Raspberry Pi in Wearable

Maciej Medyk and Santiago Aguerrevere
Semantic Web Programming, Computer Science
Florida Atlantic University

Abstract

We are entering a new time in our technology, an era of wearables, where integration and communication among devices will provide synergy in our lives. In our project we have looked into such integration between pedometer application on mobile phone and Raspberry Pi that would not only track the pedometer data results, but also control additional external device based on those results. Results are stored in semantic file system that employees ontology in its design and associates data with object's schema.

Keywords

Wearable technology, device close proximity communication, semantic web, pedometer, python application, mobile application.

Background

The project has been developed around four modules of functionality which were: data collection, data analysis, data storage, and external device control. Integration between data source and data analysis initially posted the biggest challenge as both were using different platforms and some applications were not open for modification thus any form integration. Initial focus was to find a pedometer device that would simply allow data to be shared and easily synced. After extensive research of stand-alone devices, we concluded that the best route is to use a mobile phone application either iPhone or Android based. Another challenge we came across was how to automate the process of receiving the files and loading them to the program to achieve synergy among devices. Raspberry Pi would have to be device that would be fully automated and always sniffing for new input of data from various sources without user interaction. Extension of the functionality in the Raspberry application was storage of data in semantic database and control of the external device via output pins on 40-pin port.

Method

For data source we have used two different method of collecting and obtaining data, first one through a self-made application developed for Android phone in Eclipse environment, and second one was a commercial application available in Apple App Store called Pacer. Android application was developed to gather and store pedometer data that can be transferred to Raspberry Pi via Bluetooth. Application has a simple interface and track accumulated steps which it stores separated in 15 minute intervals in csv file on the phone. Data consists of three

rows which are date, time, and cumulative steps to date. Application consist of Bluetooth module method that allows to transfer the file named "DetailedSteps_yyyy_mm_dd_tttt.csv" to Raspberry Pi. Alternative solution to Android application is commercially available application called Pacer. It also tracks the data in 15 minute intervals; however, due to Apple sharing restrictions it only allows the transfer of data via email protocol. It also sends a file named "DetailedSteps_yyyy_mm_dd_tttt.csv".

On the Raspberry Pi there is a Python based application that is enabled via script to run at start. The application was designed to be locked in infinite scanning loop in which at start of the loop application logs into designated email account to check new mail and download available attachments in order to obtain files sent to the program from iPhone by email protocol. The application is checking one designated email account "cop5859@gmail.com" and code contains login credentials for both username and password that can be changed in order to accommodate other email accounts. The protocol is written to work with Google or FAU email account. Once the email account check is done and new attachments downloaded into Downloads folder, application proceeds to scan Downloads folder for files that were not yet analyzed. In this step application will detect either newly downloaded file from email server or newly obtained file from Bluetooth download. What application does at that moment is checking for specific name file format "DetailedSteps_yyyy_mm_dd_tttt.csv" by brute force search. Simply there are looping program that is initiated that checks for year, month, day, and time of the file in ascending order from 2015_01_01_0000 to 2017_12_31_2399. For the program needs and testing the time interval has been set to two years; however, it could be easily extended and changed to a different time period per need. Once the file is found and file is recognized as new it will be opened and file name would be added to the list of analyzed files. At that moment program will load the content of that file to memory and analyze the amount of steps in last 15 minutes as compared to last 60 minutes on average. If application will determine that more steps were done in last 15 minutes than in last 60 minutes it will set the voltage on pin 3 to 3.5V allowing relay to be set to on position turning on the air fan. Inversely, if application determines that less steps were done in last 15 minutes than in last 60 minutes it will set the voltage on pin 3 to 0.0V allowing relay to be set to off position turning off the air fan. Regardless of outcome the data analysis will be stored in local SQLite database and in Ontology Web Language database file.

Raspberry Pi integration with external air fan device was available due to 40-pin interface and driver configuration that was controlled via "Wiring Pi" Python module, which was developed to allow to set input or output values on port's pins in order to control external devices or to be controlled by one. In order to connect external device we had to use a relay which would be controlled by three pins. Pin 1 would supply constant power to the relay while pin 6 would supply ground. Pin 3 would be the control pin that would either supply 3.5V or 0.0V depending on application code. Relay would then open or close a spliced power circuit going to the air fan.

Final module of our project was devoted to storage of data in semantic database. Initially we had to create a schema file in Protégé named "fancontrol.owl" that would contain object properties, classes, data properties, and annotations. Python program then would load that file

into memory each time it would need to add the individual record and save it overwriting the old file. Additionally we implemented relational database to store data and display it on the screen during application operation. Both databases are being accessed and updated upon analysis of data done by the application when determination is made to either turn the fan on or off.

Results

As a result of the project we were able to find semi-integration between both mobile applications and Raspberry Pi, we were able to automate the receiving process of files and automate the file searching and processing. The only user interaction at this moment is around mobile phone application where user has to send data to device. Raspberry Pi application and everything that follows is fully automated. In addition we were able to integrate Raspberry Pi with external air fan device which is automatically controlled based on data received from pedometer applications. Lastly we were able to create two databases both relational and semantic to store data derived from analysis of files. The databases serve as simply storage of data without search query functionality and display functionality is limited to Raspberry Pi while program is in progress. There are some additional limitations where email delivery is limited to one email account and in order to be changed the code of application has to be updated with new username and password. Another limitation is a time interval that is being checked for the new files. At this moment the time is limited to two years from January 1st 2015 to December 31st 2017. This time period can be extended and modified; however, once more code of the application needs to be updated to the new interval.

Conclusion

The project has been a learning experience in how to adapt and automate processes, how to integrate various platforms, and had a good mix of mobile development, desktop app development, database development, and electrical engineering that were essential to understand in order to develop and integrate all four modules of the project. Santiago and I, we both believe we achieved a success although if given time would pursue the idea of even closer communication and integration between devices where no interaction is necessary at the mobile application level and where database results would be available through the application. Also given more time we would have worked on limiting or eliminating the need for updating the code with email credentials or updating the code for new time interval of files that need to be checked.

References

Video Presentation at 5 x speed with annotations
http://youtu.be/IZbj7TuvPh8

Appendix

Android Mobile Application – Written by Santiago Aguerrevere – syncs via Bluetooth
https://github.com/maciejmedyk/COP5859/tree/master/Team%20Main%20Project/Mobile%20Application

IPhone Mobile Application – Pacer Pedometer plus Weight and BMI Management and Blood Pressure Tracker – commercial application available on iTunes Store – syncs via E-mail
https://itunes.apple.com/us/app/pacer-pedometer-plus-weight/id600446812

Python Application – Written by Maciej Medyk – receives and analyzes data, saves data to database, and controls external device
https://github.com/maciejmedyk/COP5859/tree/master/Team%20Main%20Project/Pi%20Clean%20Database

Wiring Pi – GPIO Interface for Raspberry Pi - allows the access to the Raspberry Pi port to control external device
http://wiringpi.com/download-and-install/