Arquitetura para Fog Computing em Sistemas de Middleware para Internet das Coisas

Matheus Crespi Schenfeld[1], Leonardo Amaral[1], Everton de Matos[1], Fabiano Hessel[1]

[1]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)

{matheus.schenfeld, leonardo.amaral, everton.matos.001}@acad.pucrs.br,

{fabiano.hessel}@pucrs.br

Resumo. *Internet das Coisas (IoT) é considerada uma evolução computacional que preconiza a existência de uma infinidade de objetos físicos embarcados com sensores e atuadores, conectados por redes sem fio e que se comunicam através da Internet. Diversas pesquisas vêm sendo realizadas tentando mitigar os desafios existentes na IoT. Assim, cloud computing ganha espaço nesses cenários. Cloud computing refere-se ao uso de memória, armazenamento e processamento de recursos compartilhados, interligados pela Internet. No entanto, cloud computing trouxe problemas para aplicações IoT sensíveis à latência de comunicação. Para tentar minimizar esse problema, foi introduzido o conceito de fog computing, cuja ideia principal é a de distribuir serviços nos dispositivos de computação localizados nas extremidades da rede. Um grande desafio de fog computing na IoT é a definição de uma arquitetura de sistema que possa ser usada em diferentes domínios de aplicação, como saúde, cidades inteligentes entre outros. Esse trabalho apresenta uma arquitetura de sistema para dispositivos IoT capaz de habilitar o processamento de dados nos próprios dispositivos ou o mais próximo deles, em diferentes domínios, melhorando a Qualidade dos Serviços (QoS) e autonomia na tomada de decisão, mesmo se os dispositivos estiverem temporariamente desconectados da rede (offline). Apresentamos também uma breve revisão de trabalhos relacionados.*

Abstract. *Internet of Things (IoT) is considered a computational evolution that advocates the existence of a infinity of physical objects embedded with sensors and actuators, connected by the wireless networks and communicating through the Internet. Several researches have been made trying to mitigate the existing IoT challenges. Thus, cloud computing gains space in these scenarios. Cloud computing refers to the use of memory, storage and processing as shared resources, interconnected by the Internet. However, cloud computing brought problems for communication latency sensitive IoT applications. Trying to solve this problem, fog computing was introduced, whose main idea is to distribute services in the computing devices located at the ends of the network. One big challenge of fog computing in IoT is a definition of a system architecture that could be used in different domains, such as healthcare, smart cities and others. This paper presents a system architecture for IoT devices able to allow data processing on the devices themselves or as close to them, in different domains, improving Quality of Services (QoS) and autonomous decision-making, even if the devices are temporarily disconnected from the network (offline). We also present a brief review of related work.*

1. Introdução

O termo Internet das Coisas (IoT) foi introduzido em 1998 [Atzori et al. 2010] e definido como o paradigma de computação que permite que pessoas e coisas (dispositivos de computação) possam ser conectados a qualquer hora, em qualquer lugar, com qualquer coisa e qualquer um, usando qualquer caminho de rede ou serviço [Perera et al. 2014]. Conectividade entre os dispositivos é uma funcionalidade crítica que é necessária para cumprir a visão da IoT. Nesse sentido, existem estatísticas atuais de mercado e previsões que mostram um rápido crescimento em implementações de dispositivos relacionados a ambientes IoT.

A visão da IoT está em todos os lugares e pode ser aplicada em diversas áreas. Portanto, ela é fundamental no cenário brasileiro para alavancar soluções relacionadas aos grandes desafios da computação do país. Olhando para o futuro, com a evolução da tecnologia, a quantidade de dispositivos IoT só vai crescer. Em 2020, estima-se que haverá de 50 a 100 milhões de dispositivos conectados à Internet [Sundmaeker et al. 2010]. Nesse sentido, a seleção e a configuração manual desses dispositivos, juntamente com a escolha de quais os dispositivos que são relevantes para os usuários e aplicativos, não são consideradas tarefas triviais. [Perera et al. 2014].

Diversas pesquisas e avanços têm sido feitos tanto no meio acadêmico quanto no da indústria para tentar mitigar os desafios existentes na IoT. O objetivo desse esforço é tentar criar um ambiente onde todos os dispositivos a nossa volta estejam conectados à Internet, além de facilitar a comunicação entre eles e prover interações com menor intervenção humana possível.

O conceito de *cloud computing* surge da necessidade de, primeiro, colocar sistemas computacionais em ambientes altamente distribuídos e escaláveis em termos de desempenho e utilização de recursos, e segundo, de tornar simples o ambiente de configuração e reúso dos recursos. Esse termo refere-se à utilização da memória e da capacidade de armazenamento e cálculo de computadores e servidores compartilhados e interligados por meio da Internet [Patidar et al. 2011]. Entretanto, *cloud computing* trouxe problemas para sistemas IoT e aplicações sensíveis a latência de comunicação, tais como, aplicações de segurança crítica, aplicações militares, sistemas médicos emergenciais, entre outros. Além disso, aplicações consumidoras de informações e dispositivos geradores de dados estão em ambientes altamente distribuídos, o que resulta na necessidade de sistemas que melhor gerenciem a alta demanda de dados, mas com uma menor latência na comunicação.

Para tentar solucionar esse problema surge então o conceito de *fog computing*, cuja ideia principal é distribuir serviços e funções para os dispositivos computacionais (como as coisas ou dispositivos da IoT) localizados nas extremidades ou bordas da rede [Bonomi et al. 2012]. *Fog computing* tem como uma de suas principais caracteristicas, criar uma camada federada, ou seja, virtual, que tem como finalidade aproximar de forma eficiente as camadas do sistema, provedoras e consumidoras de informação, e com isso, tentar minimizar o grande volume de comunicação.

Processar informação o mais perto possível dos dispositivos IoT pode trazer vantagens às aplicações requisitantes, pois ajuda a reduzir o volume de comunicação e a carga de trabalho nos nodos da rede que processam e produzem informação (como é

o caso dos sistemas de *middleware* IoT), além de também reduzir o tráfego na rede, latência, utilização de energia, e por fim melhorar a Qualidade do Serviço (QoS) provido às aplicações e usuários da IoT.

Nesse artigo é proposto o desenvolvimento de uma arquitetura de sistema para dispositivos IoT que seja capaz de habilitar o processamento de dados nos próprios dispositivos ou o mais perto destes, provendo uma melhor Qualidade do Serviço (QoS) provido e mais autonomia na tomada de decisões, mesmo que os dispositivos estejam momentaneamente desconectados (offline), baseando-se em regras pré-estabelecidas e controladas pelo processamento de eventos complexos (CEP).

O restante desse artigo está organizado como a seguir. Na Seção 2 são apresentadas algumas definições importantes, tais como IoT, *cloud computing* e *fog computing*. A proposta de arquitetura para *fog computing* é apresentada na Seção 3. Os trabalhos relacionados e uma breve discussão sobre a arquitetura proposta são apresentados na Seção 4. Por fim são apresentados a conclusão e os trabalhos futuros na Seção 5.

2. Referêncial Teórico

Nessa seção são brevemente apresentados alguns conceitos comumente citados no decorrer do trabalho. Em 2.1 é apresentada a definição de Internet das Coisas. Na subseção 2.2 é apresentado o conceito de *cloud computing*. Por fim, a subseção 2.3 apresenta a definição de *fog computing*.

2.1. Internet das Coisas

A criação da internet revolucionou o funcionamento tradicional das sociedades modernas, e com o uso de protocolos de endereçamento, tornou-se possível a comunicação entre dispositivos separados fisicamente, rompendo diversos paradigmas da distância e do tempo. A possibilidade de quaisquer dispositivos - como, Identificação por Radio Frequência (RFID), tags, sensores, atuadores, smartphones, entre outros - conectados à rede, em qualquer lugar e a qualquer hora, iniciou a era da ubiquidade, principal caracteristica e pilar fundamental para o surgimento da Internet das Coisas ou em inglês, Internet of Things (IoT) [Tiburski et al. 2015].

O conceito da IoT é uma evolução de diversos elementos tecnológicos (sensores, *hardware*, semântica, armazenamento, processamento, comunicação, entre outros), que ao serem unidos em um mesmo ambiente, representam o futuro da computação e das comunicações [Tan 2010]. A IoT tem o objetivo final de criar um mundo melhor para todas as pessoas, onde objetos a nossa volta tenham conhecimento e autonomia, agindo de forma inteligente sem instruções explícitas [Perera et al. 2014].

Uma das funcionalidades essenciais para o funcionamento de um sistema IoT é a camada de *middleware* [Atzori et al. 2010]. Dentre as suas caracteristicas, a principal é a de abstrair os protocolos técnicos, tais como, protocolo de comunicação, tipo de porta de comunicação, entre outras, das diferentes tecnologias encontradas nos dispositivos, tornando assim a tarefa do programador de aplicações algo mais fácil e prazeroso. O *middleware* tem ganhado muito mais importância nos últimos anos, visto que facilita o desenvolvimento de novos serviços e a integração de tecnologias já existentes com as novas em desenvolvimento [Amaral et al. 2015]. Isto tira a necessidade do programador

de ter o conhecimento específico dos dispositivos, que atualmente têm variados tipos de protocolos de comunicação e funcionalidades [Schenfeld et al. 2014].

Entretanto, a possibilidade de poder estar conectado a diversos dispositivos e independente de localização, assim como prover para o desenvolvedor de sistemas a chance de poder utilizar recursos que facilitam o ambiente de configuração, adaptação e monitoramento de serviços, motivou a utilização de tecnologia de *cloud computing* para melhorar ainda mais a IoT.

2.2. Cloud Computing

Cloud Computing ou Computação em Nuvem, é o resultado da evolução e adoção de tecnologias e paradigmas computacionais consolidados [Zhou et al. 2010]. A principal tecnologia que permite a computação em nuvem é a virtualização, que separa um dispositivo físico de computação em um ou mais dispositivos virtuais, fazendo com que cada dispositivo possa ser facilmente utilizado e gerenciado para executar tarefas. Com a virtualização em nível de sistema operacional, a criação de um sistema escalável de vários dispositivos de computação independentes, além de recursos computacionais ociosos, podem ser atribuídos e utilizados de forma mais eficiente. O objetivo da computação em nuvem é permitir que os usuários tirem benefício de todas essas tecnologias, sem a necessidade de conhecimento profundo ou experiência com cada uma delas [Hamdaqa and Tahvildari 2012].

Entretanto, mesmo com diversas vantagens, *cloud computing* tem apresentado algumas limitações [Patidar et al. 2011], fazendo com que aplicações sensíveis a latência de comunicação sejam afetadas, visto que aplicações consumidoras de informações e dispositivos geradores de dados estão em ambientes distribuídos. Problemas com largura de banda e consequentemente os custos decorrentes disso também fazem parte das limitações. Para solucionar estes e outros problemas surge o conceito de *fog computing* [Bonomi et al. 2012].

2.3. Fog Computing

Fog Computing surge como uma infraestrutura de computação na qual aplicações e serviços podem ser tratados tanto em servidores *cloud*, como na própria rede [Yi et al. 2015]. O conceito *fog* surge visando atender a três objetivos principais:

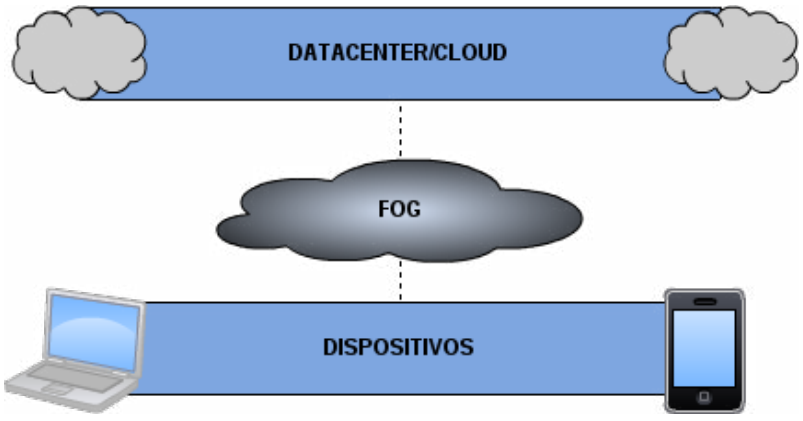

Figura 1. Camada virtual de fog computing.

- Melhorar a eficiência e reduzir a quantidade de dados que precisam ser transmitidos para que seja feito o processamento, análise e armazenamento.
- Aproximar o consumidor de informações com o provedor de dados.
- Prover segurança e conformidade para a transmissão de dados.

Em *fog computing*, grande parte do processamento ocorre nos próprios dispositivos que geram os dados, tendo em vista que nos últimos anos o seu poder de processamento e armazenamento teve significativa evolução. A arquitetura *fog* cria uma plataforma virtual que fornece serviços de processamento e de armazenamento entre a nuvem *cloud* e os dispositivos. Em outras palavras, cria uma camada federada, isto é, a união de diversas funcionalidades que formam uma camada virtual em ambiente descentralizado e mais próximo dos dispositivos de borda (ou *Edge Devices*), reduzindo a latência na rede e largura de banda, solucionando ao menos parcialmente os problemas encontrados em *cloud computing* [Lewis et al. 2014]. A Figura 1 ilustra a camada virtual criada em *fog computing* aproximando os dispositivos e a camada *cloud computing*.

3. Arquitetura do Sistema

Tendo como objetivo solucionar o problema exposto anteriormente, além de atender algumas das tendências futuras para sistemas IoT descritos por L. Tan et al. [Tan 2010] - dispositivos com inteligência; arquitetura dirigida a eventos; grande quantidade de dispositivos conectados ao sistema, bem como o volume de dados produzidos por eles - é proposto nesse artigo o planejamento e desenvolvimento de uma infraestrutura com suporte para *fog computing* em arquiteturas de *middleware* Orientados a Serviços (SOA) que provê mais autonomia para os dispositivos.

A arquitetura proposta, que pode ser vista na Figura 2, é constituida por cinco camadas: Camada de Aplicações, resposável por entregar os dados processados; Camada de *Middleware*, que executa as principais funções da arquitetura; Camada de *Fog Computing*, elemento principal da pesquisa desse trabalho; Camada de Comunicação, responsável pelas conexões entre diferentes plataformas, e por fim a Camada das Coisas, ou em outras palavras, camada dos dispositivos heterogêneos.

3.1. Camada de Fog Computing

Iniciamos a descrição da arquitetura pela camada de *fog computing*, visto que é o principal elemento do trabalho. A camada de *fog* tem como objetivo criar uma camada federada, ou seja, virtual, que tem como principal função aproximar de forma eficiente as camadas de aplicações e as camadas de dispositivos. Na arquitetura proposta, essa camada virtual será alocada em ambiente de *System-on-a-Chip* (SoC), um sistema computacional completo num único chip. Tendo como justificativa o grande poder de processamento, bem como a variedade de tipos de conexão, tais como, WiFi, *Bluetooth, Ethernet*, serial, entre outras, cada SoC atuará como um *Edge-Gateway*, que consiste em um concentrador de processamento de um ou mais dispositivos. Por exemplo, um *Edge-Gateway* de uma *SmartHome*, que pode estar conectado com o sistema de iluminação, eletroeletrônicos, carros, entre outros. Os dispositivos por sua vez, não mais se comunicam com o *middleware* central, mas sim com o *Edge-Gateway*, criando um ambiente virtual de *fog computing*.

3.2. Camada das Coisas

Essa camada consiste em sensores, sistemas embarcados, RFID tags e quaisquer outros objetos que possam estar conectados pela rede. Essas entidades são as que capturam os dados do ambiente.

3.3. Camada de Comunicação

A camada de comunicação conecta diferentes tipos de fontes de dados e serviços localizados em ambiente altamente distribuído. O primeiro estágio de tratamento dos dados coletados acontece nessa camada. Responsável pelas mensagens de roteamento, serviços *Publish/Subscribe*, e também por realizar a comunicação entre plataformas, se necessário.

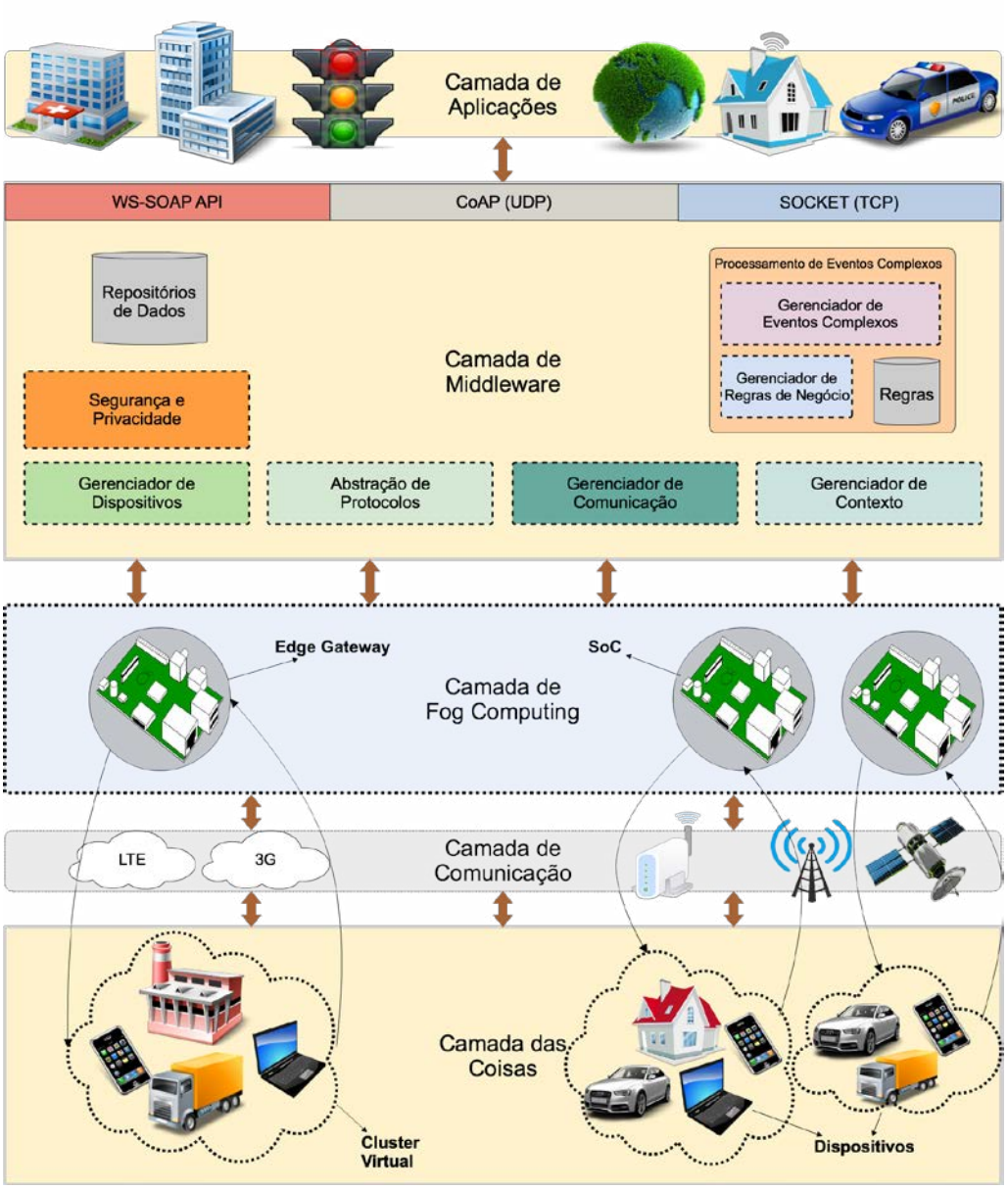

Figura 2. Arquitetura de fog computing

3.4. Camada de Middleware

Essa é a camada responsável por realizar as principais funções do sistema e opera em modo bidirecional. Ela age como uma interface entre a camada de *hardware* e a camada das aplicações e é responsável por funções essenciais, tais como, gerenciamento de dispositivos, gerenciamento de informações, filtragem dos dados, agregação de valores, análise de semântica, controle de acesso, além da abstração dos protocolos de acesso dos diversos dispositivos da camada de dispositivos. Essas funções são divididas em módulos, listados a seguir:

- **Abstração de protocolos:** Tendo em vista que os dispositivos IoT conectados à rede são heterogêneos, é esperado que cada um tenha características próprias, como protocolos de conexões, tipos de dados, ou linguagem de comunicação. É necessário que exista uma abstração desses protocolos para tornar o acesso aos dispositivos mais harmônico.
- **Gerenciador de dispositivos:** Módulo responsável por prover suporte à conexão de novos dispositivos através de informações de protocolos de comunicação e conexão de dispositivos já conhecidos.
- **Gerenciador de comunicação:** Módulo responsável por administrar as comunicações entre as camadas consumidoras e provedoras de serviços. Responsável também por definir o melhor tipo de protocolo de aplicação para ser usado. Exemplos de protocolos são: *Socket* (TCP), WS-SOAP (API) e o *Constrained Application Protocol* (CoAP).
- **Gerenciador de contexto:** O contexto é qualquer informação que possa ser extraída de uma entidade a fim de caracterizá-la. Essa entidade pode ser um usuário, um dispositivo ou um nodo *fog*. O contexto pode ser usado para prover serviços de informações relacionados a entidades. O módulo gerenciador de contexto é responsável por armazenar informações de entidades e também realizar procedimentos com essas informações (raciocínio de contexto), e assim obter informações contextualizadas.
- **Segurança e privacidade:** É essencial a presença de mecanismos de segurança em ambientes IoT. Muitas vezes as informações que trafegam em redes IoT são sigilosas, como por exemplo informações médicas ou informações pessoais. O *middleware* para IoT precisa ter funções de segurança que protejam os dados trafegados e também que não permita a alteração dos mesmos.
- **Processamento de Eventos Complexos:** O módulo para processamento de eventos complexos (CEP) opera através da avaliação da confluência de eventos, e consequentemente, da execução de uma ação. Os eventos podem ocorrer por um longo período de tempo. A correlação de eventos pode ser ocasional, temporal ou espacial. CEP requer o emprego de sofisticados intérpretes de eventos, detecção de padrões de acontecimento, e técnicas de correlação. CEP é comumente usado para detectar e responder a anomalias de negócios, ameaças e oportunidades. Geralmente, motores de ontologias e regras são usados para fornecer vocabulários comuns de um domínio e conhecimento comportamental, expressando restrições e reação a eventos [Laliwala and Chaudhary 2008].

4. Trabalhos Relacionados e Discussão

No âmbito da Internet das Coisas, diversas plataformas de *cloud computing* podem ser utilizadas para facilitar o desenvolvimento e a implantação de sistemas e aplicações para IoT. Exemplos conhecidos dessas plataformas são: ThingWorx, OpenIoT, Xively, Nimbits, Axeda, entre outras. A Tabela 1 resume algumas das características mais importantes destas e de outras plataformas de *cloud computing* para IoT (na tabela, "+"significa suporta e -"significa não suporta).

As métricas de avaliação utilizadas na Tabela 1 incluem: suporte a *Edge-Gateway*, fazendo uma ponte entre a camada de *fog* e camada de *middleware*. Gerenciamento, que diz respeito ao suporte à descoberta de dispositivos, entrega das informações, configuração e ativação de aplicações e serviços, garantindo a autonomia. Suporte a qualidade e confiabilidade de aplicações e serviços (Veracidade). E por último, o suporte de protocolos de comunicação padronizados.

Xively, por exemplo, representa uma das primeiras plataformas para IoT disponíveis na web. Xively tem como objetivo conectar os dispositivos com as aplicações, garantindo a segurança em tempo real. Xively fornece uma Plataforma como Serviço (PaaS) para os desenvolvedores de aplicações IoT, além dos prestadores de serviços. Ela é capaz de integrar os dispositivos com a plataforma através do uso de bibliotecas prontas (como ARM mbed, *Eletric Imp* e iOS/OSX) e também de facilitar a comunicação via HTTP(S), ou *Sockets/WebSocket* [Yang et al. 2013]. É possível também integrar Xively com outras plataformas que utilizam bibliotecas Java, JS, Python e Ruby.

Como outro exemplo, Nimbits é uma Plataforma como Serviço (PaaS) de código aberto que conecta dispositivos inteligentes com a nuvem [Doukas 2012]. Ele também realiza a análise de dados na nuvem, gerando alertas, além de se conectar com redes sociais e planilhas. Além disso, ele se conecta a sites e pode armazenar, compartilhar e recuperar dados dos sensores em diversos formatos, incluindo numérico, texto, GPS, JSON ou XML. A troca de dados ou mensagens XMPP é um serviço embutido no Nimbits. O núcleo do Nimbits é um servidor que fornece serviços Web REST para o registro e recuperação de dados brutos e processados.

Axeda é uma outra plataforma para integração de dados e desenvolvimento de aplicações, bem como para conexão de dispositivo e outros serviços comumente oferecidos em serviços de *cloud computing* [Bjelica et al. 2014]. Essa plataforma oferece suporte a *Edge-Gateway* desde que os dispositivos a serem conectados atendam a algumas especificações de Firewall. Axeda também dá suporte ao controle de eventos em tempo real. Além disso, ela também provê ao desenvolvedor uma interface de alto nível, tornando o ambiente de configuração e reutilização de recursos facilitado. A plataforma também dispõe de serviços e aplicações com funções de gerenciamento de dados, juntamente com aplicações web e APIs que tem como finalidade estender a plataforma usando Java SDK.

Como pode ser visto nos trabalhos relacionados da Tabela 1, mesmo que existam diversas soluções até mesmo comerciais, elas ainda não atendem por completo aos requisitos de uma arquitetura de *cloud* e *fog computing* explanados em Al-Fugaha et al. [Al-Fuqaha et al. 2015]. Exemplos desses requisitos (ou deficiências) são: nenhuma plataforma oferece suporte a CoAP, mesmo ele sendo uma padronização que é baseada em

Tabela 1. Plataformas Cloud IoT e Suas Características

Plataforma	Gateway	Gerenciamento	Veracidade	REST	CoAP
Arkessa	-	+	+	+	-
Axeda	+	+	+	+	-
LittleBits	-	-	-	+	-
NanoService	+	+	+	+	-
NinjaBlocks	+	-	-	+	-
Nimbits	-	-	-	+	-
RealTime.io	+	+	-	+	-
TempoDB	-	-	-	+	-
Thingworx	-	+	+	+	-
Xively	+	+	+	+	-

REST, porém com diversas melhorias. Em cinco plataformas, não há suporte para a conexão dos *edge-gateways*, o que impede que um conjunto de aplicações possa ser executada na plataforma, por exemplo, em alguns cenários de aplicações existe a necessidade de utilizar as capacidades de um *edge-gateways*, dentre elas, agregação, eliminação e filtragem dos dados, processamento em tempo real, análise de dados, entre outras. Além disso, também há pouco suporte no quesito de manter a qualidade e confiabilidade das informações.

A arquitetura de *cloud* e *fog computing* apresentada neste trabalho (Figura 2) identifica várias vantagens em comparação com trabalhos similares encontrados. Entre elas pode-se destacar a conexão com diversos tipos de equipamentos e dispositivos através do uso de diferentes redes de comunicação, como por exemplo, WiFi, *Bluetooth*, LTE, entre outras. A utilização do CoAP como uma das alternativas de protocolos de aplicações também pode ser citada, tendo em vista que representa um método simplificado para a tranferência de dados entre servidores e consumidores através do HTTP. Diferente do REST, o CoAP está vinculado ao UDP (não ao TCP) por padrão, o que o torna mais viável para aplicações IoT. As capacidades da camada de *middleware* também tornam-se uma vantagem, pois, garantem a qualidade e confiabilidade do sistema. Entre as capacidades pode-se citar o suporte a descoberta, gerenciamento, configuração e adaptação, tanto de dispositivos como de contexto dos mesmos. O módulo de segurança também garante a transferência de dados de forma segura e privada. Além disso, a utilização de um SoC como *Edge-Gateway* torna-se um ponto positivo devido ao baixo custo e ao alto poder de processamento.

Além das vantagens apresentadas, a arquitetura também está preparada para atender algumas das tendências futuras para IoT, como por exemplo, estar preparada para o tamanho da rede, visto que algumas pesquisas mostram que até 2020 existirão entre 50 e 100 bilhões de dispositivos conectados a internet [Sundmaeker et al. 2010].

5. Conclusão e Trabalhos Futuros

Nesse trabalho foi proposta e detalhada uma arquitetura de *fog computing* para IoT que tem como objetivo aproximar de maneira eficiente as camadas consumidoras e provedoras de serviços.

É importante mencionar que *fog computing* não é um substituto de *cloud com-*

puting, mas sim, que essas duas tecnologias complementam uma a outra. A principal motivação do trabalho são as funções complementares que *fog* e *cloud* são capazes de proporcionar, as quais permitem ao usuário experimentar uma nova geração da computação, e também servem como requisito para que aplicações de tempo real e de baixa latência possam ser executadas nas bordas da rede. Além disso, essa combinação também permite suporte para análises completas de uma grande quantidade de dados na camada principal da rede.

Na continuidade dos trabalhos, a arquitetura proposta na Figura 2 terá sua implementação finalizada, bem como a realização de testes e validações em casos de uso reais. Os testes serão realizados no *Smart City Innovation Center* da PUCRS em parceria com a empresa Huawei. Tanto a PUCRS quanto a Huawei têm investido significativamente no desenvolvimento de novas tecnologias para impulsionar o crescimento da IoT voltado para cidades inteligentes.

Referências

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., and Ayyash, M. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols and Applications. *IEEE Communications Surveys & Tutorials*, PP(99):1–1.

Amaral, L. A., Tiburski, R. T., de Matos, E., and Hessel, F. (2015). Cooperative middleware platform as a service for internet of things applications. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing - SAC '15*, pages 488–493, New York, New York, USA. ACM Press.

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

Bjelica, M. Z., Golan, G., Radovanovic, S., Papp, I., and Velikic, G. (2014). Adaptive device cloud for internet of things applications. In *Consumer Electronics-China, 2014 IEEE International Conference on*, pages 1–3. IEEE.

Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog Computing and Its Role in the Internet of Things. *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16.

Doukas, C. (2012). *Building Internet of Things with the ARDUINO*. CreateSpace Independent Publishing Platform.

Hamdaqa, M. and Tahvildari, L. (2012). *Cloud Computing Uncovered: A Research Landscape*, volume 86.

Laliwala, Z. and Chaudhary, S. (2008). Event-driven service-oriented architecture. In *Service Systems and Service Management, 2008 International Conference on*, pages 1–6. IEEE.

Lewis, G., Echeverria, S., Simanta, S., Bradshaw, B., and Root, J. (2014). Tactical cloudlets: Moving cloud computing to the edge. In *Military Communications Conference (MILCOM), 2014 IEEE*, pages 1440–1446.

Patidar, S., Rane, D., and Jain, P. (2011). A survey paper on cloud computing. *Proceedings - 2012 2nd International Conference on Advanced Computing and Communication Technologies, ACCT 2012*, pages 394–398.

Perera, C., Zaslavsky, A., Christen, P., and Georgakopoulos, D. (2014). Context aware computing for the internet of things: A survey. *Communications Surveys Tutorials, IEEE*, 16(1):414–454.

Schenfeld, M. C., Vargas, F. D., Rebonatto, M. T., and Paixão, O. (2014). Middleware para equipamentos médicos em System on a Chip. *Brazilian Congress on Biomedical Engeneering*, pages 2608–2611.

Sundmaeker, H., Guillemin, P., and Friess, P. (2010). *Vision and challenges for realising the Internet of Things*. Number March.

Tan, L. (2010). Future internet: The Internet of Things. *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, pages V5–376–V5–380.

Tiburski, R. T., Amaral, L. A., Matos, E. D., and Hessel, F. (2015). The importance of a standard securit y archit ecture for SOA-based iot middleware. *IEEE Communications Magazine*, 53(12):20–26.

Yang, K.-p., Alkadi, G., Gautam, B., Sharma, A., Amatya, D., Charchut, S., and Jones, M. (2013). Park-a-lot: An automated parking management system. *Computer Science and Information Technology*, 1(4):276–279.

Yi, S., Li, C., and Li, Q. (2015). A Survey of Fog Computing. *Proceedings of the 2015 Workshop on Mobile Big Data - Mobidata '15*, pages 37–42.

Zhou, M., Zhang, R., Zeng, D., and Qian, W. (2010). Services in the cloud computing era: A survey. *2010 4th International Universal Communication Symposium, IUCS 2010 - Proceedings*, pages 40–46.