DATA ACQUISITION TECHNOLOGIES AND SENSOR NETWORKS

Project Report

Smart Outlet: Access and Control Your Power Outlets Remotely

By Eno Ciraku, Diogo Cosin, & Ralph Florent

November 13, 2019

Abstract

[Here goes the absctract of the documentation...]

1 Introduction

[Here goes the introductory part of the documentation...]

2 State of Art

[Here goes the state of art part of the documentation...]

3 Intrumentation & Resources

Recalling that the SO prototype is the complete integration of a set of hardware and software, we detail in this section more in-depth specifications on them. Below are enlisted the tools used to set up and carry out successfully the current version of the project.

3.1 Hardware and other materials

The SO hardware refers to the physical components that add up to the *Webduino* module. Although there are many possible options in choosing different kinds of hardware to build up the circuitry, we opt for the most reasonable[1] choice, which is to reuse already-prebuilt modules and integrate them into one. Hence, in Table 1 are listed the items with their corresponding details.

Hardware & Other Materials			
	Models	**Quantity**	**Cost**
Microcontroller Board	MEGA 2560 R3	1	€ 14,99
4-Channel Relay	GE-EL-SM-006	1	€ 6,99
Wi-Fi Module	ESP8266-01	1	€ 6,99

Table 1: Detailed information on materials and hardware used for the SO prototype.

Additionally, other useful materials such as:

- 12x male-to-male jumper cables (20 cm)
- 12x male-to-female jumper cables (20 cm)

[1] Building a separate module from scratch requires time and work, plus other tasks to refine a working component.

- 12x female-to-female jumper cables (20 cm)
- 1x access point or router (tp-link TL-WR940N)
- 2x computing devices or laptop (Lenovo T460p and Macbook Air)

remain useful to connect the different components and have them work as one stable module. More technical specifications are given on each one of the materials and devices, including their working conditions, in Section 2.

Notes: Observe that the monetary budget reaches the sum of €28,97 for only one SO prototype that eventually includes the monitoring and control of 4 (four) power outlets. Most of the materials were bought online on Amazon.de and some of them, provided by the *Data Acquisition* lab.

Considering that the main purpose of building this prototype is totally educational, we do not account for commercial goals. Obviously, this leaves room for improvement in the future if we want to aim for industrialization of the project. That signifies at some point we might need to rethink the choosing of tools and software in order to optimize the build of the hardware module, which may, in turn, favor a cost reduction in our financial records.

3.2 Tools and software

As for the programming tools used to create, debug, maintain the code of Smart Outlet at both hardware- and software-level, we use free programs yet efficient that are available for most of the OS[2] platforms. These programs are of different types such as code editors (e.g Visual Studio Code), compilers (e.g LaTeX), online platforms (e.g GitHub) and so forth. With that being said, we present the set of tools and software used at the time of implementing the prototype:

- Operating systems (GNU/Linux, Mac OS, and Windows)
- Visual Studio Code (lightweight text editor)
- Git[3] (version control)
- GitHub (web-based hosting service for Git versioning system)
- Jupyter Notebook (workspace for scripting and simulation)
- Arduino IDE (development environment for Arduino boards)

Regarding the software versions, it is highly recommended to use the exact versions mentioned in Table 2 to avoid conflicts and compiling errors. On the other hand, the developer may want dig

[2]OS: Operating Systems like Windows, Mac OS, Linux, and so on.
[3]Git is also available as a bash emulation for other platforms for free (e.g., Git Bash for Windows).

Tools & Software			
	Versions	**Sources**	**Cost**
Visual Studio Code	1.40.1	See [1]	Free
Git	2.7.4	Built-in Linux program	Free
GitHub	N/A	See [2]	5 free users
Python	3.7.1	See [3]	Free
Jupyter Notebook	5.7.4	See [4]	Free
Arduino IDE	1.8.10	See [5]	Free

Table 2: Detailed information on the tools and software used for the Smart Outlet coding procedure and the technical documentation.

into the breaking changes (if that is the case) that most of the time require to refactor parts of the implementation to have a fully working prototype. However, it is also recommended to check the *changelog* of the updates or releases, if there are any.

Note that some tools mentioned above are just a matter of personal preferences. Other preferred options are more than welcome as long as the developer keeps in mind the development speed and productivity. For example, many developers would choose Sublime over *Visual Studio Code*. But they both end up facilitating the same routine: text edition.

3.3 Programming languages

Finally, we use the following programming languages:

- C/C++ (for the webduino)
- Python (for the web API service)
- Angular Framework - JavaScript/TypeScript (for the web application)

Important: *Though we highly recommend that the exact versions of the software and the exact models of the materials and devices are used to test out or replicate this project, keep in mind that these hardware might no longer be available in the market as well as the software components might be* *outdated at some point in time. If that is the case, stay alerted to the updates as we intend to support this project until 2022.*

4 Methodology

In this section, we describe the techniques and strategies as well as the procedural methods used to implement the core functionality of this project. This description includes the components of the system, the workflow diagram, the third-party libraries, the algorithm and data structure, and finally, the code implementation.

It is important to stand out the fact that we use external resources to come up with daily results in terms of programming tools and full-on working devices. That is, before adventuring into using the materials and devices as well as certain software tools, different research sources were consulted. Some of these resources are: the datasheets of the prebuilt modules of the materials, the assistance of the instructor and teaching assistants (TAs), search engines (mainly Google Search/Internet), and finally some related books and references (e.g. materials provided by the professor).

Being exceptionally helpful, these resources have been the source of truth for any decision-making regarding the correct use of the hardware modules.

4.1 Components of the prototype

Roughly speaking, Smart Outlet comprises 3 (three) principal modules:

- ***Webduino***: a low-level, modular circuitry formed by an Arduino and a network of sensors. The Arduino board, a microcontroller, acts as a supervisor of micro tasks. Being the core component of the hardware systems, it controls the different input/ouput functions of the connected chips.

- ***Web API***: an API service attending HTTP requests from the *webduino* (its only consumer). It coordinates the communication between the Wi-Fi module of the webduino (client) over the air medium (wireless) and the available API resources on an HTTP server. The API service contains various layers of interactons, including the database for data persistence.

- ***Web APP***: (short for web application) a single page application (SPA) to visualize the historical content or performed actions during the webduino's operations. It allows user-friendly interactions between an end-user and the prototype itself. The web app is also responsive. That is, it can be accessed and used via mobile devices (tablets, smartphones, etc.).

Each one of these modules deep down contains a set of characteristics that requires more than a brief description to highlight their corresponding functionality. However, in this document, we intend to only explain how to connect them together and make them work properly as a whole. We indeed provide full access to the online repository as specified in Appendix A so that anyone can dig any further into the datasheets if needs be.

4.2 Workflow diagram

The Smart Outlet project has an easy-to-understand workflow. This workflow explains how each module is connected to each other (not fully as in a mesh connection) and performs a specific task. In the section, we provide a diagram and explain the corresponding role of the inner contents taking part in that workflow in other to achieve the complete functionality of Smart Outlet as a whole.

As shown in Figure 1, the workflow diagram consists of the following parts:

- some end-user devices accessing the web application
- a server infrastructure attending requests over TCIP/IP network
- the webduino connected to the same network waiting to update the states of the outlets, if any.

How do the components interact with each other?

1. given the initial conditions, where all the components are up and running properly[4]
2. the webduino will constantly attempt to request the last states of the power outlets from the web API service. Once obtained the data over Wi-Fi, it proceeds by updating the states of the relays, which in turn will open or close the switch of a power outlet.
3. A user accessing web browser may view the last updates of the power outlets. When a user updates[5] the state of an outlet, the changes are saved into the database through the API service and immediately (2 seconds) reflected in the webduino.
4. The historical records can also be visualized on a separate views using both a tabular form and a scatter plot.

4.3 Algorithm and data structure

4.4 Code implementation

[4]The devices (database, API web service, webduino) should be powered on and running. Each device should be assigned a local IP address and should be reachable by other devices over the network.

[5]Set an update for a power outlet is basically to change its state from ON-OFF or viceversa.

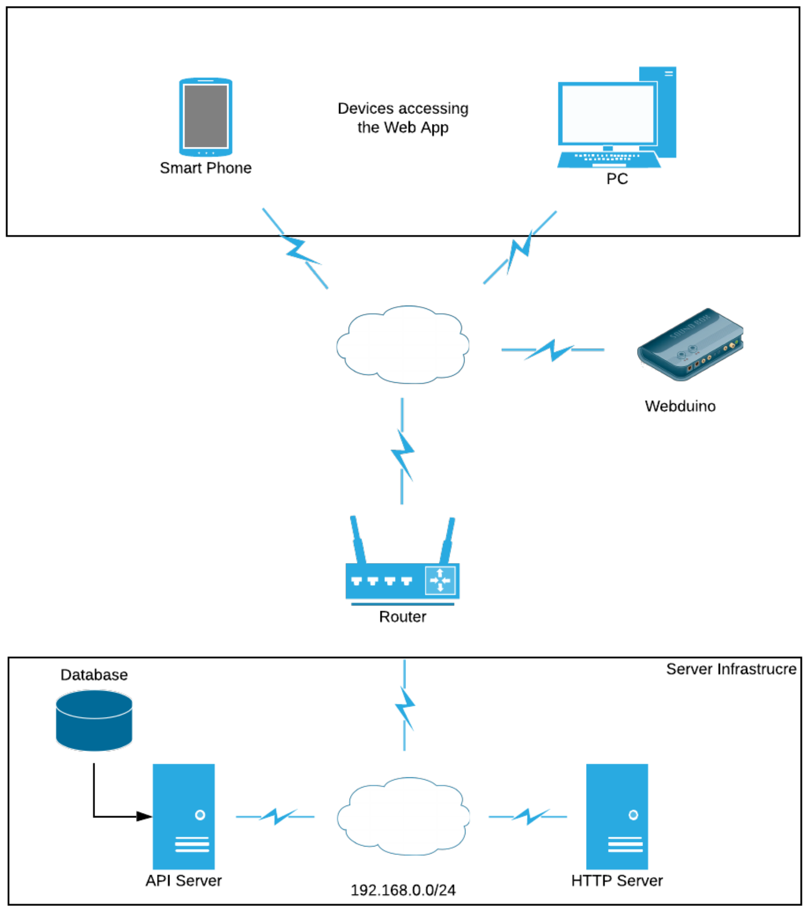

Figure 1: Workflow diagram

(credits: made with *Lucidchart*)

References

[1] Microsoft Corporation. *Visual Studio Code*. Oct. 2019. URL: https://code.visualstudio.com/.

[2] GitHub Inc. *The world's leading software development platform*. Nov. 2019. URL: https://github.com/.

[3] Python Software Foundation. *Welcome to Python.org*. Mar. 2019. URL: https://www.python.org/.

[4] Project Jupyter. *Project Jupyter*. Jan. 2019. URL: https://jupyter.org/. (Last updated April 12, 2019).

[5] Arduino. *Arduino - Software*. Oct. 2019. URL: https://www.arduino.cc/en/main/software. (Last updated October 20, 2019).

Appendix A Code Repository

All the code implemented during the execution of the prototype described in this report is available on the GitHub repository https://github.com/ralflorent/smart-outlet.