Validação de arcabouço de WoT para uma assistente virtual*

Miguel de Mello Carpi[1], **Antonio Deusany de Carvalho Junior**[1], **Alfredo Goldman**[1]

[1]Instituto de Matemática e Estatística - IME - USP
Rua do Matão, 1010 – CEP – São Paulo – SP – Brazil

miguelmello@usp.br,{dj,gold}@ime.usp.br

Abstract. *The A.D.A. project (Advanced Distributed Assistant) aims to develop a personal assistant able to interact with the user by means of voice commands and execute them on different IoT devices. However, due to the heterogeneity of Internet of Things (IoT) devices, each with its own technologies and communication protocols, the task of managing several of these devices becomes difficult. In order to solve this problem, several frameworks have been proposed and studied over time. In this paper, we validate the integration of a Web of Things (WoT) framework in A.D.A., showing how it deals with the heterogeneity problem and how we can use it.*

Resumo. *O projeto A.D.A.*[1] *(Assistente Distribuída Avançada) tem como objetivo desenvolver uma assistente pessoal capaz de interagir com o usuário por meio de comandos por voz e executá-los em diferentes dispositivos da Internet das Coisas (IoT). Entretanto, devido à heterogeneidade de dispositivos IoT, cada um com seus meios e protocolos de comunicação próprios, a tarefa de gerenciá-los se torna difícil. Visando resolver ou amenizar esse problema, vários arcabouços (frameworks) foram propostos e estudados ao longo do tempo. Neste trabalho, validamos a integração do framework Web of Things (WoT) na A.D.A., mostrando como ele lida com o problema de heterogeneidade e como podemos utilizá-lo.*

1. Introdução

Um assistente comandado por voz é um agente artificial virtual capaz de reconhecer comandos verbais, extrair seu significado e, a partir de uma representação de conhecimento expressa por ações pré programadas, executar uma ou mais tarefas pretendidas pelo usuário. Deste modo, o assistente realiza, na prática, o papel de uma camada intermediária simplificadora e compatível com o modelo mental do usuário, desobrigando-o de interagir com inúmeras interfaces distintas e com as mudanças de contexto envolvidas.

Neste contexto, o projeto A.D.A. (Assistente Distribuída Avançada) visa a criação de um agente virtual capaz de interagir com o usuário a partir de um ecossistema de dispositivos, como os da Internet das Coisas (IoT, do inglês *Internet of Things*), por meio de comandos de voz em português. Para cumprir o seu objetivo, a assistente precisa gerenciar e interagir adequadamente com os dispositivos que integram o ecossistema do usuário, o que demanda interoperabilidade de interfaces de programação de aplicações e escalabilidade de processos.

*Este projeto foi financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e pelo grupo de extensão USPCodeLab (uclab.xyz/site).

[1]Site do projeto A.D.A.: uclab.xyz/ada

Neste trabalho, discutiremos a integração de um *framework* para o gerenciamento de dispositivos IoT a uma assistente virtual. Analisamos como uma interface flexível e padronizada para a comunicação com dispositivos auxilia em seu gerenciamento e pode resolver problemas encontrados no desenvolvimento do interpretador. Ao final, descrevemos outras capacidades dessa abordagem em ambientes diferentes que serão exploradas.

2. Referencial teórico

Um dos grandes desafios da IoT está relacionado à descoberta de dispositivos, locais ou remotos, tanto por usuários quanto por outros dispositivos inteligentes [Bröring et al. 2016]. Esse desafio surge devido à grande quantidade e diversidade de dispositivos necessária para atender os mais diversos e variados casos de uso bem como fabricantes para produzi-los, o que resulta em dispositivos com tecnologias e protocolos de comunicação diferentes [Alam and Noll 2010]. Esse problema afeta diretamente a parte de gerenciamento de dispositivos da A.D.A. [Freire et al. 2020], que precisa registrar todos os dispositivos para poder se comunicar com eles e assim atender aos comandos do usuário. Neste caso, aarquitetura baseada em diretórios, como apresentada por Bröring, torna-se apropriada.

Para mitigar o problema dos protocolos de comunicação uma saída é utilizar alguma API (do inglês *Application Programming Interface*) como interface de comunicação para qualquer dispositivo. Essas funcionalidades estão bem descritas na literatura, como pode ser visto no *framewok* proposto por [Alam and Noll 2010], na arquitetura apresentada em [Gupta et al. 2011], e estão implementadas no WebThings Gateway[2], sendo este último o foco deste trabalho. O WebThings Gateway é um exemplo de como um *framework* e arquitetura, semelhantes aos modelos citados acima, pode ser implementando. O *gateway* em questão permite: conectar dispositivos que seguem a API WoT[3], controlá-los através de uma interface Web, automatizá-los através de um sistema de regras; e ainda estender/adicionar funcionalidades ao sistema através de *add-ons*. Consequentemente, a fim de validar o uso deste *framework* no projeto A.D.A., realizamos testes unindo o WebThings Gateway e o WebThings Framework. Este último é um conjunto de programas que têm como finalidade prover comunicação de dispositivos inteligentes pela API WoT.

3. Metodologia

Como o WebThings Gateway está disponível em várias plataformas, decidimos testar a versão para Linux e a imagem Docker . Esses testes tinham como intuito avaliar a usabilidade, desempenho, funções disponíveis em cada plataforma e possível integração com o ecossistema da A.D.A.. Em cada uma das plataformas, levamos em conta o tempo gasto e a dificuldade para conseguir instalar os *softwares* necessários e executar o *gateway*.

Partindo de um código exemplo disponível no repositório do WebThings Framework e utilizando um microcontrolador ESP32 junto com o ambiente Arduino, prototipamos uma lâmpada compatível com a WoT API. Como o código exemplo tinha sido desenvolvido para o microcontrolador ESP8266, alguns ajustes foram feitos para adequá-lo e garantir o seu funcionamento no ESP32. Por fim, a lâmpada foi usada para testar a interface via *gateway* e via HTTP.

[2]WebThings Gateway: `https://webthings.io/`

[3]WebThings WoT API: `https://iot.mozilla.org/wot/`

Para podermos utilizar a lâmpada inteligente com o WebThing Gateway foi necessário instalar um *add-on* (extensão). A instalação pode ser feita através da interface gráfica. No caso, utilizamos o *thing-url-adapter*[4], responsável por descobrir o dispositivo na rede, estabelecer conexão com ele de forma que o usuário, através da interface gráfica, consiga acessar os dados e serviços oferecidos. Caso o dispositivo não seja descoberto automaticamente pelo *add-on*, é possível adicioná-lo manualmente, a partir da inserção da URL do dispositivo em questão. Já para a comunicação direta com a lâmpada, através da API HTTP, utilizamos a ferramenta *Curl*[5] disponível no Linux. Uma representação da arquitetura final pode ser vista na Figura 1.

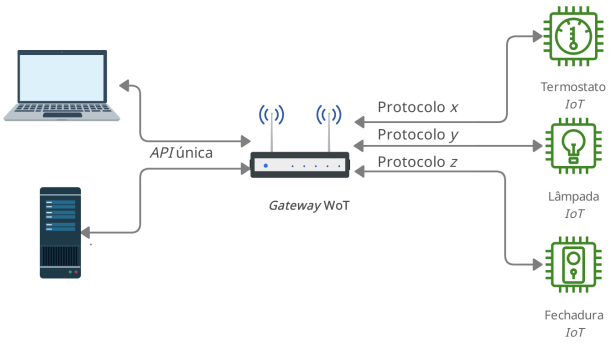

Figura 1. Arquitetura de rede com o WebThing Gateway

4. Resultados e discussão

Tanto a versão para Linux quanto a imagem Docker do WebThing Gateway apresentam as mesmas funcionalidades e não percebemos diferenças no desempenho. Desse modo a imagem Docker é vantajosa, pois requer menos passos para ser instalada e configurada e, além disso, a implementação da A.D.A. já é baseada no uso de contêineres Docker para sua execução distribuída. [Freire et al. 2020].

Na utilização do *add-on* (extensão) para interconectar a lâmpada ao sistema verificou-se que o desenvolvimento da extensão pode não acompanhar o da aplicação principal (*gateway*) por ser muito dependente das especificações atuais, o que pode vir a ser um problema futuro. Além disso, ao desconectar a lâmpada da rede e reconectá-la, o *gateway* podia acabar não a reconhecendo como a mesma lâmpada, pois o *gateway* utiliza por padrão o endereço IP como identificador de dispositivo e o endereço IP da lâmpada é dinâmico, isto é, pode variar. No entanto, através do próprio WebThings Framework foi possível definir um UUID (do inglês *Universally Unique Identifier*) para o dispositivo e assim evitar que o *gateway* usasse o endereço IP como identificador.

No caso da WoT API, realizamos um teste de qualidade de experiência: um com a interface gráfica do *gateway* e o outro através da API HTTP. No caso da interface gráfica, avaliamos a facilidade de manuseio desta e se as ações feitas através dela resultavam no comportamento esperado. No caso da API HTTP o teste consistiu no envio de 30 requisições, com intervalo de um segundo entre uma e outra, alternando entre comandos

[4]WebThings Url Adapter: `https://github.com/WebThingsIO/thing-url-adapter`

[5]*Curl* é uma ferramenta de linha de comando para transferir e obter dados em diversos protocolos. A ferramenta, geralmente, está disponível nativamente na maioria das plataformas Unix

para acender e apagar a lâmpada. Em ambos os testes, os comandos enviados foram executados corretamente. Isso demonstra que a interface gráfica facilita o manuseio de dispositivos e que o *framework* não precisa ser utilizado junto com o WebThings Gateway, isto é, podemos nos comunicar com os dispositivos WoT diretamente.

5. Conclusão

Ao final dos testes, concordamos que o WebThings Gateway é um bom exemplo de como um *framework* pode ser utilizado para separar a parte de conexão de dispositivos IoT dos serviços que eles oferecem. Com essa separação feita, a realização de operações com os dispositivos como coletar dados, conectá-los entre si e acionar atuadores se torna única e pode ser automatizada facilmente. Além disso, sua API é padronizada, bem documentada e faz uso do formato JSON que facilita sua integração com diversos projetos.

Com o sistema de extensibilidade (*add-ons*) é possível usar diretamente o WebThing Gateway no projeto A.D.A., entretanto isso não parece ser o ideal a longo prazo. Como o *gateway* é um projeto grande, independente e em constante evolução, o custo de manter a integração e softwares necessários para obter as funcionalidades desejadas operando corretamente pode ser maior do que o de desenvolver um sistema mais simples focado exclusivamente para os requisitos da A.D.A..

No entanto, vale ressaltar que as vantagens dessa abordagem são: trazer dispositivos IoT facilmente para o ecossistema do *gateway* já existente, adicionar serviços Web, customizar a interface gráfica e adicionar funcionalidades extras. Dessa forma o *gateway* age como uma camada intermediária, permitindo abstrair a comunicação com os dispositivos, e tornando o desenvolvimento de um interpretador mais simples, pois a interface de comunicação com os dispositivos se torna única. Como trabalho futuro, estamos fazendo uma comparação entre o WebThings Gateway e uma solução própria para o gerenciamento de dispositivos.

Referências

Alam, S. and Noll, J. (2010). A semantic enhanced service proxy framework for internet of things. In *2010 IEEE/ACM Int'l Conference on Green Computing and Communications Int'l Conference on Cyber, Physical and Social Computing*, pages 488–495. GreenCom.

Bröring, A., Datta, S. K., and Bonnet, C. (2016). A categorization of discovery technologies for the internet of things. In *Proceedings of the 6th International Conference on the Internet of Things*, IoT'16, page 131–139, New York, NY, USA. Association for Computing Machinery.

Freire, F., Rosa, T., Feulo, G., Carlos Elmadjian, R. C., Moura, S., Andrade, A., Omena, L. A. D., Vicente, A., Marques, F., Sheffer, A., Hideki, O., Nascimento, P., Cordeiro, D., and Goldman, A. (2020). Toward development of ada–advanced distributed assistant. *XXI Simpósio de Sistemas Computacionais de Alto Desempenho (WSCAD), Brasil, 2020*.

Gupta, V., Goldman, R., and Udupi, P. (2011). A network architecture for the web of things. In *Proceedings of the Second International Workshop on Web of Things*, WoT '11, New York, NY, USA. Association for Computing Machinery.