Automated Security and Whereabouts Tracker

CS426 Final Project Report

STYX MEISELES, Binghamton University, United States

The popularity of Internet of Things paves the way for constant inter-device connection - perhaps allowing *us* to disconnect. This project focuses on how we can use IoT devices to acheive peace of mind while still allowing for a social life and home security. To do so, I constructed a network of three Raspberry Pi devices, one TelosB device, and various sensors, which when connected, can replace much of the core functionality of smart devices while allowing for a distraction-free environment.

Additional Key Words and Phrases: Raspberry Pi, TelosB, NesC, IoT, buzzer, external display, connection

1 INTRODUCTION AND MOTIVATION

Smart devices can be a very useful tool. They allow for communication between friends, provide a distraction from boredom, and equip us with a social hub. However, they also promote lower attention spans. It is quite common to scroll endlessly through social media posts to distract oneself from homework and responsibilies. The aim of this project is to minimize the *need* to have a phone on ones' person, and thus help broaden attention spans, promote work efficiency, and heighten overall life satisfaction.

2 REQUIREMENTS

A number of devices and technologies were used in this project: three Raspberry Pi 3's, a TelosB device, a buzzer, an external display, and a (surprisingly difficult to find) steady power supply. To program these devices, we used TinyOS' NesC, Java, Python3, and NodeJS.

3 PROGRAMMING

The devices were named based on their functionality: one Raspberry Pi kept inside the room and connected to the TelosB, one placed on my door with the external display, and one carried around with me with a buzzer attached to it.

I programmed the TelosB device in NesC to forward data from its light sensors to the Raspberry Pi through a serial connection. A button located on the TelosB also allows for calls to be sent to me, with an LED providing further user feedback. This information is also forwarded to the Raspberry Pi.

Upon receiving the sensor data, the Raspberry Pi then calculates whether the light is on or off, connects to an external API to discover its current location, and forwards the data through a NodeJS WebSockets connection to the central hub - the Raspberry Pi mounted on the door.

Simultaneously, the device I carry with me calculates *its* location, and transfers that to the central hub as well.

Upon receiving data from the three supporting units, the central hub calculates whether I am in my room, whether someone *else* is in my room, and whether someone is calling me. Thereafter, an AdaFruit Python script running in the background gathers this information and displays it on external monitor. If someone is calling me, or if the system detects an intruder, it also alerts the other Raspberry Pi, which starts either a buzzing call, or a harsher alarm sound, depending on the circumstance.

4 DESIGN

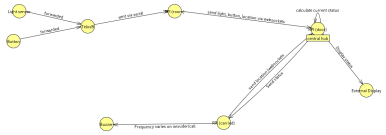

Fig. 1. Design Diagram - Logic Flow

5 IMPLEMENTATION

The project implementation started with programming the TelosB to gather light and call data. This was done inside of a virtual machine running TinyOS, and then the program was copied over to the actual TelosB device.

The TelosB was then connected through USB to a Raspberry Pi, which pings the TelosB on a set interval for data. Upon receiving input on whether a call was initiated or the room light was toggled, it sends that data to the central hub.

The central hub receives the data, pings the carried Raspberry Pi for its location, compares the two, and then communicates with the external monitor to display necessarily information (eg, whether the user is asleep, out, available, or busy). It also communicates with the carried device to activate an alarm for a call, or a buzzer for an intruder.

Altogether I used 3amp charging stations for each Raspberry Pi, a set of jumper wires, and OLED module display, and 3V buzzers. Programming resources included use of an external monitor and keyboard to initiate the connection with the Raspberry Pi's.

Author's address: Styx Meiseles, Binghamton University, United States.

2021. XXXX-XXXX/2021/5-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

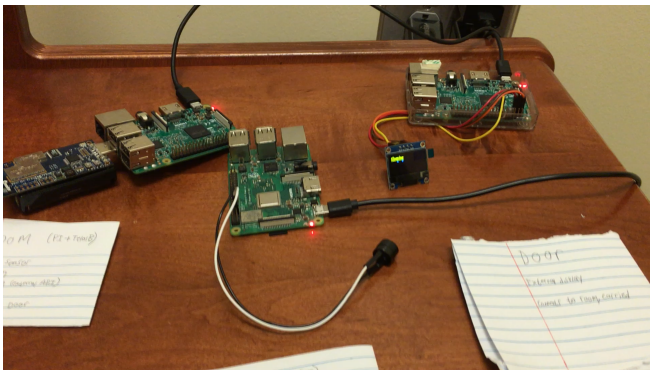

Fig. 2. Sensors and Devices

6 CHALLENGES

This project contained many technical challenges. Foremost was the initial inability to connect to the Raspberry Pi's. Due to working in a University dorm environment, it was impossible to connect through a router to initiate the connection. After much trials I was able to borrow a router and monitor from the TA, as well as an external keyboard from a friend. Upon doing this, I was successfully able to connect to the Raspberry Pi and start programming.

However, the University's baseball stadium required almost the entire university's internet to be reset. Thus, my devices lost their static IP addresses and I had to repeat this arduous process once more.

On the programming side, NesC is a niche language with poor documentation, dated project maintainance, and little programming editor support. Through looking at examples online I was able to overcome this hurdle, however installing TinyOS components required communicate with the TelosB on my Raspberry Pi was another challenge in itself.

Also, I elected to use NodeJS for my webservers. However, AdaFruit uses Python3. Thus, I had to come up with a cross-language communication protocol using files as a communication medium.

In the end, despite these difficulties, I was successful in implementing the entire planned project scope.

7 RESULTS AND FUTURE DEVELOPMENT

The current network is merely a prototype demonstrating the possibility of creating a more refined version of itself. It uses bare devices, as well as WiFi. A future version would be equipped with a cellular signal, a battery pack, and more accessible buttons. Due to financial considerations, these niceties were omitted.

8 CONCLUSION

This project showcased the ability of IoT devices to allow us - somewhat ironically - to *disconnect* from the internet. Allowing us to spend more time in the real world, focusing on the moment, and on personal and professional responsibilies. This network prototype successfully achieves its purpose of allowing for peace of mind while limiting ones use of smart devices.