, dependendo do protocolo utilizado na comunicação. O IoT Agent é o componente responsável pela coleta de dados dos sensores, seguido por sua padronização, para que seja possível o acesso uniforme a esses dados dentro da plataforma [Sampaio 2018].

A FIWARE [Fazio et al. 2015] é uma plataforma configurável composta de componentes desenvolvidos por uma comunidade aberta e independente. Os componentes *Generic Enablers* (GEs) possibilitam a implementação adaptativa da plataforma. Cada aplicação pode construir sua plataforma de acordo com os GEs que julgar necessários para seu uso. A Figura 3 mostra as principais categorias de GEs.

Observando os GEs responsáveis pela comunicação com sensores, notamos que na FIWARE, semelhante à dojot, temos os compontentes *IoT Agent*, que são implementados

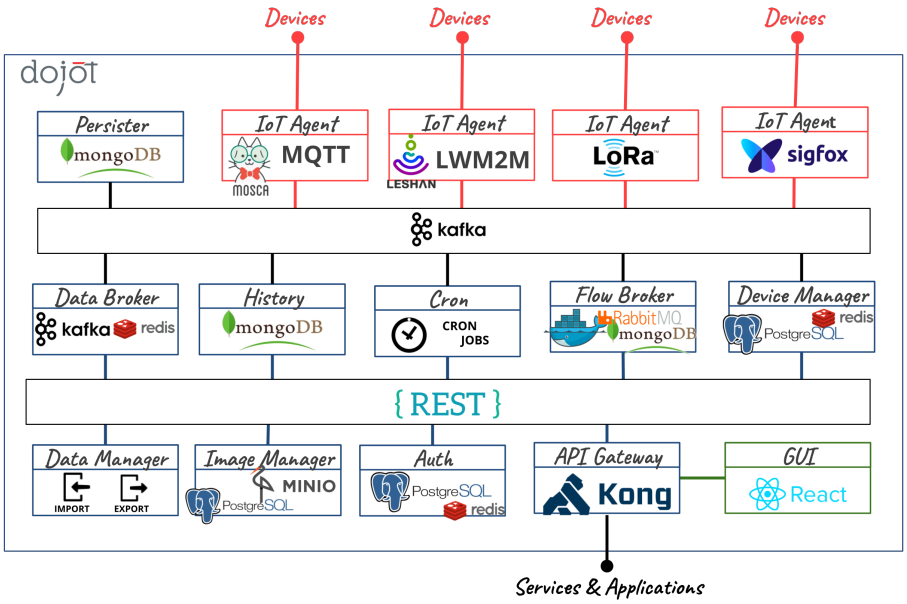

Figura 2. Arquitetura da dojot [dojot 2020]

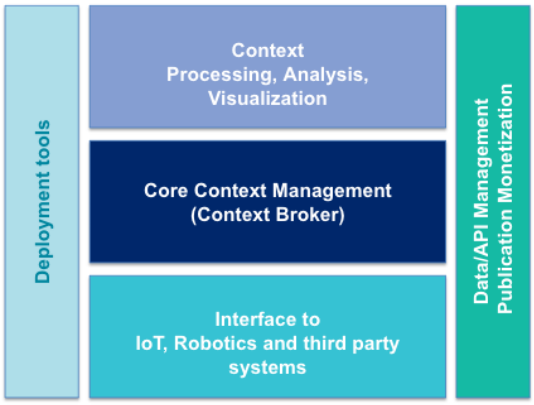

Figura 3. Áreas de desenvolvimento dos Generic Enablers da FIWARE[FIWARE 2020].

de acordo com o protocolo utilizado nos sensores. Na FIWARE os IoT Agents traduzem dados dos respectivos protocolos dos sensores para o protocolo utilizado internamente entre os GEs, que é o NGSI.

1.2. A Plataforma InterSCity

A plataforma escolhida para realizar a integração com um ambiente de testes foi a plataforma InterSCity [Del Esposte et al. 2019]. Ela é uma plataforma de código aberto para aplicações em cidades inteligentes baseada em microsserviços e que possui uma arquitetura focada em escalabilidade. A plataforma foi desenvolvida pelo INCT da Internet do Futuro para Cidades Inteligentes, sediado no Instituto de Matemática e Estatística da Universidade de São Paulo, e do qual participam outras universidades brasileiras, dentre as quais a UFG. O objetivo principal da plataforma é fornecer serviços e APIs para auxiliar no desenvolvimento de novas aplicações para cidades inteligentes.

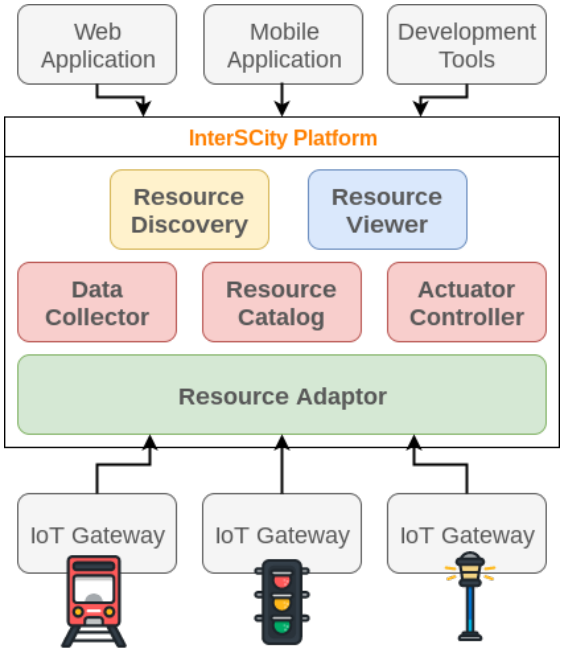

Figura 4. Arquitetura da Plataforma InterSCity [Del Esposte et al. 2019]

Conforme a arquitetura exposta na Figura 4, a plataforma recebe dados enviados pelos recursos da infraestrutura da cidade (sensores e atuadores, por exemplo) por meio de *IoT Gateways*, os quais por sua vez os enviam ao microsserviço *Resource Adaptor*. Esse microsserviço é responsável por realizar a comunicação entre os *IoT Gateways* e os demais microsserviços, bem como cadastrar recursos (sensores e atuadores) e interagir com os recursos por meio de consultas e do envio de comandos.

O processo de cadastro e envio de dados dos recursos à InterSCity é realizado através de requisições REST[3], contendo informações sobre o sensor (como capacidades, descrição e tipos de dados que serão enviados). Após realizar o cadastro, a plataforma gera um UUID[4] responsável pela identificação do recurso. A partir daí, o acesso e envio de dados é realizado por meio de requisições REST, endereçando os dados ao recurso usando seu UUID.

O *IoTGateway* é a estrutura responsável por enviar dados coletados à plataforma InterSCity. A forma com que esses dados são enviados ao *IoTGateway* é abstraída nos níveis superiores.

Existem diversos tipos de sensores e protocolos usados por dispositivos de IoT. Consequentemente, há uma grande heterogeneidade entre os sensores, tanto no nível de comunicação quanto no nível dos dados [Arasteh et al. 2016]. Devido à heterogeneidade, é necessário que o *IoTGateway* transforme os dados recebidos dos sensores em requisições REST de acordo com o formato esperado para que seja possível a integração com a plataforma InterSCity.

[3] *Representational State Transfer*
[4] *Universally Unique IDentifier*

1.3. *Testbed* para Redes de Sensores

Devido ao desenvolvimento de ambientes cada vez mais complexos para aplicações em Cidades Inteligentes, onde são utilizados sensores, cria-se a necessidade do desenvolvimento de um ambiente para realização de testes.

Um *testbed* para uma rede de sensores pode automatizar o processo de conexão dos dispositivos na rede, além de fornecer um meio para que seja possível a configuração dos sensores remotamente [Gao et al. 2020].

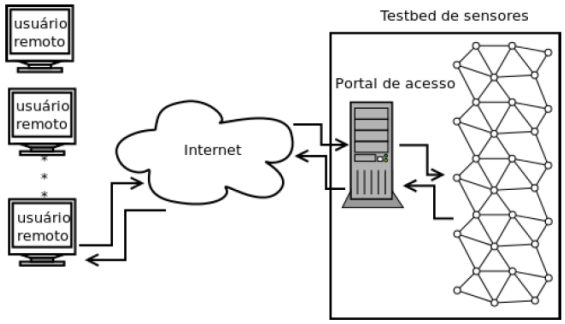

Figura 5. Visão geral do testbed CeuNaTerra [Rossetto et al. 2015]

Para este trabalho optamos pela utilização do *testbed* CeuNaTerra, que é um *testbed* de acesso remoto para educação e pesquisa em aplicações híbridas distribuídas [Rossetto et al. 2015]. Ele se conecta a uma rede de sensores, configura e disponibiliza os dados dessa rede remotamente, conforme mostrado na Figura 5.

O *testbed* possui diversas atuações, conforme mostrado na Figura 6, onde é possível identificar os elementos principais que o compõem: rede de sensores, controle geral e interface de operação. Nos atentaremos apenas ao último deles, que é utilizado para a realização dos serviços fornecidas pelo *testbed*.

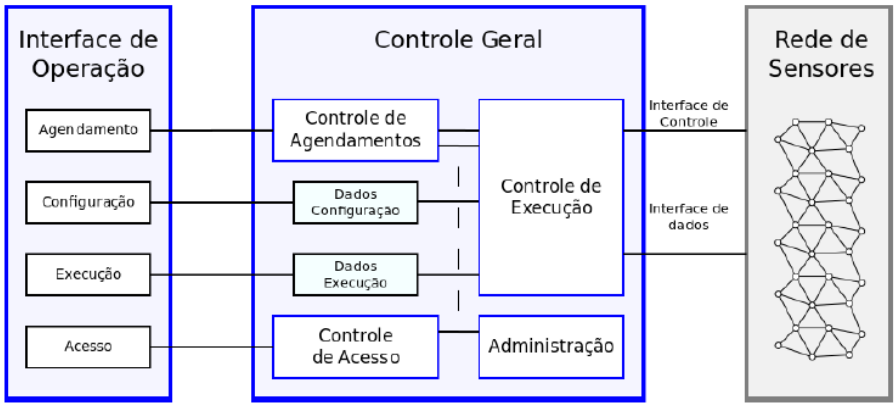

Figura 6. Arquitetura funcional do *testbed* CeuNaTerra [Rossetto et al. 2015]

Para a execução de testes no CeuNaTerra, inicialmente é utilizada a interface de configuração, que permite a ativação e desativação de nós, bem como o envio de executáveis. Posteriormente, a interface de execução permite ao usuário executar e monitorar os sensores, além disso, o *testbed* possui a interface de agendamento que possibilita

especificar um horário para a realização dos testes, o que pode automatizar o processo de execução. Por fim, temos a interface de acesso, que controla o acesso aos recursos do *testbed*.

2. Metodologia

Inicialmente realizamos uma pesquisa exploratória com o objetivo de entender os sensores, protocolos e microcontroladores que possibilitam o envio de dados de sensores a uma plataforma de cidades inteligentes.

As tecnologias utilizadas nos testes iniciais de envio de dados foram TelosB, Microbit, Raspberry Pi e Arduino, com os protocolos MQTT[5] e CoAP[6]. Como são ferramentas amplamente conhecidas no mundo IoT, existem muitas aplicações e exemplos de implementações dessas tecnologias disponíveis na Internet, como tutoriais para a implementação de protocolos em dispositivos de sensores [Cabé 2017, Lloyd 2015]. Utilizamos o Contiki OS [7] para o envio de dados via CoAP. Para o envio via MQTT foi necessária a utilização de um *broker*, como o Eclipse Mosquitto [8] e o CloudMQTT [9].

Posteriormente realizamos um estudo das APIs da InterSCity e dos padrões de formatação de dados da plataforma, assim como testes simples com a criação de recursos dentro da plataforma e envio de dados a esses recursos. Com essa estrutura em mente, realizamos a criação de um *IoTGateway* que facilita a integração de diversos sensores à plataforma (Seção 3). Construímos uma implementação do *IoTGateway*[10] em Python 2.7, na qual realizamos duas implementações do Protocol Client, utilizando o Eclipse Paho[11] e CoAPthon[12], respectivamente, para a integração com os protocolos MQTT e CoAP. Para validar a funcionalidade do *IoTGateway* como meio para conexão entre *tesbeds* de redes de sensores à plataforma InterSCity, realizamos a integração com o *testbed* CéuNaTerra (Seção 3.2).

3. Resultados e discussão

Nesta seção, são descritos: a arquitetura do *IoTGateway* proposto para plataformas de cidades inteligentes; a integração dessa arquitetura com o testbed CéuNaTerra; e a comparação com trabalhos relacionados.

3.1. Arquitetura do *IoTGateway*

O *IoTGateway* é o componente que possibilita o envio de dados de um sensor à plataforma. O componente foi desenvolvido no contexto da plataforma InterSCity, com o objetivo de enviar dados de redes de sensores à plataforma, através de um protocolo de comunicação que segue o estilo REST, conforme descrito nas próximas subseções.

O *IoTGateway* possui como funções o cadastro de recursos (sensores e atuadores) na plataforma, assim como o envio de dados obtidos desses recursos à plataforma, por

[5]MQ Telemetry Transport
[6]Constrained Application Protocol
[7]http://www.contiki-os.org
[8]https://mosquitto.org
[9]https://www.cloudmqtt.com
[10]Implementação disponível em https://github.com/rafaelsilvabr/IoTGateway
[11]https://www.eclipse.org/paho/
[12]https://pypi.org/project/CoAPthon/

meio da conversão dos protocolos de comunicação nativos dos recursos para o protocolo utilizado pela plataforma. Sua arquitetura permite que ele seja facilmente adaptado para uso com novos protocolos utilizados pelos sensores.

Figura 7. Visão simplificada do IoTGateway.

3.1.1. Componentes do IoTGateway

A arquitetura proposta possui três componentes internos principais: *Protocol Client*, *ID Mapper* e *Sender*. O componente *Protocol Client* é responsável pela comunicação com os sensores e implementa os métodos específicos de cada protocolo. O componente *ID Mapper* é responsável por intermediar as interações de registro de recursos e realizar o mapeamento entre os identificadores locais utilizados pelos recursos e os identificadores utilizados pela plataforma. O componente *Sender*, por sua vez, é responsável por preparar e enviar dados dos recursos (e.g., sensores) previamente registrados à plataforma.

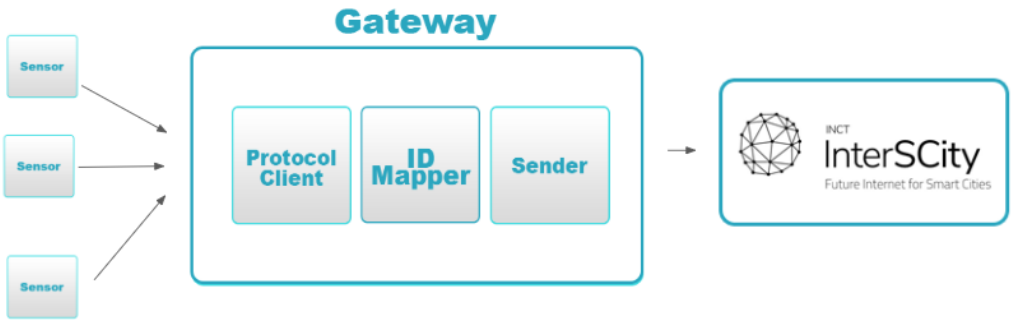

Figura 8. Componentes do IoTGateway.

Por ser o componente que realiza a comunicação com os sensores, o *Protocol Client* pode possuir diversas implementações, de acordo com os protocolos utilizados para comunicação com os sensores. Até o momento, desenvolvemos duas implementações, para os protocolos CoAP e MQTT, respectivamente.

Geralmente são definidos padrões de identificadores para os recursos subjacentes utilizados por uma plataforma. Entretanto, devido ao fato de alguns sensores possuírem capacidade de memória extremamente limitada, bem como ao fato de ser comum o uso de protocolos proprietários, onde a API interna não é fornecida ao desenvolvedor, torna-se impossível a renomeação do dispositivo, ou mesmo a adoção, no nível dos dos dispositivos físicos, do mesmo padrão de identificadores utilizado pela plataforma. Por esse motivo, o *IoTGateway* precisa mapear os identificadores locais dos recursos, aqui chamados de *localIDs*, para os identificadores utilizados na plataforma. No caso da InterSCity,

um padrão de UUID[13] é utilizado para identificar os recursos utilizados pela plataforma.

O *ID Mapper* é o componente do *IoTGateway* responsável por esse mapeamento entre os identificadores dos recursos. Como citado na Seção 1.2, a plataforma InterSCity possui um sistema de cadastro de recursos. Esse componente, portanto, é responsável pelo cadastro e consulta na plataforma. Além disso, ele possui uma base de dados local, onde guarda dados dos identificadores dos recursos, além da UUID fornecida pela InterSCity no momento de cadastro do recurso, o que simplifica a implementação dos sensores já que o *ID Mapper* se ocupa da tarefa de mapeamento dos identificadores.

Finalmente, o *Sender* é o componente responsável pelo envio de dados dos recursos à plataforma. Posteriormente ao registro do recurso na plataforma realizado pelo *ID Mapper*, os dados dos dispositivos de sensores são enviados, na forma de mensagens REST, para o microsserviço *Resourcer Adaptor* da plataforma InterSCity.

3.1.2. Funcionamento dos Componentes

Devido à heterogeneidade das tecnologias de sensores, em uma infraestrutura de cidades inteligentes é possível encontrar sensores com suporte abrangente para envio e recebimento de diversos tipos de mensagens, assim como sensores mais simples e que possuem severas limitações, sendo capazes apenas de enviar mensagens de dados simples e sem suporte para o recebimento de mensagens. Além disso, é possível encontrar sensores com software proprietário, cuja API interna não é fornecida para os desenvolvedores de forma que possam ser adaptados para o protocolo de interação com a plataforma. Desta forma, o *IoTGateway* precisa fornecer suporte para a integração transparente de diferentes tipos sensores, desde os mais simples aos mais sofisticados, à plataforma InterSCity. Um exemplo disto é o registro de recursos junto à plataforma. Embora todo recurso utilizado pela plataforma InterSCity deva ser registrado no microsserviço *Resource Adaptor*, nem todo sensor possui essa capacidade. O *IoTGateway*, portanto, define dois padrões para interação com sensores: com registro e sem registro. Estes dois padrões são descritos a seguir.

No protocolo de envio com registro, o recurso envia dois tipos de mensagens, mensagens de registro e mensagens de publicação. De acordo com a Figura 9, que demonstra o envio de um novo dado à plataforma, após uma mensagem de registro ser enviada ao *IoTGateway*, contendo as informações de registro do recurso, a mensagem é recebida pelo *Protocol Client*, que identifica o tipo de mensagem (registro) e a encaminha ao *ID Mapper*. Ao receber a mensagem, o *ID Mapper* verifica na base de dados local o cadastro do recurso pelo *LocalID*. Como se trata de um dado novo, nada será encontrado. O *ID Mapper* então realiza o cadastro do sensor como um recurso novo na plataforma, mapeia o *LocalID* do recurso para o identificador fornecido pela plataforma (no caso da InterSCity, um UUID). Posteriormente, o *ID Mapper* retorna ao *Protocol Client* um sinal de que o cadastro foi realizado e o *Protocol Client* retorna o mesmo sinal ao sensor. A partir deste ponto, o recurso pode passar a enviar dados e medições.

Após a fase de registro, o sensor passa para a fase de publicação, na qual envia mensagens de dados para o *IoTGateway*. Essas mensagens são recebidas pelo *Protocol*

[13] *Universally Unique IDentifier*

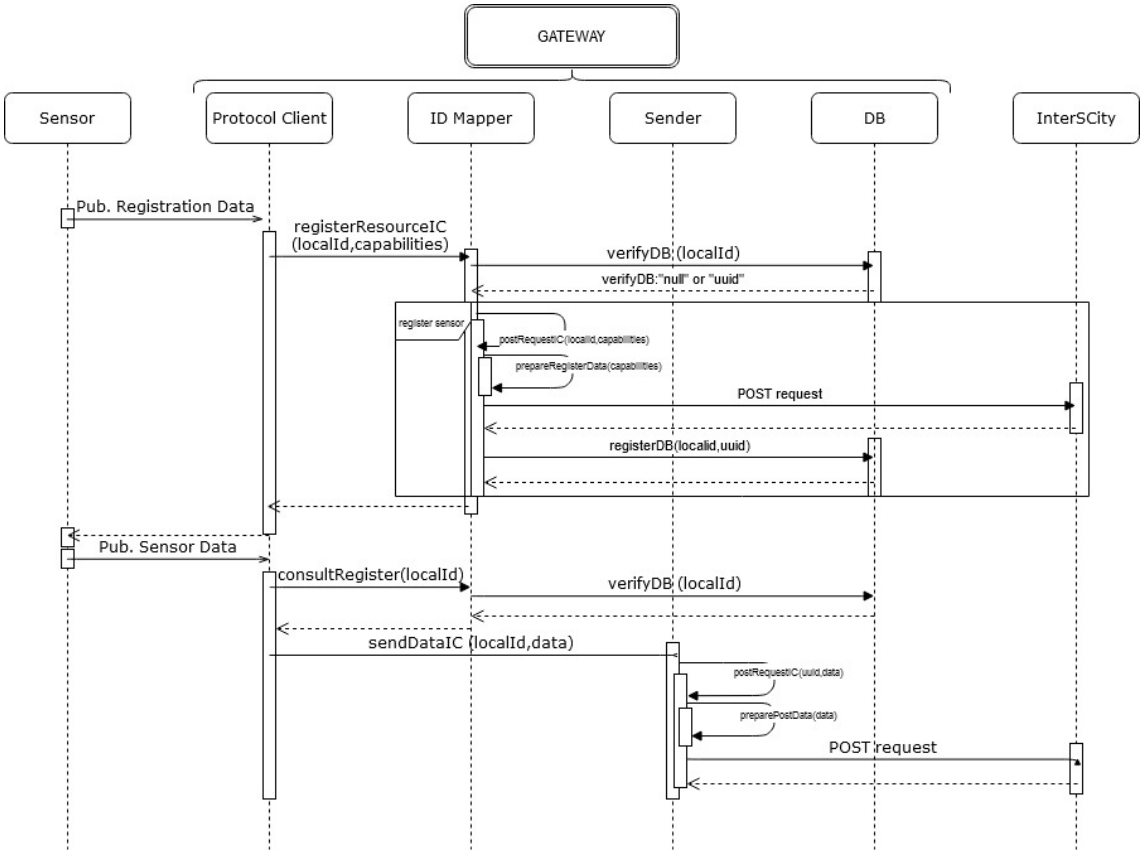

Figura 9. Protocolo de envios com registro do IoT Gateway.

Client, que consulta o *ID Mapper* sobre o cadastro do recurso através dos identificadores salvos na base de dados local. Com o registro verificado, o dado é enviado para o *Sender*, onde é formatado e enviado à plataforma.

No protocolo de envio sem registro, conforme a Figura 10, a fase de registro ainda existe, embora de forma implícita. Neste caso, o recurso é responsável apenas pelo envio de mensagens de dados, contendo informações de identificação do recurso juntamente com os dados propriamente ditos. Portanto, ao receber uma mensagem de dados, o *Protocol Client* realiza o processo de verificação de cadastro através do *ID Mapper*. Caso o cadastro não exista, ele é realizado de forma transparente para o recurso. Em seguida, o *Protocol Client* envia os dados ao *Sender*, que então os envia à plataforma InterSCity.

3.2. Integração com *testbed*

Nesta seção descrevemos como a integração com o *testbed* CeuNaTerra (Seção 1.3) foi planejada e realizada. A Figura 11 demonstra a arquitetura utilizada para o envio de dados de um *testbed* de sensores à plataforma InterSCity. A integração com o CeuNaTerra é realizada em três etapas: configuração do *testbed*, cadastro dos sensores e envio de dados ao *IoTGateway*.

A configuração do *testbed* CeuNaTerra é realizada através do portal web de acesso, responsável pela gestão dos sensores. O portal de acesso controla os sensores ativos dentro da rede e tem a capacidade de gerenciar e automatizar a inicialização dos testes, carregando os executáveis para os nós. Para a integração, utilizamos um TelosB

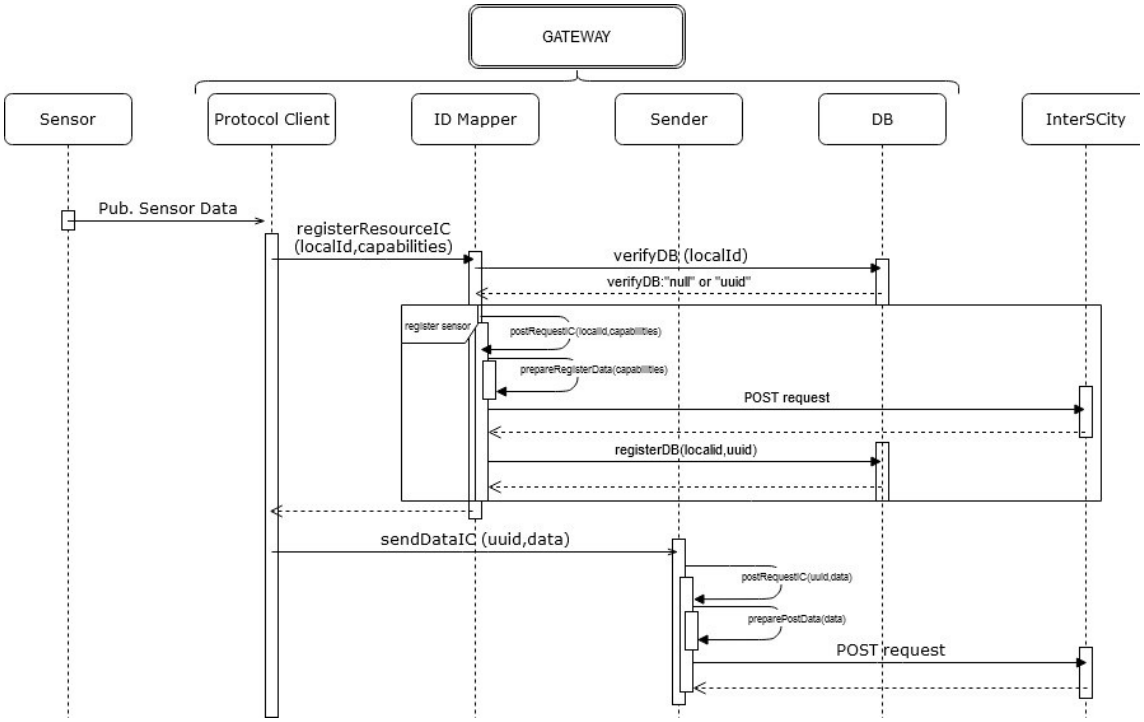

Figura 10. Protocolo de envios sem registro do *IoTGateway*.

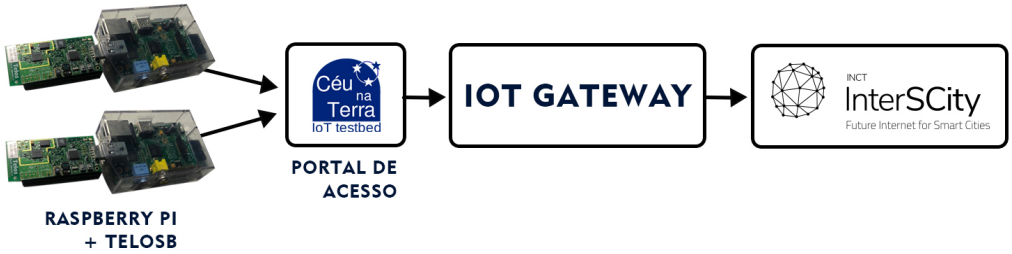

Figura 11. Modelo para implementação do *IoTGateway* com o *testbed* CeuNaTerra e com a plataforma InterSCity

ligado a um Raspberry Pi, como demonstrado na Figura 11. Os dados enviados dos sensores via CoAP chegam ao portal de acesso do CeuNaTerra e são reencaminhados ao cliente através de uma conexão de rede previamente estabelecida durante a configuração inicial.

O funcionamento do *IoTGateway* dentro da arquitetura de implementação com o CeuNaTerra segue o protocolo com registro (Seção 3.1.2) com uma alteração principal. A fase de registro dos sensores é realizada por um usuário administrador no momento da configuração dos sensores no *testbed*. Após o cadastro, o *IoTGateway* consulta os endereços cadastrados no portal de acesso periodicamente e, ao receber um dado, o componente realiza o tratamento e envio dos dados à plataforma InterSCity.

3.3. Comparação com trabalhos relacionados

Analisando o *IoTGateway* proposto para a plataforma InterSCity neste trabalho, podemos estabelecer conexões com trabalhos relacionados presentes na literatura. Conforme citado na seção 1.1, onde as plataformas CoT, dojot e FIWARE foram introduzidas, todas essas plataformas possuem características semelhantes no envio de dados de sensores.

No *Gateway* da CoT, o framework DYAMAND é responsável pela coleta de dados e pelo tratamento da interoperabilidade. Na FIWARE e na dojot o componente responsável por essa função é o *IoT Agent*, que transforma os dados dos protocolos ao padrão interno utilizado nas plataformas. No *IoTGateway* o *Protocol Client* é o responsável por essa função. Para isso o *Protocol Client*, como citado anteriormente, deve ser instanciado levando em consideração o protocolo de sensor utilizado. Novos protocolos podem ser implementados de uma forma simplificada, pois se trata de uma classe abstrata e requer apenas a implementação dos métodos que possibilitam comunicação com os sensores. Isso é semelhante ao que ocorre com os *IoT Agents* presentes na dojot e FIWARE, que possuem implementações específicas para cada protocolo utilizado.

Um dos diferenciais do *IoTGateway* é que ele não se limita apenas ao envio de mensagens. Como descrito na seção 3.1.2, os protocolos de envio com e sem registro se responsabilizam pelo cadastro dos recursos na plataforma InterSCity, abrangendo então uma quantidade maior de dispositivos para conexão e comunicação em comparação com as outras plataformas.

4. Considerações finais

Neste trabalho, apresentamos uma proposta de *IoTGateway* que possibilita a integração de sensores físicos à plataforma InterSCity por meio de um *testbed* de redes de sensores, com suporte para diversos tipos de sensores. O *IoTGateway* possibilita a implementação facilitada de novos protocolos por meio da implementação de novas versões do componente *Protocol Client*. Durante a implementação o *IoTGateway* se mostrou um elemento crucial para o envio de dados de sensores e não somente para a conexão com *testbeds*, como inicialmente proposto.

Alguns trabalhos futuros imediatos incluem uma avaliação de desempenho do *IoTGateway* para determinar seu impacto (i.e., seu *overhead*) no tempo de envio de dados dos sensores à plataforma, assim como o limite máximo de carga suportado (definido, por exemplo, pelo número máximo de sensores que pode ser ligado a um mesmo *IoTGateway* sem comprometer seu desempenho). Outros trabalhos futuros incluem a implementação e avaliação do uso de cache dentro o *IoTGateway*, assim como uma avaliação do uso de um banco de dados compartilhado entre os *IoTGateways* (na implementação atual, cada instância do *IoTGateway* possui seu próprio banco de dados). Outro aspecto importante a ser investigado em trabalhos futuros refere-se à segurança, notadamente à autenticação dos recursos ligados ao *IoTGateway*.

Referências

Arasteh, H., Hosseinnezhad, V., Loia, V., Tommasetti, A., Troisi, O., Shafie-khah, M., and Siano, P. (2016). Iot-based smart cities: A survey. In *2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC)*, pages 1–6.

Batty, M., Axhausen, K. W., Giannotti, F., Pozdnoukhov, A., Bazzani, A., Wachowicz, M., Ouzounis, G., and Portugali, Y. (2012). Smart cities of the future. *The European Physical Journal Special Topics*, 214(1):481–518.

Cabé, B. (2017). Using MQTT-SN over BLE with the BBC micro:bit. `https://blog.benjamin-cabe.com/2017/01/16/`

using-mqtt-sn-over-ble-with-the-bbc-microbit. Accessed: 2020-09-30.

Del Esposte, A. d. M., Santana, E. F., Kanashiro, L., Costa, F. M., Braghetto, K. R., Lago, N., and Kon, F. (2019). Design and evaluation of a scalable smart city software platform with large-scale simulations. *Future Generation Computer Systems*, 93:427–441.

dojot, P. (2020). Plataforma dojot. https://www.dojot.com.br. Accessed: 2020-09-30.

Esposte, A. M. D., Kon, F., Costa, F. M., and Lago, N. (2017). InterSCity: A scalable microservice-based open source platform for smart cities. In *Proceedings of the 6th International Conference on Smart Cities and Green ICT Systems*, pages 35–46.

Fazio, M., Celesti, A., Marquez, F. G., Glikson, A., and Villari, M. (2015). Exploiting the fiware cloud platform to develop a remote patient monitoring system. In *2015 IEEE Symposium on Computers and Communication (ISCC)*, pages 264–270. IEEE.

FIWARE (2020). Fiware: The open source platform for our smart digital future. https://www.fiware.org/. Accessed: 2020-09-30.

Gao, Y., Zhang, J., Guan, G., and Dong, W. (2020). Linklab: A scalable and heterogeneous testbed for remotely developing and experimenting iot applications. In *2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 176–188. IEEE.

Latre, S., Leroux, P., Coenen, T., Braem, B., Ballon, P., and Demeester, P. (2016). City of things: An integrated and multi-technology testbed for IoT smart city experiments. In *2016 IEEE International Smart Cities Conference (ISC2)*, pages 1–8.

Lloyd, P. (2015). Introduction to the MQTT protocol on NodeMCU. https://www.allaboutcircuits.com/projects/introduction-to-the-mqtt-protocol-on-nodemcu/. Accessed: 2020-09-30.

Nelis, J., Verschueren, T., Verslype, D., and Develder, C. (2012). DYAMAND: Dynamic, adaptive management of networks and devices. In *37th Annual IEEE Conference on Local Computer Networks*, pages 192–195.

Rossetto, S., Silvestre, B., Rodriguez, N., Branco, A., Kaplan, L., Lopes, I., Boing, M., Santana, D., Lima, V., Gabrich, R., et al. (2015). Ceunaterra: testbed para espaços inteligentes.

Sampaio, W. L. A. (2018). Avaliando a eficiência energética de uma conexão com a internet através do GPRS em um cenário IoT: um estudo de caso com o sim800l e o middleware dojot. Trabalho de Conclusão de Curso (Tecnologia em Redes de Computadores), UFC(Universidade Federal do Ceará), Quixadá, Brasil.

A. Informações complementares

A seguir, apresento o certificado de participação no 3° Workshop do INCT InterSCity, no qual apresentei o trabalho descrito neste relatório.

UNIVERSIDADE DE SÃO PAULO

CERTIFICADO DIGITAL

O Instituto de Matemática e Estatística
certifica que

IME-USP

Rafael Rodrigues Silva

participou do "3º Workshop do Projeto InterSCity do Instituto Nacional de Ciência e Tecnologia da Internet do Futuro para Cidades Inteligentes", com carga horária de 20 horas, no período de 7 de abril a 9 de junho de 2020

INCT
InterSCity
Future Internet for Smart Cities

Documento emitido às 16:21 horas do dia 14/07/2020 (hora e data de Brasília)
Código de controle: 9EA8-BJSW-UYF6-XY5B
A autenticidade deste documento pode ser verificada na página da Universidade de São Paulo https://uspdigital.usp.br/webdoc/

Figura 12. Certificado de participação no 3° Workshop do projeto InterSCity do Instituto Nacional de Ciência e Tecnologia da Internet do Futuro para Cidades Inteligentes