CoAP-CTX: Extensão Sensível ao Contexto para Descoberta de Objetos Inteligentes em Internet das Coisas

Felipe M. Barreto, Windson Viana, Marcio E. F. Maia, Rossana M. de C. Andrade*

Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat)
Mestrado e Doutorado em Ciências da Computação (MDCC)
Universidade Federal do Ceará (UFC)
Fortaleza – CE – Brasil

{felipebarreto, windson, marcio, rossana}@great.ufc.br

Abstract. *In the Internet of Things vision, smart objects are interconnected in order to allow the creation of applications embedded in everyday environments (e.g., house, cars, schools, buildings). The number of smart objects tends to increase in the next years, creating an overload of objects to be controlled and configured by the users. Alternatively, Context-aware discovery services have the potential to minimize this problem by applying rules to determine wich smart objetcs will be discovered at a certain time. This work proposes the CoAP-CTX, an extension of the built-in discovery service present in CoAP protocol that aims to provide support to a selective discovery of smart objects. Aditionally, smart objetcs that are not on the user's interest go into an idle state, optimizing the network and battery usage. Experiments have shown that CoAP-CTX reduces the total number of messages transmitted in the local network, at a cost of a acceptable latency overhead to perform the discovery.*

Resumo. *Em um ambiente de Internet das Coisas, objetos inteligentes estão interligados de modo a permitir a criação de aplicações em lugares da vida cotidiana (por exemplo, casas, carros, escolas, edifícios). O número desses objetos inteligentes tende a aumentar mais ainda nos próximos anos, criando uma sobrecarga de objetos a serem controlados e configurados pelos usuários. Serviços de descoberta sensíveis ao contexto tem o potencial de minimizar este problema aplicando regras para determinar quais objetos inteligentes serão descobertos a cada vez. Este trabalho propõe o CoAP-CTX, uma extensão do serviço de descoberta padrão do protocolo CoAP que tem por objetivo dar suporte a uma descoberta seletiva de objetos inteligentes. Além disso, objetos inteligentes que não são de interesse do usuário entram em um modo de espera, otimizando o uso da rede e da bateria. Experimentos mostraram que o CoAP-CTX reduz o número total de mensagens transmitidas na rede local, a um custo de um aumento aceitável no tempo gasto para realizar a descoberta.*

1. Introdução

Internet das Coisas, do inglês *Internet of Things* (IoT), é um paradigma tecnológico que emerge no cenário já consolidado das redes de comunicações sem fio. A ideia principal desse conceito é a constante e invisível presença, no cotidiano das pessoas, de uma

*Bolsista de produtividade DT-2 do CNPq

enorme variedade de dispositivos computacionais. Alguns desses dispositivos possuem poder computacional, juntamente com a capacidade de comunicação, e são chamados de Objetos Inteligentes (OI). Esses objetos são unicamente endereçáveis, capazes de interagir, trocar dados entre si e ainda cooperar com seus vizinhos para realizarem tarefas em comum [Atzori et al. 2010]. Ao longo dos últimos anos, o volume mundial de OI cresceu rapidamente. A previsão é de que no ano de 2020 será alcançada uma marca de 50 bilhões de OI conectados [Dave 2011] e o número de dispositivos conectados à rede pode chegar à 13,6 por pessoa [Cisco 2016]. Grande parte desses OI já são compatíveis com arquiteturas e protocolos Web (HTTP, REST, entre outros).

As abordagens de descoberta de OI tradicionais tem por objetivo principal descobrir e tornar acessível todos os OI alcançáveis. Entretanto, se o número de OI for muito elevado, uma abordagem mais apropriada seria priorizar, ou recomendar, os OI de maior interesse para o usuário, por exemplo, a partir do contexto capturado por seu *smartphone* (e.g, localização, histórico, entre outros). Além das vantagens diretamente relacionadas ao usuário, com uma descoberta mais seletiva é possível economizar os recursos computacionais desses OI, os quais possuem por natureza limitações de energia e memória [Duarte et al. 2014].

Dispositivos móveis, como o *smartphone*, são uma ótima escolha para serem utilizados como elementos que permitem a interação com OI que estejam presentes ao redor do usuário, porém apresentam limitações com relação a suficiência energética [Ríos et al. 2016]. Dentre as principais operações que consomem energia nos dispositivos móveis, as que mais têm impactado para o aumento desse consumo são as operações de comunicação, como troca de mensagens [Tarkoma et al. 2014]. Dispositivos ainda mais limitados computacionalmente, como é o caso da grande parte dos OI existentes, sofrem do mesmo problema energético, porém de maneira mais intensificada.

A seguir, é apresentado um cenário motivador de descoberta de OI em IoT em que é possível identificar onde a sensibilidade ao contexto pode ser utilizada. Ana é uma pessoa bastante ligada ao uso de tecnologia, costuma acordar cedo para ir ao trabalho, mas não sem antes conferir a previsão do tempo pela TV. Com seu *smartphone*, Ana consegue acessar e ligar a TV, bem como selecionar o canal da previsão. Ela então segue para o trabalho. Usando o *smartphone*, seleciona o noticiário econômico no rádio do carro. Ao chegar no seu destino, Ana prepara a sala de seminários da empresa para uma reunião, novamente utiliza seu *smartphone* para ligar o projetor e as luzes do ambiente.

O cenário apresentado possui três ambientes distintos: a casa, o carro e o local de trabalho. A sensibilidade ao contexto atua em todos eles. Na situação da casa e do carro, embora a tendência seja que o número de dispositivos conectados aumente, pode-se assumir que o volume total ainda será controlado. Nesses casos, a sensibilidade ao contexto pode atuar na melhoria do gerenciamento dos dispositivos, reduzindo o consumo de energia. Por exemplo, os OI que não fossem do interesse de Ana naquele momento (televisão e rádio) poderiam entrar em modo de espera. Já na outra situação, em um ambiente de trabalho, geralmente o número de OI é bem maior, e geralmente são compartilhado por vários usuários (microfones, caixas de som, projetores, etc.). Nesse caso, a vantagem de se utilizar de técnicas de sensibilidade ao contexto na descoberta desses objetos está na interação entre usuário e sistema.

Para a comunicação com os OI, já existem protocolos especializados em dispositivos limitados computacionalmente, como é o caso do *Constrained Application Protocol*[1] (CoAP) e do MQ Telemetry Transport[2] (MQTT). Este trabalho apresenta uma extensão do CoAP, protocolo de troca de mensagens que já possui um serviço de descoberta de OI, de modo a permitir que essa descoberta utilize informações contextuais para selecionar os OI de interesse do usuário. Desse modo, é possível identificar e fazer com que os OI que não sejam de interesse entrem em modo de espera, reduzindo o número de mensagens trocadas na rede.

O restante deste documento está organizado como segue: a Seção 2 apresenta definições e conceitos utilizados neste trabalho. A Seção 3 descreve a extensão do CoAP proposta, bem como um estudo de caso implementado em um ambiente real. A Seção 4 apresenta a avaliação realizada por meio de simulações. A Seção 5 traz discussões sobre trabalhos relacionados com descoberta de OI. Por fim, a Seção 6 apresenta as considerações finais e trabalhos futuros.

2. Fundamentação Teórica

2.1. Sensibilidade ao Contexto

[Dey et al. 2001] definem contexto como qualquer informação que pode ser usada para caracterizar a situação de um elemento que é relevante para a interação entre usuário e sistema, incluindo como elementos, o próprio usuário e o sistema. A definição que será adotada neste trabalho, proposta por [Viana et al. 2011], pode ser vista como uma extensão da anterior, removendo a limitação do contexto sobre a necessidade de interação entre usuário e sistema. Contexto agora passa a ser um conjunto de informações que podem descrever a situação das entidades (e suas relações) envolvidas em uma ação que seja julgada importante para o sistema (e.g., interações, busca de dispositivos, entre outros). A Figura 1 exemplifica visualmente essa definição.

O contexto é definido como a interseção entre duas zonas. A primeira delas, chamada de Zona de Interesse (ZI), representa o conjunto de entidades que o sistema julga importante em um determinado instante de tempo, ou seja, quais informações contextuais o sistema tem interesse em obter em um instante *t*. A segunda zona, chamada de Zona de Observação (ZO), é composta pelas entidades que podem ser acessadas e obtidas nesse mesmo instante de tempo pela infraestrutura de aquisição de contexto. Essa definição de contexto possui duas características importantes. O contexto é tanto dinâmico (as informações fornecidas por cada uma das entidades muda com o tempo) quanto evolutivo (as próprias entidades que compõem o contexto podem mudar) [Duarte et al. 2015]. Isso faz com que a aplicação dessa definição em sistemas móveis, ubíquos e de IoT seja bastante adequada, pois os elementos que compõem esses sistemas são bastante voláteis, o que faz com que as zonas mudem constantemente.

2.2. Descoberta de Objetos Inteligentes

Descoberta de serviços é um aspecto conhecido do desenvolvimento de sistemas distribuídos. [Crasso et al. 2008] definem essa descoberta como um processo de encontrar os serviços adequados para resolver uma determinada tarefa em particular. Esses serviços

[1]CoAP: https://tools.ietf.org/html/rfc7252

[2]MQTT: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

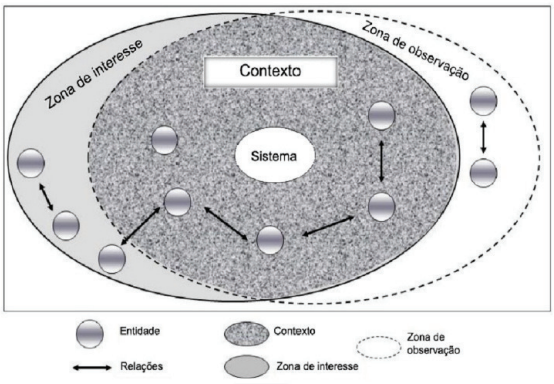

Figura 1. Contexto como sendo a interseção entre as Zonas de Interesse e de Observação [Duarte et al. 2015].

podem ser vistos tanto como abstrações de softwares disponíveis (e.g., um servidor de um jogo) quanto como recursos ou dispositivos computacionais passíveis de serem encontrados (e.g., impressoras em uma rede). *Dynamic Host Configuration Protocol* (DHCP), *Simple Service Discovery Protocol* (SSDP) e *Universal Plug and Play* (UPnP) são exemplos de protocolos que possuem mecanismos de descoberta de serviços. Já para a descoberta de OI em IoT, é preciso levar em consideração requisitos que são específicos para ambientes inteligentes, como os levantados por [Guinard et al. 2010]:

1. **Mínimo *Overhead* no serviço de descoberta:** como a maioria dos OI do mundo real são dispositivos com baixo poder computacional, existe uma necessidade de utilização de paradigmas de descoberta mais leves;
2. **Mínimo esforço no registro de OI:** um OI deve ser capaz de anunciar seus serviços a um servidor de registros através da rede. Esse processo precisa ser realizado sem nenhuma intervenção humana. A quantidade de informações necessárias para esse registro também deve ser bem reduzida;
3. **Suporte à busca contextual e dinâmica de OI:** os algoritmos de busca devem ir além de uma simples busca por palavras-chave. Eles devem levar em consideração dados dinâmicos do contexto do usuário, como localização;
4. **Suporte à alocação de OI sob demanda:** os serviços ofertados pelos OI devem ser ativados sob demanda, evitando assim a má utilização dos recursos.

2.3. CoAP

O CoAP oferece suporte a uma comunicação entre aplicações e OI seguindo o paradigma requisição/resposta. Esse protocolo possui um serviço de descoberta já implementado, baseado no conceito de diretórios de OI. Sua estrutura é baseada no HTTP, facilitando assim a integração com os recursos disponíveis na WEB. Entretanto, diferente do HTTP, o CoAP cumpre alguns requisitos específicos para dispositivos com limitações computacionais, como o baixo *overhead* na troca de mensagens. Clientes utilizam servidores CoAP para acessar uma lista de OI mantida por eles. Cada OI é representado por uma URI, seguindo o formato especificado pelo *Constrained RESTful Environments*[3] (CoRE). A descoberta pode ser realizada com a utilização de filtros, que são parâmetros adicionados a *string* de consulta, enviada ao respectivo diretório de recursos.

[3]CoRE: https://tools.ietf.org/html/rfc6690

O CoRE também define um conjunto de atributos que representam os recursos presentes em um diretório. Os principais atributos são: *Resource Type*, que é o responsável por identificar a função de um determinado recurso (temperatura, luminosidade, impressora, etc.); *Interface Description*, que indica os métodos que podem ser utilizados para a comunicação com esse recurso (*GET, POST*, etc); e *Context Type*, que representa o formato dos dados fornecidos pelo recurso. O CoAP já possui inúmeras implementações disponíveis, para as mais diversas plataformas[4], isso é um bom indicativo que existem inúmeras solução que são compatíveis com essa especificação. Com base nisso, criar soluções que também utilizem o CoAP pode garantir a interoperabilidade com essa gama imensa de dispositivos já ativos. Embora essa interoperabilidade não possa ainda ser estendida para IoT em geral, garantir a compatibilidade com o CoAP é uma estratégia que potencializa a interoperabilidade da proposta deste trabalho.

3. CoAP-CTX

CoAP-CTX CoAP ConTeXtual) é uma extensão do serviço de descoberta do CoAP que visa atender os requisitos de descoberta contextual e alocação de objetos inteligentes sob demanda (Vide Seção 2.2). O CoAP-CTX segue um processo de descoberta sensível ao contexto de objetos inteligentes que pode ser dividido em 8 etapas. (1) Aquisição do contexto do usuário, informações contextuais essas que são a entrada do processo; (2) Inferência dos objetos inteligentes de interesse; (3) Representação e identificação dos OI de interesse; (4) Criação de *strings* de consulta CoAP que realizam uma pré-filtragem; (5) Busca por OI que satisfaçam os filtros da consulta CoAP; (6) Aquisição da lista de objetos inteligentes disponíveis no ambiente e que são de interesse do usuário; (7) Representação dos OI relevantes e disponíveis; (8) Listagem dos OI selecionados, que são a saída do processo. Todas essas etapas podem ser visualizadas na Figura 2.

Na primeira etapa (1), as informações que descrevem o contexto atual do usuário (e.g., localização, situação, atividade, etc.) são obtidas (2) e utilizadas para a identificação de quais OI são de interesse do usuário em um dado momento. A Figura 2 apresenta um interesse que inclui os OI com as seguintes funcionalidades: Controle de acesso (controle das portas); Horário atual; Alarme sonoro; Dispositivo móvel; Impressora. (3) Cada OI é representado por meio de uma única *String* que agrega informações contextuais e que pode ser utilizada para identificação desse dispositivo. Como os OI utilizam o CoAP como protocolo de comunicação e interação, pode-se utilizar mecanismos já disponíveis nesse protocolo, como as *strings* de consulta CoAP, geradas na etapa (4). Nesse caso, um exemplo de consulta ao servidor CoAP seria: *GET /.well-known/core?rt="access-control time alarm mobile-device printer"*. A referência aos OI que se registraram no servidor CoAP (5) e que atenderam às restrições expressas na consulta realizada é retornada ao serviço de descoberta (6). Antes de gerar a lista final de objetos relevantes e disponíveis (7), é preciso verificar as informações dinâmicas do contexto do usuário e dos OI, como localização. Por fim, essa lista final é apresentada ao usuário (8), que por sua vez poderá interagir com esses OI e realizar operações do cotidiano mais facilmente.

A solução proposta foi desenvolvida para atuar em ambientes inteligentes que possuam três tipos de elementos básicos: um *smartphone* Android, um ou mais servidores CoAP e um conjunto de objetos inteligentes. O *smartphone* funciona como o elemento

[4]http://coap.technology/impls.html

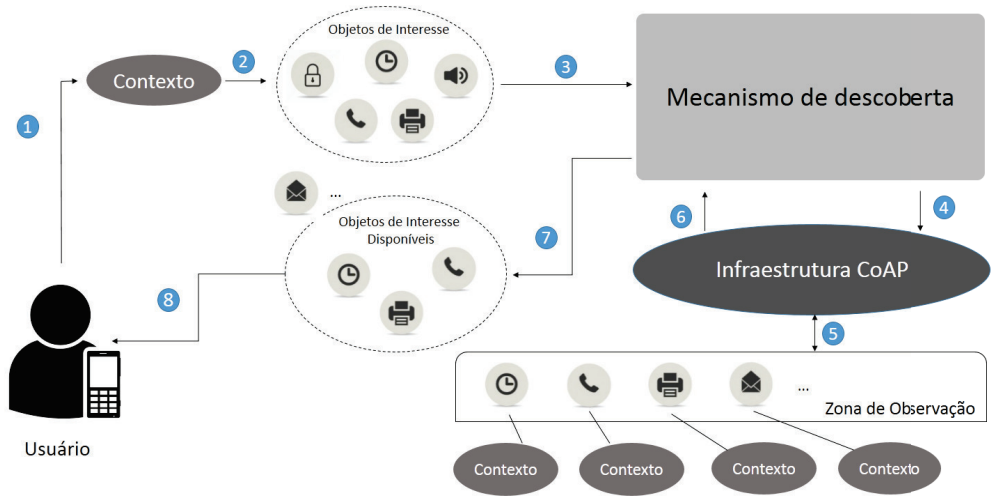

Figura 2. Visão geral do processo de descoberta de objetos inteligentes

coordenador de todo o processo de descoberta, gerenciando assim todos os OI que são de interesse do usuário. A Figura 3 apresenta a arquitetura do CoAP-CTX.

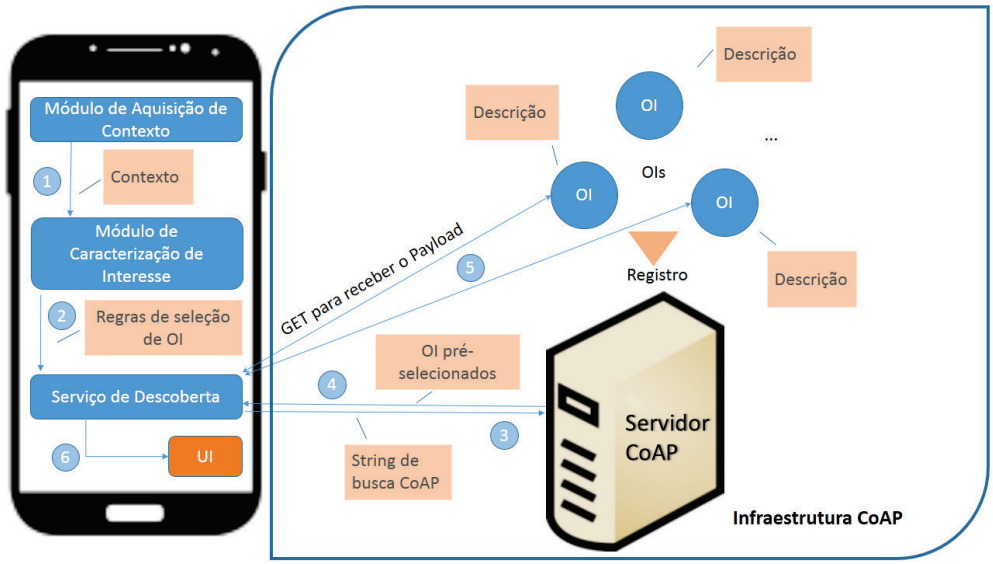

Figura 3. Arquitetura do CoAP-CTX

A primeira tarefa do sistema é obter o contexto (1) por meio do Módulo de Aquisição de Contexto. Esse módulo é responsável pela detecção da situação (e.g., usuário acordou) e fornecimento das informações contextuais suficientes para caracterizar seu interesse naquele momento. Com base nessas informações contextuais, o Módulo de Caracterização de Interesse constrói uma forma simples de representação (2), uma lista de *Strings* contendo características separadas por ”.”, de forma hierárquica. Por exemplo, se o usuário tem interesse em acessar a TV do quarto, esse interesse é representado por *control.ambient.tv* e *context.ambient.quarto*. Esse módulo foi desenvolvido de modo a permitir futuras extensões, com abordagens de caracterização do interesse mais complexas, como *machine learning*.

Com essa representação do interesse, o Serviço de Descoberta pode realizar um mapeamento entre interesse e *String* de consulta CoAP (3). Cada campo das *Strings* de interesse é mapeado para campos que definem e descrevem os objetos inteligentes que seguem a especificação CoRE. Os campos *control e ambient* dizem que o objeto deve ser capaz de atuar no ambiente, o que pode ser representado por um filtro no campo *Interface-Description* do OI, limitando seu valor para apenas *actuator*. Já o campo *tv* da *String* de interesse, representa o tipo, ou classe, do objeto inteligente, e pode ser mapeado para o campo *Resource-Type*, aceitando apenas OI que possuam esse atributo igual a *tv*. Com isso é possível gerar a requisição para o Servidor CoAP em busca de todas as TVs que podem ser controladas pelo usuário.

O Servidor CoAP mantém o registro de todos os OI disponíveis no ambiente, isso inclui até os que não fazem parte do interesse do usuário naquele momento. Ao receber a requisição feita em busca dos OI, o servidor aplica os filtros sobre os campos *Resource-Type* e *Interface-Description* e retorna a lista de OI (4) que atendem à requisição. Pode ocorrer de vários OI serem retornados, por exemplo as TVs do quarto e da sala. Não é possível realizar uma consulta para identificar, ainda no Servidor CoAP, qual é a televisão do quarto especificamente. Isso porque a informação contextual do OI, que é dinâmica, não fica disponível no serviço de descoberta do Servidor CoAP. Para ter acesso a essas informações, clientes CoAP devem realizar uma operação de GET (5) nesses OI e receber seu contexto no *Payload* da mensagem.

O Serviço de Descoberta, ao receber a lista com os OI pré-selecionados pelo servidor, gera uma requisição GET para cada um deles. Com base nas informações contidas no *Payload* da mensagem, é possível aplicar as regras de seleção que envolvem características dinâmicas, como localização. A referência dos OI selecionados é por fim repassada para o gerenciador da interface gráfica da aplicação (6), que é responsável por exibir na tela do *smartphone* os OI selecionados, além de permitir que o usuário os acesse. Todos os outros OI que estão registrados no servidor CoAP são postos em modo de espera, até que o contexto mude e uma nova descoberta aconteça.

3.1. Prova de Conceito

Uma primeira prova de conceito (PoC) foi concebida e implementada como objetivo de ilustrar uma descoberta seletiva de OI, com base no contexto. Para essa PoC, foram implementadas uma aplicação Android, um servidor CoAP baseado na implementação jCoAP[5] e quatro OI por meio da plataforma Arduino (duas TVs, um Ar Condicionado, e um sensor de luminosidade). O contexto utilizado foi o da chegada do usuário na sua casa. A localização, obtida pelo GPS do *smartphone*, detecta o evento da chegada, enquanto que as informações de histórico sugerem que o usuário tem interesse nos OI que controlam o estado da casa, ou seja, atuadores. O filtro aplicado para esse contexto foi no campo *Interface-Description*, selecionando apenas os atuadores, no caso, três dentre os quatro OI. A aplicação Android lista os OI que foram selecionados após a aplicação das regras de seleção (OI colocalizados com o usuário e que funcionam como atuadores), além de permitir que o usuário acesse cada um dos OI selecionados. A Figura 4 apresenta *prints* da tela da aplicação executando. Durante o carregamento da aplicação (a), são realizadas as consultas ao servidor CoAP e aplicadas as regras de seleção. Em (b), são listados os OI

[5]http://www.ws4d.org/ws4d-jcoap/

que satisfizeram as regras de seleção contextual. Já as telas (c) e (d) mostram a interface de controle da TV e do Ar Condicionado (AC), respectivamente.

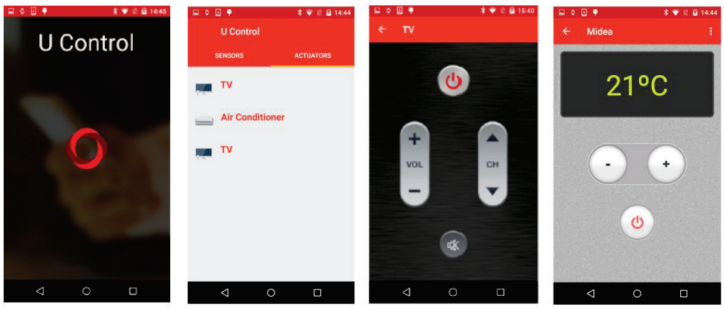

Figura 4. Aplicação Android desenvolvida como prova de conceito.

Para a implementação dos OI, foram utilizados três Arduinos Mega e um Arduino Yún, com interfaces de comunicação *Bluetooth* e *WiFi*, respectivamente. A Figura 5 apresenta os hardwares utilizados para a PoC. Em (a) está o Arduino Yún com o sensor de luminosidade. Em (b) os componentes para controlar o Ar Condicionado. Em (c) são exibidos os componentes utilizados para controlar as TVs (replicação do hardware). O controle das TVs e do Ar Condicionado foi feito utilizando infravermelho (IR), com os códigos IR de cada comando obtidos por engenharia reversa.

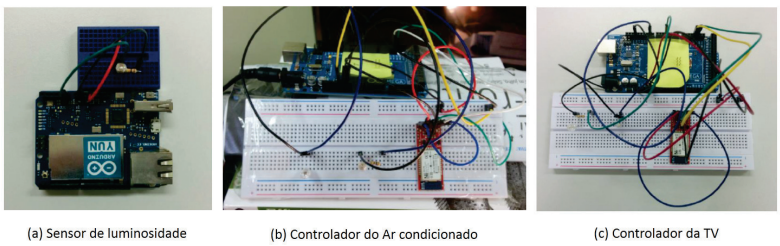

Figura 5. Hardware utilizado na prova de conceito.

4. Avaliação

Esta seção descreve os experimentos realizados para avaliação do CoAP-CTX em três diferentes ambientes inteligentes: casa, carro e prédio.

4.1. Objetivo da Avaliação

A prova de conceito (Vide Seção 3.1) mostrou que o CoAP-CTX pode ser utilizado para reduzir o número de OI apresentados ao usuário, diminuindo a sobrecarga visual que um grande número de OI poderiam causar. Além disso, espera-se que a proposta diminua o número total de mensagens trocadas na rede, no caso em que OI possam entrar em modo de espera. Como o número de mensagens trocadas é proporcional ao consumo de energia, reduzir o primero também tem por objetivo reduzir o consumo energético da rede como um todo. Além disso, como está sendo adicionado uma camada de complexidade acima do serviço de descoberta padrão do CoAP, um aumento do tempo de descoberta

final também é previsto. Logo, o objetivo da avaliação é verificar o comportamento do CoAP-CTX com relação ao número de mensagens trocadas na rede e o tempo gasto para realizar a descoberta de OI.

4.2. Materiais e Métodos

Para essa avaliação, foi utilizado o simulador Cooja[6], que é voltado especificamente para redes de sensores sem fio. O Cooja foi executado por meio de uma máquina virtual Linux, disponibilizada pelos próprios desenvolvedores do simulador. Todos os componentes foram implementados e simulados para a plataforma de *hardware Tmote Sky*[7]. Essa plataforma possui um baixo tempo de transição para sair do modo de espera, além disso, utiliza o módulo CC2420[8] para comunicação sem fio, que é compatível com o padrão IEEE 802.15.4. O Cliente CoAP realiza a descoberta a cada 10 segundos, enquanto os OI que não estão em modo de espera enviam seu estado a cada 200*ms* para o servidor CoAP.

4.3. Procedimento

Os resultados dos experimentos dependem diretamente do contexto do usuário e dos OI, podendo sofrer grandes variações para contextos diferentes. Como forma de contornar esse problema, foram analisados três casos específicos para cada ambiente: o melhor e o pior caso, e um intermediário. Consideramos como melhor caso quando apenas um OI é pré-selecionado e não há necessidade de consulta por informações dinâmicas no *payload* da mensagem desse OI específico. Já o pior caso ocorre quando todos os OI são pré-selecionados e há necessidade da consulta individual, resultando em uma seleção final de todos os OI. O caso intermediário é analisado separadamente para cada cenário simulado, utilizando valores empíricos para a definição do número de OI. Foi então analisada a performance da descoberta usando diretamente o CoAP (sem contexto), e depois usando o CoAP-CTX (com contexto).

Cada ambiente é composto por um cliente CoAP, que representa o *smartphone*, um ou mais servidores CoAP, e um conjunto de OI que se registram nesses servidores. A Figura 6 apresenta a topologia de cada rede gerada. Para cada caso de cada ambiente foram realizadas 10 simulações de 15 minutos, os resultados são apresentados na forma de *box plots*, permitindo a visualização dos valores médios, variância e limites de *outliers*. Foi utilizado apenas um cliente CoAP em cada simulação porque esse cliente representa um único usuário interagindo com o ambiente inteligente em questão. Múltiplos usuários simultâneos trazem novos requisitos que estão fora do escopo deste trabalho, como controle de acesso.

4.4. Casa Inteligente

Casas inteligentes geralmente possuem um número moderado de OI, para a simulação foram utilizados 20 OI, 1 Servidor CoAP e 1 Cliente CoAP. Para o caso intermediário, supôs-se que o interesse do usuário é mais genérico, fazendo assim com que a descoberta retorne um grande número de OI. Nesse caso, 15 OI são pré-selecionados pelas *Strings* de consulta CoAP, desses, 5 são descartados após a aplicação das regras de seleção com informações dinâmicas, totalizando 10 OI ativos e 10 em modo de espera. As Figuras 7

[6]http://anrg.usc.edu/contiki/index.php/Cooja_Simulator

[7]http://wirelesssensornetworks.weebly.com/1/post/2013/08/tmote-sky.html

[8]http://www.ti.com/product/CC2420

Figura 6. Redes criadas com o Cooja para avaliar os cenários propostos

(a) e (b) mostram os resultados encontrados para esse ambiente, onde é possível observar que uma redução de cerca de 50% do número de OI descobertos gerou uma redução de aproximadamente 41% no número total de mensagens. Já o tempo de descoberta aumentou em 100*ms*, também no caso intermediário, com relação ao CoAP padrão.

4.5. Carro Inteligente

Carros possuem um número de OI mais reduzido, por isso foram utilizados 10 OI, 1 Servidor CoAP e 1 Cliente CoAP. O cenário de utilização dos OI do carro também são mais específicos. Enquanto dirige, o usuário tende a ter mais interesse em OI relacionados a navegação, já os OI de multimídia ganham prioridade quando o carro está parado. Com base nisso, definiu-se o caso intermediário como sendo 4 OI pré-selecionados pelo servidor CoAP, dentre esses 3 se tornam disponíveis ao usuário, totalizando 7 OI em modo de espera. Os resultados, apresentados nas Figuras 7 (c) e (d), mostram que, para o caso intermediário, o CoAP-CTX obteve uma redução de cerca de 57% no número de mensagens, com um aumento no tempo de descoberta médio de apenas 63*ms*.

4.6. Prédio Inteligente

Um prédio inteligente é uma estrutura física que possui uma elevada densidade de OI disponíveis, sendo esses de propósito mais geral, para atender vários usuários. Para esse cenário, utilizou-se um total de 100 OI, 3 Servidores CoAP e um Cliente CoAP. Nesse cenário, informações de contexto dinâmicas, como localização *indoor*, são fundamentais para limitar os resultados da descoberta. Para o caso intermediário, considerou-se que 70 OI seriam pré-selecionados, mas que apenas 15 seriam de interesse do usuário após a obtenção das informações contextuais dinâmicas. As Figuras 7 (e) e (f) mostram os resultados obtidos. Houve uma drástica redução no número de mensagens trocadas na rede quando utilizada a solução proposta, cerca de 80%, para o caso intermediário. Entretanto, para esse caso, o tempo de descoberta médio ficou acima de 1,5 segundos.

4.7. Discussão

Os experimentos mostraram que, em geral, quando comparado ao serviço de descoberta padrão do CoAP, o CoAP-CTX apresenta uma redução no número total de mensagens trocadas na rede, em troca de um aumento no tempo necessário para a realização da descoberta. Porém, esse comportamento se modifica quando as informações contextuais levam

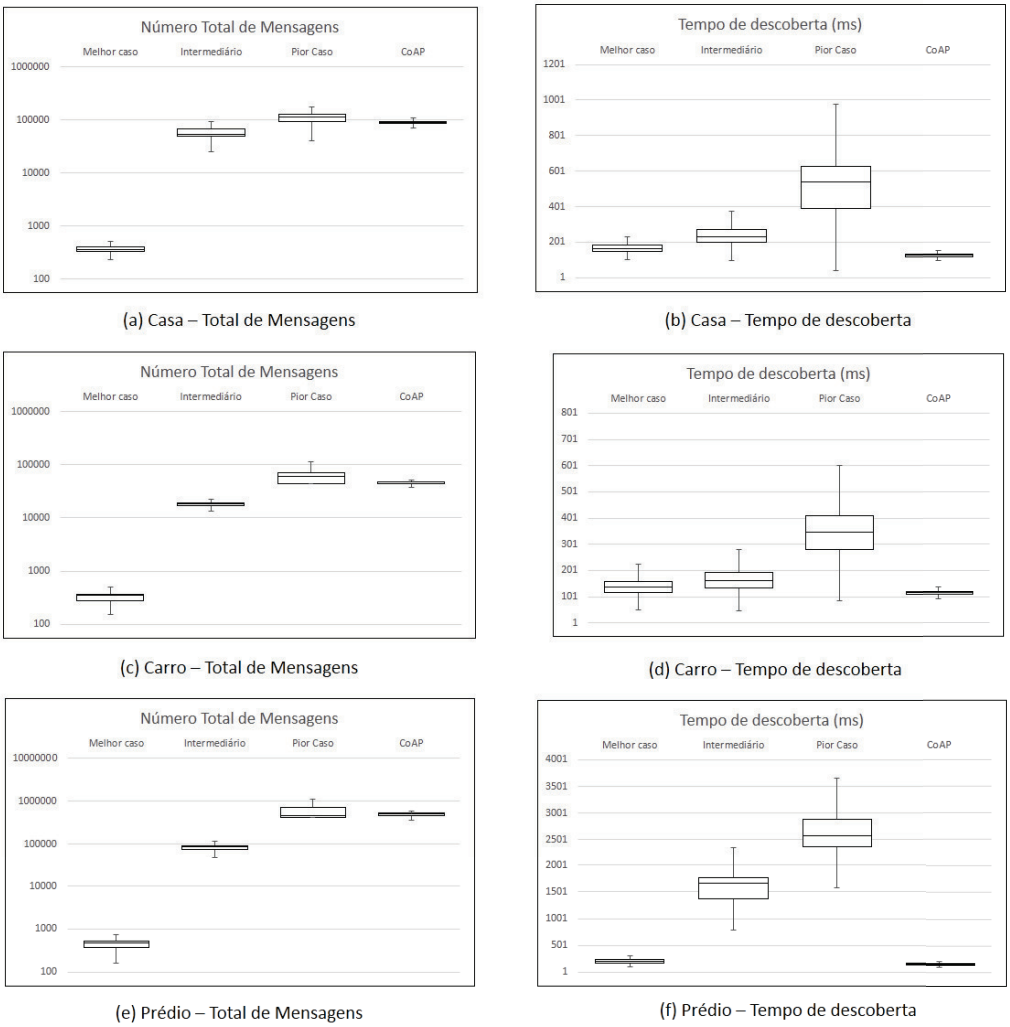

Figura 7. Resultados obtidos para os três ambientes inteligentes: casa, carro e prédio

a um cenário próximo do pior caso, onde o número de mensagens tende a se aproximar dos valores encontrados para o CoAP padrão, podendo até aumentar um pouco. Esse aumento será pequeno pois é devido ao envio de requisições para obtenção de informações contextuais dinâmicas, que representa um percentual bem menor quando comparado a mensagens de atualização dos OI.

Notou-se que o número total de mensagens depende fortemente do número final de OI em modo de espera, enquanto que o tempo de descoberta é afetado pela quantidade de OI com informações dinâmicas a serem obtidas para a aplicação das regras de seleção. Portanto, conclui-se que em cenários onde os OI tem uma função mais especializada, ou seja, em que seja possível pré-selecionar um conjunto próximo do final de OI de interesse, o aumento no tempo de descoberta é baixo. Já em em ambientes genéricos, como prédios inteligentes, onde a maior parte da seleção ocorre com base nas informações contextuais dinâmicas, o tempo de descoberta aumenta consideravelmente. Uma solução para esse aumento seria a migração das regras contextuais dinâmicas para o próprio servidor CoAP,

pois assim todo o processo de descoberta seria realizado com uma única requisição ao servidor CoAP. Entretanto, essa migração faria com que a solução perdesse parte da interoperabilidade que o CoAP proporciona, pois ambientes que já possuem servidores CoAP tradicionais não seriam compatíveis com a aplicação de regras dinâmicas.

5. Trabalhos Relacionados

Nesta seção são discutidos os trabalhos relacionados ao tema de descoberta de OI em um cenário de IoT. Foi dada preferência a trabalhos que propõem algum tipo de serviço ou infraestrutura de descoberta de OI.

[Liu et al. 2013] propõem uma arquitetura de descoberta distribuída de dispositivos focada em IoT. Cada dispositivo é considerado um nó de uma rede P2P e é capaz de realizar o processo de registro, bem como ajudar no serviço de descoberta. A identificação de cada dispositivo é feita utilizando URIs que seguem o modelo CoAP. Por trazerem uma abordagem totalmente distribuída, os autores eliminam os possíveis problemas de um elemento centralizador causados pela mobilidade e volatilidade, intrínsecas ao cenário de IoT. Por outro lado, como cada dispositivo deve ser capaz de realizar o registro, isso pode ser inviável para dispositivos estritamente limitados computacionalmente. Já o Digcovery é uma proposta de descoberta global de OI que usa uma infraestrutura centralizada na qual OI podem se registrar [Jara et al. 2013]. Para o acesso a essa infraestrutura, foi desenvolvido um serviço para dispositivos móveis, que permite a descoberta e o acesso aos objetos inteligentes. Características contextuais de geo-localização são levadas em consideração durante a fase de descoberta, coordenada pelo *smartphone* do usuário.

[Cirani et al. 2014] apresentam uma arquitetura auto-configurável e escalável de descoberta de serviços. No trabalho, é proposta uma topologia P2P com a utilização de diversos *gateways*. Eles são responsáveis por manter a lista de OI presentes na rede. Esses *gateways* são baseados nos servidores CoAP cujos clientes podem realizar uma operação GET para a URI */.well-know/core* a fim de receber a lista de dispositivos disponíveis. DiscoWoT é um serviço de descoberta de OI proposto por [Mayer and Guinard 2011], e que é possível definir estratégias de descoberta em tempo de execução por meio de interfaces RESTful[9]. Esse serviço de descoberta se utiliza de *Microformats* e *Microdata* em conjunto com outras tecnologias WEB para representar semanticamente os objetos inteligentes. A implementação dessa representação é feita utilizando JSON, para garantir a interoperabilidade. [Ishaq et al. 2012] propõem um mecanismo de descoberta baseado em CoAP e DNS. Traduções entre CoAP e HTTP são disponibilizadas, permitindo a descoberta de qualquer objeto inteligente compatível com o padrão *IPv6*.

A Tabela 1 apresenta um comparativo entre esses trabalhos relacionados. Os critérios utilizados na classificação foram escolhidos com base nos requisitos da descoberta de OI. Nota-se que a sensibilidade ao contexto ainda não é amplamente utilizada nas soluções de descoberta de OI para IoT. Apenas algumas das soluções utilizam elementos contextuais e nenhuma combinou o contexto do usuário e do OI em um único serviço. Nota-se também uma certa tendência de utilização do CoAP como base para a descoberta, o que pode ser justificado tanto pela interoperabilidade quanto pelo fato desse protocolo já possuir um serviço de descoberta, que pode ser reutilizado e estendido.

[9]RESTful *web services* são aqueles que seguem o modelo arquitetural REST.

Tabela 1. Comparativo entre os trabalhos relacionados.

Trabalhos Relacionados	CoAP	Elementos Contextuais	Tipo de Contexto
Liu et al.	✓	-	-
Digcovery	-	✓	Usuário
Cirani et al.	✓	-	-
DiscoWoT	-	✓	Objeto Inteligente
Ishaq et al.	✓	-	-
CoAP-CTX	✓	✓	Usuário + OI

6. Considerações Finais

Este trabalho apresenta uma proposta de extensão do CoAP para a descoberta sensível ao contexto de objetos inteligentes. São utilizadas informações contextuais do usuário e dos OI para realizar uma seleção mais direcionada, bem como reduzir o número de mensagens trocadas pelos dispositivos, otimizando assim o consumo de energia. A criação de uma prova de conceito mostrou que a solução é factível e que pode ser utilizada em um cenário real. A avaliação da solução por meio de um simulador garante que, para cenários próximos aos simulados, a utilização do CoAP-CTX reduz o número total de mensagens na rede, reduzindo assim o consumo de energia em geral. Além disso, o aumento do tempo de descoberta é aceitável em cenários onde o interesse do usuário não mude tão rapidamente, característica essa que é comumente atingida.

Como próximos passos desta pesquisa, notou-se a necessidade de tornar a solução mais genérica, permitindo que desenvolvedores estendam os mecanismos de aquisição de contexto e regras de seleção. Logo, o objetivo futuro é transformar o CoAP-CTX em um *framework* para descoberta de OI. Além disso, espera-se realizar mais avaliações, sobretudo para analisar se a lista final de OI descobertos condiz com o real interesse do usuário. A solução atual não oferece suporte a múltiplos usuários, sendo necessária a implementação de uma política de controle de acesso para permitir vários usuários interagindo com os OI ao mesmo tempo. Por fim, para reduzir o tempo de descoberta em ambientes com muitos OI genéricos, pode ser implementado um mecanismo de verificação do servidor CoAP. Logo, as regras de contexto dinâmico podem ser aplicadas por servidores com suporte a essa funcionalidade, ou, caso contrário, executadas no próprio *smartphone*.

Agradecimentos

Gostaríamos de agradecer à CAPES pela concessão da bolsa de apoio à Pós-graduação, que permitiu a realização da pesquisa de mestrado na qual este trabalho foi originado.

Referências

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

Cirani, S., Davoli, L., Ferrari, G., Léone, R., Medagliani, P., Picone, M., and Veltri, L. (2014). A scalable and self-configuring architecture for service discovery in the internet of things. *IEEE Internet of Things Journal*, 1(5):508–521.

Cisco (2016). Cisco global cloud index: Forecast and methodology, 2015-2020. *CISCO white paper*.

Crasso, M., Zunino, A., and Campo, M. (2008). Easy web service discovery: A query-by-example approach. *Science of Computer Programming*, 71(2):144 – 164.

Dave, E. (2011). The internet of things: how the next evolution of the internet is changing everything. *CISCO white paper*.

Dey, A. K., Abowd, G. D., and Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Hum.-Comput. Interact.*, 16(2):97–166.

Duarte, P. A., Gomes, F. A., Barreto, F. M., Viana, W., and Trinta, F. M. (2014). Loccamconfigurator: Uma ferramenta para modelagem visual de configurações de um middleware de suporte à aplicações conscientes de contexto. In *Proceedings of the 20st Brazilian Symposium on Multimedia and the Web*, WebMedia '14.

Duarte, P. A., Silva, L. F. M., Gomes, F. A., Viana, W., and Trinta, F. M. (2015). Dynamic deployment for context-aware multimedia environments. In *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*, WebMedia '15, pages 197–204, New York, NY, USA. ACM.

Guinard, D., Trifa, V., Karnouskos, S., Spiess, P., and Savio, D. (2010). Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services. *IEEE Transactions on Services Computing*, 3(3):223–235.

Ishaq, I., Hoebeke, J., Rossey, J., Poorter, E. D., Moerman, I., and Demeester, P. (2012). Facilitating sensor deployment, discovery and resource access using embedded web services. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pages 717–724.

Jara, A. J., Lopez, P., Fernandez, D., Castillo, J. F., Zamora, M. A., and Skarmeta, A. F. (2013). Mobile digcovery: A global service discovery for the internet of things. In *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, pages 1325–1330.

Liu, M., Leppänen, T., Harjula, E., Ou, Z., Ylianttila, M., and Ojala, T. (2013). Distributed resource discovery in the machine-to-machine applications. In *2013 IEEE 10th International Conference on Mobile Ad-Hoc and Sensor Systems*, pages 411–412.

Mayer, S. and Guinard, D. (2011). An extensible discovery service for smart things. In *Proceedings of the Second International Workshop on Web of Things*, WoT '11, pages 7:1–7:6, New York, NY, USA. ACM.

Ríos, L., Endler, M., and Colcher, S. (2016). An energy-aware iot gateway, with continuous processing of sensor data. XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC2016, Salvador, Brasil.

Tarkoma, S., Siekknen, M., Lagerspetz, E., and Xiao, Y. (2014). Smartphone energy consumption: modeling and optimization.

Viana, W., Miron, A. D., Moisuc, B., Gensel, J., Villanova-Oliver, M., and Martin, H. (2011). Towards the semantic and context-aware management of mobile multimedia. *Multimedia Tools Appl.*, 53(2):391–429.