International Journal of Research in Engineering, Science and Management
Volume-2, Issue-2, February-2019
www.ijresm.com | ISSN (Online): 2581-5792

Monitoring Sensor's Data with IFTTT for Smart Home

Rupali P. Wankhade[1], Sneha M. Wagh[2], A. G. Waghade[3]
[1,2]*Student, Department of Computer Science and Engineering, Desscoet, Dhamangaon Rly., India*
[3]*Professor, Department of Computer Science and Engineering, Desscoet, Dhamangaon Rly., India*

***Abstract*: Smart home is an emerging technology for intelligently connecting a large variety of smart sensors and devices to facilitate automation of home appliances, lighting, heating and cooling systems, and security and safety systems. The intention of the platform is to provide an easy to use and highly customizable system for collecting and processing sensor data in a variety of environments and applications. In this paper, the design and implementation of the system through the use of a Python client, a PHP and Flask Web Server/API, a Relational Database Management System (RDBMS) MySQL and external connections to services such as If This Then That (IFTTT) are discussed in detail. Our study has revealed several interesting concepts which can be expressed as role/application/security models. Adapting these models for the domain of home automation systems, a multi-tier software framework is proposed for automating home activities according to the key concept "IF-This-Then-That".**

***Keywords*: Home Automation, Software Framework, Open Source, Raspberry Pi.**

1. Introduction

Today the smart home market is still in its infancy. Although numerous vendors have produced and marketed hundreds or thousands types of smart home devices, such as smart lights, smart switches and smart outlets, many of them can only interact with products from the same manufacturers. To foster intervendor compatibility and encourage community-based software development ecosystems (e.g., iOS App Store), some major players in the market have developed a few smart home platforms to encourage manufacturers to produce compatible devices and software developers to develop applications with a uniform abstraction of smart devices. Recent technological advances in networking hardware and software [1] have expanded the scope of home automation applications to smart grid [2] and healthcare [3]. Compared to other types of buildings, software for home automation systems poses several challenges especially the trade-off between functionality and usability and also the trade-off between connectivity and privacy. Consequently, there is still no dominant technology in the field of home automation systems.

There are several academic works aim to implement advance functions for home automation systems, e.g. voice interface [4], remote operations [5], and cloud computing [6]. Nevertheless these works are constrained in the sense of scalability and portability with respect to the scope of applicable hardware platforms. On the other hand, there are several technical books [7], [8] that explain how to build DIY home automation systems using off-the-shelf components and open-source software. As opposed to academic works, their functionality and technical levels are limited by features provide by their underlying software platforms. While almost of existing commercial frameworks provide high-level of abstraction to achieve functionality and usability. These frameworks suffer from the heterogeneity of devices in the market.

The main concept is to formulate a minimal automaton that integrates multiple code segments that a user chooses from a repository. That is, a sequence of automated activities is programmed by its home owner by selecting and configuring code from available pool. With such a wide variety of options out there for sensors, let alone ways to implement them, there can be a lot of unnecessary redundancy in development effort. We aim to provide a flexible solution that will enable the average user to harness these technologies to perform desirable tasks. When looking at the current mainstream alternatives, such as sensors and commercial Internet of Things (IoT) devices, they are almost always proprietary hardware coupled with limited and closed source software. This means their usefulness is dependent on what the developer provides in terms of functionality and that they may not work on all platforms. However, with the Sensorian Hub, we aim to make it platform agnostic, working with multiple languages by using a REST API to allow all of the sensors to interface. Finally, by making these kits easier to set up, they will have a smaller learning curve and be more attractive and customizable than other standard offerings.

2. Overview of the system

The use of the Raspberry Pi as a Sensor Web node for home automation has been explored. Using a similar strategy for displaying the data through a RESTful service and Apache, our implementation differs in that it is designed to be hosted separately from the Pi and to store the data from as many sensor nodes as desired. While the server could also be hosted on one of these nodes, having it hosted in a central location and with unrestricted access to the Internet gives the added benefit of being able to interface with other useful services such as If This

Then That (IFTTT).

IFTTT model are based on some model as discuss as follows:

A. *Role model*

Within the IFTTT environment, there are three groups of stakeholders, namely IFTTT itself, channel owners, and end users. Each stakeholder will have their own objective, scope of access, functions, and benefits. The use case diagram (see Fig.1) represents the relationship among stakeholders.

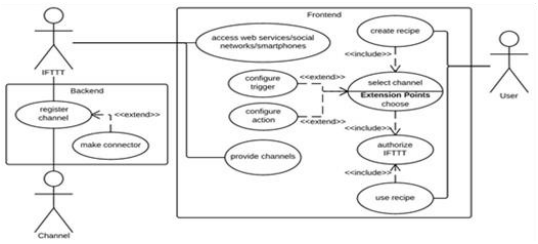

Fig. 1. Role model within the IFTTT environment.

The role of IFTTT itself is to provide channels to end users and to access web services/social networks/smartphone on the behalf of each user. Each channel represents a collection of available data/operations from its corresponding service. The role of channel owners is to make connectors to bridge with IFTTT backend services. Companies that would like to promote their services/products as IFTTT channels must submit their APIs for the approval. There are two main functions for end users: recipe creation and recipe usage. Each IFTTT recipe is created by joining between a trigger and an action resulting in the information flow from one service to another. By separating roles of IFTTT/channels/users, good usability is achieved for almost end users due to its simple workflow. Even UI is simple, functionality is compensated by the availability of channels and corresponding triggers/actions. These affect a reasonable trade-off between functionality and usability. This role model should also serve reasonably well for home automation systems in which homeowners prefer customizable automated activities and easy-to-use UI.

B. *Application model*

There are two different perspectives for the application model of IFTTT; namely developer perspective (channel) and user perspective (recipe). For the provision of channels, IFTTT manages basic channels for popular services, while third-party channel owners are responsible for their web APIs (authenticate/ authorize/access). The scope of development is to abstract and wrap API-related operations as a collection of triggers/actions in each channel. For end users, automated activities can be customized with respect to the creation or selection of recipes. The procedure to create a recipe is outlined as follows:

- Register and login to IFTTT.
- Create a trigger and an action:
- Select a channel from the provided list
- Activate by authorize IFTTT with the service provider underlying such channel.
- Select a potential trigger/action from the available list.
- Fill needed information for the selected trigger/action.
- Fill description and activate recipe.
- Share recipe if needed.

3. System architecture

A. *Sensorian hub*

The proposed solution to the lack of an easy, allen compassing client for use of a Raspberry Pi as a sensor node was the creation of the Sensorian Hub client and its partner web application. The main goal of this solution was to create a basic, modular code base for getting the average user off the ground easily with their sensor-based application and started with deeper integration of its functionality.

As shown in Fig. 2 the Sensorian Hub is broken into three main parts. The Web Server hosts the site which receives the sensor data from the Clients, stores and retrieves this data from a MySQL Database for display on the site to users, and accepts registrations of new users and sensors. The public-facing Relay allows for communication with Clients behind secure networks by listening for connections established by the Clients first. This allows for commands to be sent from users outside the local network or other cloud services such as IFTTT.

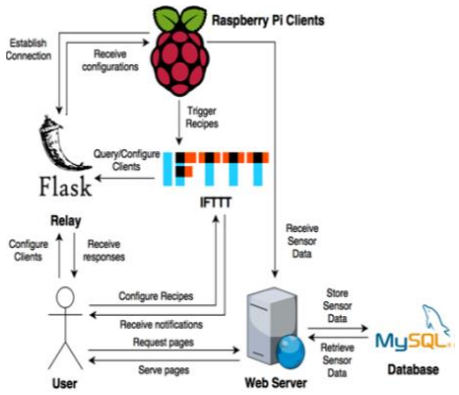

Fig. 2. High-level Sensorian Hub overview

The objective of this work is to develop a framework that enables the integration of DIY components for automating home activities. To achieve both functionality and usability, the following modifications to the IFFFT model are proposed.

- There are three groups of stakeholders: homeowners, open-source developers, and web services/social networks.
- The platform consists of three computer systems including a programmable gateway, a public cloud, and a user terminal (e.g., computer, smartphone).
- Channels are provided by scripts in the programmable gateway and web services.
- Both channel review and activation processes are

handled by the homeowners themselves.

These modifications correspond to the nature of DIY home automation systems in which some scenarios may conflict with mandatory requirements of IFTTT or similar Internet of Things platforms. For example, homeowners may build their systems by purchasing and assembling off-the-shelf parts. Internet connection may not exist or not reliable. Almost use cases of home automation systems prefer restricted group access and unbalanced connectivity (at-home and anywhere) features.

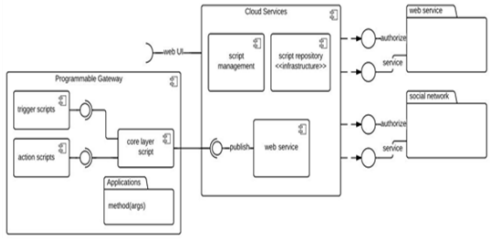

Fig. 3. Software architecture of our proposed framework.

The software system on the public cloud (see Fig. 3) acts as a code repository that allows developers to upload their code and provides custom scripts and configuration files for each user. Another service is to serve as a broker for the request of access tokens from web services/social networks that enforce the OAuth authorization flow. The software system on the programmable gateway will handle automated jobs according to the concept of "IF-This-Then-That" by running code expressed in configuration files. The user terminal will only be used for the authorization of our cloud services with web services/social networks.

4. Conclusion

Our framework consists of two software systems: web applications as cloud services and software stack on home automation gateways. Our cloud services handle two major functions: repository of automated scripts and API bridges for web services/social networks. The software stack integrates the simplified core script, a collection of trigger/action scripts, and software applications providing features. The main advantage of our proposed framework is its simplified design while preserving extensibility and usability.

References

[1] Paul Black. 2008. Levenshtein Distance. In Dictionary of Algorithms and Data Structures.
[2] David Brumley and Dan Boneh. 2005. Remote timing attacks are practical. In Computer Networks.
[3] Shuo Chen, Rui Wang, XiaoFeng Wang, and Kehuan Zhang. 2010. Side-channel leaks in web applications: A reality today, a challenge tomorrow. In IEEE Symposium on Security and Privacy (S&P).
[4] Kyong-Tak Cho and Kang G. Shin. 2017. Viden: Attacker Identification on Vehicle Networks. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17). ACM, New York, NY, USA, 1109–1123.
[5] Smart Things Community. 2015. How many Smart Things users? https://comm unity. smartthings.com/t/how-many-st-users/10928/3.
[6] Q. Mahmoud and D. Qendri, "The Sensorian IoT platform", 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC), pp. 286 - 287, 2016.
[7] S. Banerjee, D. Sethia, T. Mittal, U. Arora and A. Chauhan, "Secure sensor node with Raspberry Pi", IMPACT-2013, pp. 26 - 30, 2013.
[8] S. Ferdoush and X. Li, "Wireless Sensor Network System Design Using Raspberry Pi and Arduino for Environmental Monitoring Applications", Procedia Computer Science, vol. 34, pp. 103-110, 2014.
[9] "RFC 7252-The Constrained Application Protocol (CoAP)", Tools.ietf.org, 2016. https://tools.ietf.org/html/rfc7252.
[10] V. Vujovic and M. Maksimovic, "Raspberry Pi as a Wireless Sensor node: Performances and constraints", 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1013 - 1018, 2014.