A CoAP-Based Hypermedia Framework for Always-On and Sleepy Devices in Smart Home Environment

Maxim Kolchin, ITMO University, Saint Petersburg, Russia
Alexey Andreev, ITMO University, Saint Petersburg, Russia
Daniil Garayzuev, ITMO University, Saint Petersburg, Russia
Nikita Chursin, ITMO University, Saint Petersburg, Russia
Dmitry Mouromtsev, ITMO University, Saint Petersburg, Russia
Danil Zakoldaev, ITMO University, Saint Petersburg, Russia

ABSTRACT

Lack of commonly accepted standards for the connected devices’ APIs caused the situation when each manufacturer or Smart Home enthusiast creates its own mobile application to control their devices. The authors propose a lightweight framework to the design of self-descriptive API on top of CoAP protocol using Hydra Core, Schema.org and Semantic Sensor Network vocabularies which allow to create a mobile application with an adaptive user interface to interact with any type of device. In this paper the authors enumerate the requirements, describe the framework, evaluate it on three exemplified devices built using ESP8266 Wi-Fi module and describe the architecture of the mobile application for interacting with such devices.

KEYWORDS
Constrained Devices, Hypermedia API, Resource Directory, Sleepy Devices, Smart Home

INTRODUCTION

Smart Home (or Smart Living) is one of the vibrant areas where devices connected to a network (i.e. smart appliances) make human life more comfortable and effective in terms of performing everyday tasks (Fogli, Lanzilotti, & Piccinno, 2016; Risteska Stojkoska & Trivodaliev, 2017; Saad al-sumaiti, Ahmed, & Salama, 2014). One of the problems preventing Smart Home from becoming an ordinary thing in our life is the heterogeneity of devices caused by the diversity of protocols, technologies and APIs used by manufacturers and solution providers (Saad al-sumaiti et al., 2014; Wilson, Hargreaves, & Hauxwell-Baldwin, 2014). To make development of interoperable solutions for Smart Home easy, it was proposed to reuse the principles and standards of the Web architecture. This idea is known as Web of Things (Kamilaris, Andreas, Andreas, & Vlad, 2011). Web of Things (WoT) approach exposes devices as Web resources that are accessible through standard Web mechanisms, where the REST architecture style is one of these fundamental mechanisms (Fielding, 2000).

Connected devices implementing their APIs, which follow the principles of the REST only partially, caused the heterogeneity problem on another level - the application level, where devices are tightly coupled with a mobile (or web) application from their manufacturer. The principles which most of the APIs skip are: (a) self-descriptive messages and (b) HATEOAS (Hypermedia as...
the Engine of Application State). This is also very common for traditional web services that don’t implement all the REST principles, therefore applications are tightly coupled with a particular API (Bülthoff & Maleshkova, 2014).

This is especially the case for Do-It-Yourself (DIY) devices that are developed by Smart Home enthusiasts. Apart from creating a DIY device they need to provide a user interface that may result in developing its own custom protocol or adapter for existing user interfaces. REST principles although provide another way to tackle this issue by defining the API and functionality of a device in a declarative and self-descriptive way.

In this paper, we extend our previous work (Andreev, Garayzuev, Kolchin, Chursin, & Shilin, 2016) and propose a lightweight framework for connected devices in Smart Home Environment based on the REST principles and existing standard RDFS/OWL ontologies. The aim is threefold: (a) to bring the Web of Things approach to the DIY community, (b) to research some hypotheses about using RDFS/OWL ontologies on connected devices with limited resources, (c) to develop an architecture and a prototype of a mobile application that uses the REST principles and ontologies to provide an adaptive user interface driven by self-descriptive APIs.

The framework should take possible limitations of connected devices into account. Firstly, such devices may be equipped with bare minimum amount of energy, computing power, etc. Secondly, some of them may be switched off (i.e. remain in a sleep mode) most of the time to minimize the energy usage. Taking the ESP8266 Wi-Fi module as an example of a popular low-cost controller for DIY devices we answer the following hypotheses about RDFS/OWL ontologies: Is there enough space in such a low-cost and constrained device to host self-descriptive message? Are self-descriptive messages small enough to fit in a single UDP packet? Is RDF(S) reasoner too resource consuming to use it?

In the rest of the paper we define requirements for the framework, then we present the framework itself, the ways it meets the requirements and the architecture of the mobile application. At the end of the paper we present a case study and evaluation of the framework, compare it with the state-of-the-art and conclusion.

REQUIREMENTS FOR THE FRAMEWORK

Based on state-of-the-art literature (Kamilaris et al., 2011; Järvinen, 2015; Kamilaris & Intizar Ali, 2016), our own experience in developing smart home applications and the Web of Things approach we proposed the following requirements for the framework:

R1: Device discovery mechanism is needed to provide a way to discover available connected devices which belong to a specified type. It should allow to find new device at the local network by request or such device should notify about itself.

R2: REST API on devices. Connected devices should provide a REST API to interact with them. The API should implement self-descriptive messages and hypermedia links between resources. This is the key thing that will allow to build an adaptive user interface of the mobile application.

R3: Service description. The API of devices should describe device’s model, common characteristics, supported functionality, etc. Since some devices have limited resources, so there should be a way to refer a client to additional description of device on the Internet or local network.

R4: Notification mechanism should be implemented to notify the client about new data, such as new observation or state of the device’s control.

R5: Sleepy devices should be supported. Such devices may not be directly discovered at any time due to their sleep modes (Keranen, Ersue, & Bormann, 2014). This requirement makes possible to provide permanent access to impermanent resources via registration and updating device’s data at the additional permanent proxy server.

R6: Gateway as an option. It’s not always feasible to set up a gateway as a separate node of the network to interact with connected devices, therefore the framework should support both options.
Related Content

Multiple Description Coding for Multipath Video Streaming
www.igi-global.com/chapter/multiple-description-coding-for-multipath-video-streaming/135467?camid=4v1a

Design and Implementation of a Low-Complexity Multiuser Vector Precoder
www.igi-global.com/article/design-implementation-low-complexity-multiuser/62991?camid=4v1a
Mathematical Model for Cyber Attack in Computer Network
[www.igi-global.com/article/mathematical-model-for-cyber-attack-in-computer-network/180323?camid=4v1a](www.igi-global.com/article/mathematical-model-for-cyber-attack-in-computer-network/180323?camid=4v1a)

The Model-free Adaptive Composite Control Method for Permanent Magnet Linear Motor
[www.igi-global.com/article/the-model-free-adaptive-composite-control-method-for-permanent-magnet-linear-motor/132671?camid=4v1a](www.igi-global.com/article/the-model-free-adaptive-composite-control-method-for-permanent-magnet-linear-motor/132671?camid=4v1a)