Uma arquitetura descentralizada para virtualização de sensores em plataforma de cidades inteligentes

Vinicio Vieira dos Santos[1], **Fábio Moreira Costa** [2]

[1]Estudante, Universidade Federal de Goiás – Instituto de Informática – Goiânia – GO.

[2]Orientador, Universidade Federal de Goiás – Instituto de Informática – Goiânia – GO.

vvs50@discente.ufg.br, fmc@ufg.br

Resumo. *Há um grande na aumento quantidade de sensores e dispositivos de internet das coisa disponível no dia-a-dia das pessoas. Com a maior facilidade de acesso, o uso de Plataformas de cidades inteligentes vem sendo cada vez mais importante para a integração desses dispositivos. Nesse artigo, apresentamos o projeto e implementação de protocolos para a comunicação e virtualização de sensores de baixo nível, assim como a criação de um protocolo de comunicação não centralizado entre tais sensores e a plataforma de cidades inteligentes InterSCity. E então disponibilizar um sistema distribuído pronto para o recebimentos de tecnologia de computação de Borda e nevoa (Edge e Fog Computing).*

1. Apresentação

1.1. Introdução Conceitual

A Internet das Coisas (*IoT*)[1] reúne e transmite dados de vários sensores e aparelhos inteligentes, e nos últimos anos as plataformas de cidades inteligentes (*Smart Cities*) vem sendo de grande importância para integração de sensores em sistemas distribuídos de larga escala, como em grandes centros urbanos [de M. Del Esposte et al. 2019][Bhajantri et al. 2020].

Para habilitar a comunicação entre sensores e uma plataforma de cidades inteligentes (*InterSCity*) [de M. Del Esposte et al. 2019], é necessário um *IoTGateway*[2] [R. Silva 2020], que recebe os dados dos sensores e os repassa para a plataforma. Além disso, para permitir a criação de sensores virtuais complexos a partir de sensores mais simples, um componente da plataforma denominado *IoTVirtualizer*[3] [R. Silva 2021] pode ser usando para receber dados de sensores do *IoTGateway* e os combinar antes de passá-los para os demais serviços da plataforma. Contudo, plataformas de cidades inteligentes geralmente executam na forma de serviços hospedados na nuvem, o que aumenta latência ao acessa-los. Assim, o uso de computação em névoa (*Fog Computing*) [Mahmud et al. 2018] e computação na borda (*Edge Computing*) [Garcia Lopez et al. 2015] torna-se uma forma efetiva de reduzir a latência.

O restante deste relatório está organizado como se segue. A seções 1.2, 1.3, 1.4 apresentam: a plataforma de cidades inteligentes *InterSCity*; o acesso e

[1] *Internet of Things*

[2] *https://github.com/rafaelsilvabr/IoTGateway [R. Silva 2020]*

[3] *https://github.com/rafaelsilvabr/IoTVirtualizer [R. Silva 2021]*

virtualização de sensores; e o conceito de computação em névoa. Em seguida, respectivamente, as seções 2, 3, 4 e 5 apresentam: Metodologia; Resultados e discussão; Considerações finais e trabalhos futuros;e Informações complementares.

1.2. Plataforma InterSCity

As Plataformas de cidades inteligentes oferecem suporte para o desenvolvimento e integração de aplicativos de cidades inteligentes [Santana et al. 2017]. Com o aumento da concentração demográfica em centros urbanos nas últimas décadas, cidades inteligentes contribuem para da melhora a qualidade de vida e da infraestrutura, além de otimizar o uso de recursos. Neste trabalho, a plataforma escolhida para realizar a integração do ambiente foi a plataforma *InterSCity* [de M. Del Esposte et al. 2019].

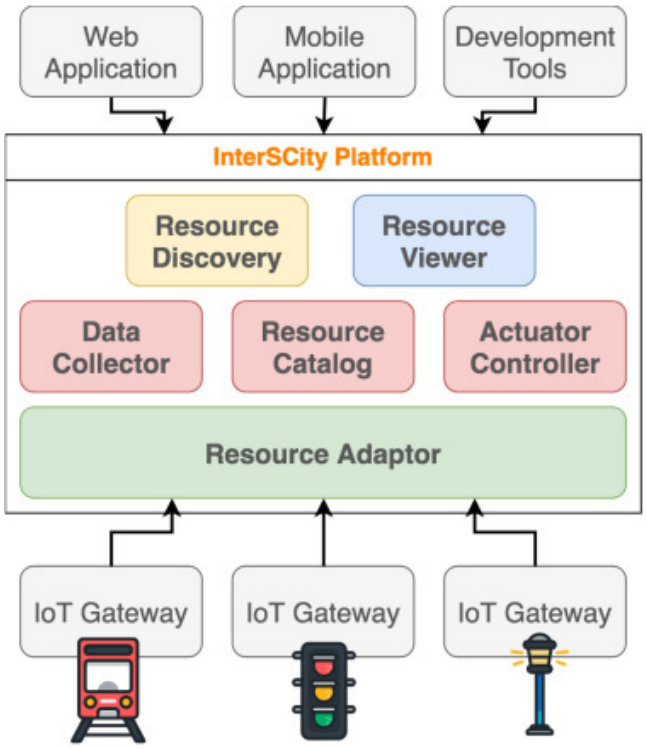


Figura 1. Arquitetura da plataforma *InterSCity*

A *InterSCity* é uma plataforma de código aberto baseada em microsserviços. Como mostrado na Figura 1, a plataforma recebe os dados de sensores, e atuadores por meio do *IoTGateway*., Esses dados são rebebidos pelo *Resource Adaptor*, o microsserviço que cadastra e é responsável pela consulta e envio de comandos aos atuadores.

O cadastro de recursos e o envio de dados para a *InterSCity* são feitos com o uso de requisições *REST*[4] que contém dados e informações dos sensores. Após o cadastro, a plataforma gera um *UUID*[5] do recurso. Devido aos diversos

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

tipos de protocolos usados nos sensores *IoT*, há a necessidade do *IoTGateway*, que traduz os dados para *REST* antes de repassá-los para a *InterSCity*.

1.3. Acesso e virtualização de sensores

A fragmentação e a concorrência no mercado de *IoT* levou à incompatibilidade na comunicação devido à existência de vários padrões. O *IoTGateway* é o componente que possibilita o envio de dados de um sensor à Plataforma *InterSCity* usando padrão *REST*, e provendo um forma de cadastrar recursos na plataforma, permitindo o uso de novos protocolos utilizados pelos sensores [R. Silva 2020].

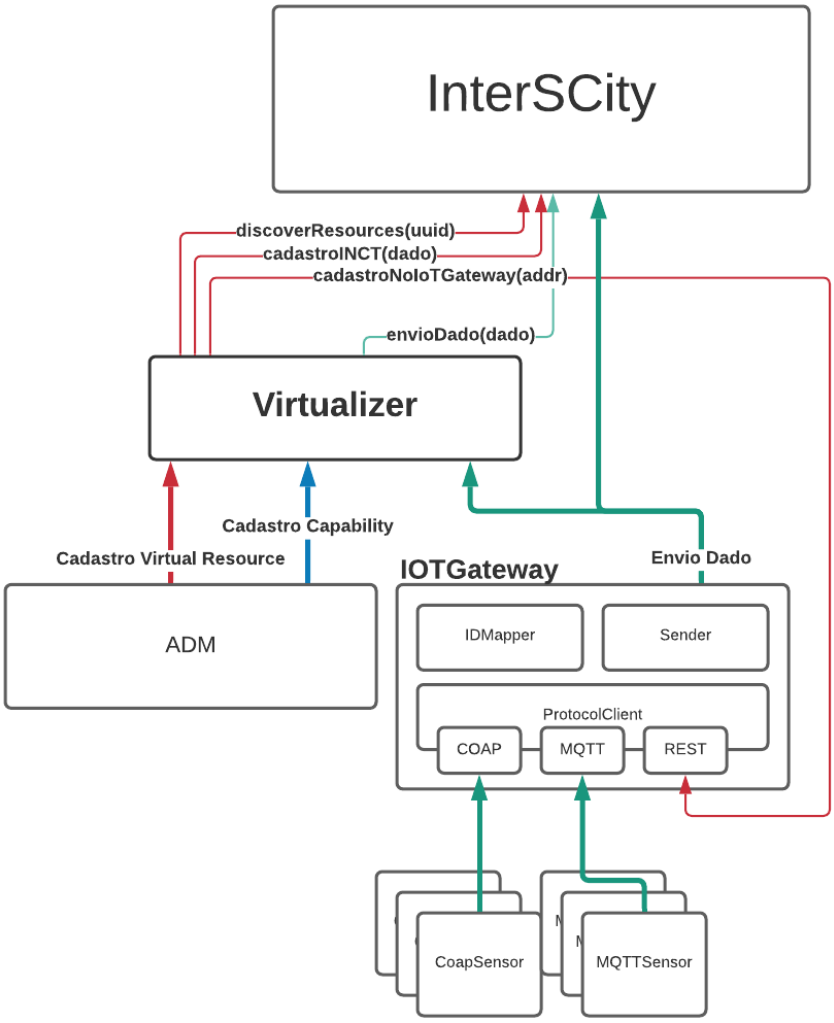


Figura 2. Fluxo de dados entre *IoTGateway*, *Virtualizar* e *InterSCity*.

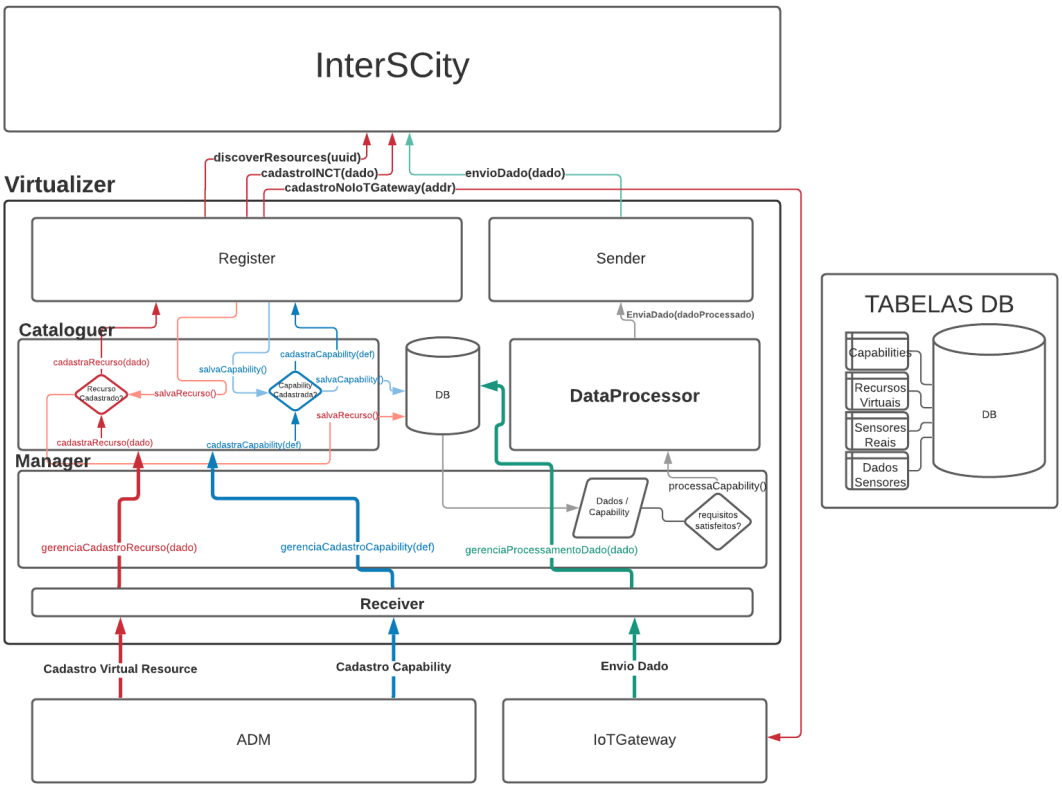


Figura 3. Fluxo de dados entre *IoTGateway*, *Virtualizar* e *InterSCity* e Arquitetura do *IoTVirtualizer*.

O *IoTVirtualizer*, por sua vez, realiza a virtualização de recursos com base nos conceitos de *Resource* e *Capabilities* da plataforma *InterSCity*. Um *Resource* possui definições de um recurso de *IoT* cadastrado na plataforma, e sensores, que podem ser definidos a partir das primitivas contidas na plataforma. As *Capabilities* possuem os dados referentes ao cadastro de um recurso e juntamente com uma definição para processamento de dados, essa definição contem a operação a ser realizada, e o tipo de dados utilizados para esse cálculo [R. Silva 2021].

No entanto, *IoTGateway* e *IoTVirtualizer* foram implementados e testados individualmente e possuem uma integração limitada. Como exemplo, o cadastro do *IoTVirtualizer* no *IoTGateway* é feito de forma estática, por meio da configuração de endereço IP e número de porta no código. Além disso, sua arquitetura é centralizada, não permitindo a criação de múltiplas instâncias desses componentes para melhorar sua escalabilidade.

1.4. Computação em nevoa (*Fog Computing*)

Fog Computing é um paradigma de computação distribuída que atua entre a Nuvem e a infraestrutura de sensores e atuadores da *IoT*, como visto na Figura 4. Os dispositivos de *IoT* são distribuídos na Borda da rede e são, em geral, sensíveis a latência. Assim, o uso de computação na névoa facilita o reconhecimento da localização, suporte a mobilidade, interações em tempo real,

escalabilidade e interoperabilidade. Deste modo, a computação em névoa resulta em melhor desempenho em termos de latência de serviço, consumo de energia, tráfego de rede, despesas operacionais e distribuição de conteúdo, entre outros [Mahmud et al. 2018].

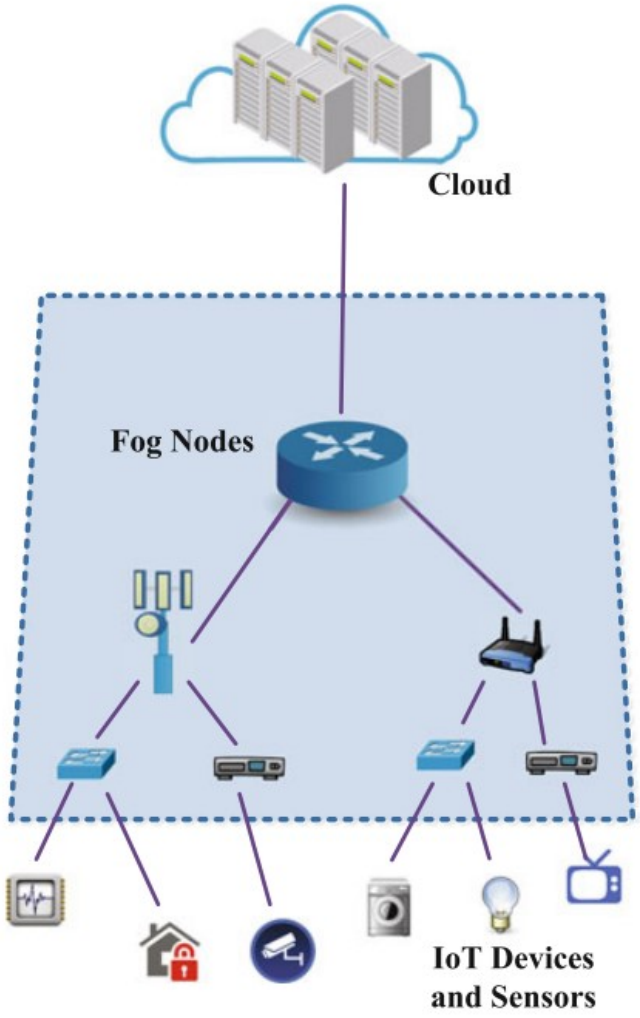


Figura 4. Sistema de *Fog Computing*

Sendo assim, considerando os benefícios da computação em névoa e a sensibilidade à latência dos dispositivos *IoT*, temos como o objetivo neste projeto implementar, nessa camada, os serviço subjacentes da *InterSCity*, notadamente o *IoTVirtualizer*. Para isso, e necessário adaptar as versões desse serviço desenvolvidas em projeto anteriores.

2. Metodologia

Este projeto é a continuação de outros projetos e, por isso, inicialmente realizamos o estudo desses projetos, para então integrar, testar e corrigir ou refatorar os serviços, *IoTGateway* e *IoTVirtualizer*, bem como alterar sua arquitetura para executá-los na névoa.

Foi também realizada uma busca na literatura para o encontro de outras abordagens para virtualização de recursos de *IoT*, notadamente com o uso de computação na névoa.

3. Resultados e discussão

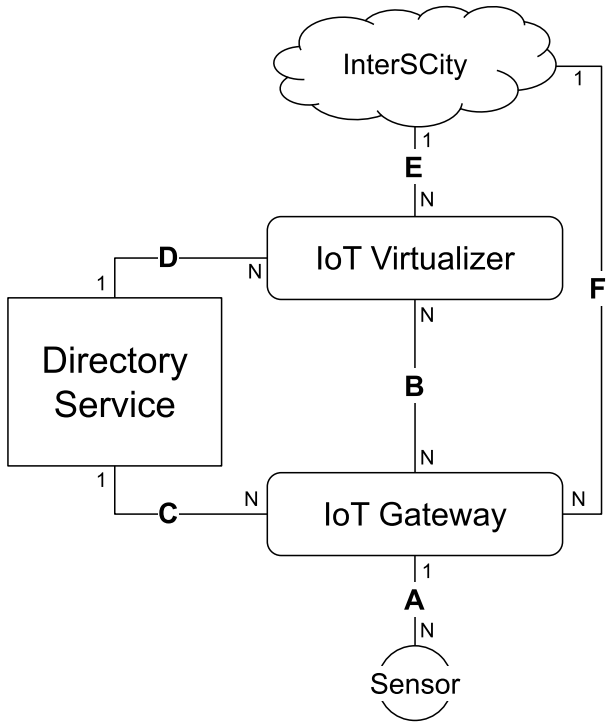
3.1. Descentralização do *IoTVirtualizer* e integração com o *IoTGateway*

Como mostrado anteriormente nas Figuras 2 e 3, no projeto anterior cada módulo foi implementado individualmente com uma integração limitada e com uma arquitetura centralizada. Portanto, foi necessário desenvolver uma maneira de executar várias instância do *IoTGateway* e de várias instância do *IoTVirtualizer* em paralelo.

Para isso, criamos um novo componente na arquitetura: um serviço de diretório, o qual é usado para registrar e permitira descoberta das as instâncias do *IoTVirtualizer* e do *IoTGateway*.

3.2. Arquitetura

A Figura 5 apresenta a arquitetura, onde as ligações entre os componente representam suas cardinalidades e as legenda descreve o significado de cada ligação.



A. Sensores se registra no IoT Gateway e envia dados.

B.

1. Após **D.2** IoT Virtualizar consulta os IoT Gateway(s) pela lista de recursos físicos e registra os que têm interesse.

2. IoT Virtualizar recebe dados dos IoT Gateway(s) registrados no **B.1**.

C. IoT Gateway se Registra no Directory Service.

D.

1. IoT Virtualizer se Registra no Directory Service.

2. IoT Virtualizer consulta Directory Service pelos IP(s) e Porta(s) dos IoT Gateway disponível(is).

E. IoT Virtualizer se registra e manda/recebe dados para InterSCity.

F. IoT Gateway se registra e manda/recebe dados para InterSCity.

Figura 5. Arquitetura com com implementação de *Directory Service*

Para a comunicação entre vários *IoTGateways* e *IoTVirtualizers*, foi implementado um serviço de diretórios para que cada parte desse sistema distribuído conheça todos os endereços dos *IoTGateways* e *Virtualizers* disponíveis

no sistema. Aos iniciá-los, eles solicitarão um endereço IP e número de porta do *IoTDirectoryService*, enquanto os outros processos rodam em paralelo.

Nas interação **C** e **D.1** da Figura 5, é feito o cadastro dos endereços IP e número de Porta dos *IoTGateways* e *IoTVirtualizers*. No *IoTDirectoryService* usanmos *ZeroMQ*[6] com modelo *request / reply* e modificamos os *IoTGateways* e *Virtualizers* para fazer o cadastro no *IoTDirectoryService*. Na interação **D.1** da Figura 5, também usamos *ZeroMQ* com o modelo Publish / subscribe, mas ainda não foi possível implementar o cadastro representado pela interação **B.1** da Figura 5, pois o *IoTGateway* não possui as *capabilites* dos sensores no banco de dados interno, e deverá ser modificado no futuro para incorporar esta característica.

4. Considerações finais e trabalhos futuros

Com a nova arquitetura, o cadastro do *IoTVirtualizer* no *IoTGateway* não é feito de forma estática. Além disso, a arquitetura não é centralizada. Isso torna possível a criação de múltiplas instâncias e também que o *IoTVirtualizer* e o *IoTGateway* conheçam todas as instâncias desses componentes no sistema distribuído.

Assim, podemos fazer a implementação na névoa neste sistema. No entanto, como mencionado anteriormente falta a implementação da interação **B.1** da Figura 5, pois o *IoTGateway* não possui as *capabilites* dos sensores em seu banco de dados interno. Também será necessária a implementação das *capabilites* no *IoTGateway* para o seu registo no *IoTVirtualizer*.

5. Informações complementares

O código-fonte dos componentes *IoTGateway*, *IoTVirtualizer* e *IoTDirectoryService* pode ser encontrado em: *https://github.com/VVSRevolution/IC_code*

Devido a choques de horários e ao preenchimento duplo de formulário para distribuição de certificados das palestras do canal do YouTube da instituição, não foi possível participar ou preencher novamente o formulário para o recebimento do certificado.

Referências

[Bhajantri et al. 2020] Bhajantri, L. B. et al. (2020). A comprehensive survey on resource management in internet of things. *Journal of Telecommunications and Information Technology*.

[de M. Del Esposte et al. 2019] de M. Del Esposte, A., 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.

[6] *ZeroMQ*

[Garcia Lopez et al. 2015] Garcia Lopez, P., Montresor, A., Epema, D., Datta, A., Higashino, T., Iamnitchi, A., Barcellos, M., Felber, P., and Riviere, E. (2015). Edge-centric computing: Vision and challenges.

[Mahmud et al. 2018] Mahmud, R., Kotagiri, R., and Buyya, R. (2018). Fog Computing: A Taxonomy, Survey and Future Directions. In Di Martino, B., Li, K.-C., Yang, L. T., and Esposito, A., editors, *Internet of Everything: Algorithms, Methodologies, Technologies and Perspectives*, Internet of Things, pages 103–130. Springer, Singapore.

[R. Silva 2020] R. Silva, R. (2020). Uso de um testbed de rede de sensores para teste e avaliação de aplicações de cidades inteligentes. Relatório Final do Programa Institucional de Bolsa de Iniciação Científica 2019-2020, Universidade Federal de Goiás (UFG), Instituto de Informática.

[R. Silva 2021] R. Silva, R. (2021). Adaptabilidade e Tolerância a Falhas No Uso De Recursos De IoT Em Uma Plataforma De Cidades Inteligentes. Relatório Final do Programa Institucional de Bolsa de Iniciação Científica 2020-2021, Universidade Federal de Goiás (UFG), Instituto de Informática.

[Santana et al. 2017] Santana, E. F. Z., Chaves, A. P., Gerosa, M. A., Kon, F., and Milojicic, D. S. (2017). Software platforms for smart cities: Concepts, requirements, challenges, and a unified reference architecture. *ACM Computing Surveys (Csur)*, 50(6):1–37.