Midterm Report

Mark Swanson – Z23325068
COP5859 - Dr. Shankar
2015-07-17

1 Personal Health and Fitness Analysis Using Semantic Web and Embedded Systems

Fitness bands and other personal health tracking devices have become commonplace for people of all health and lifestyle backgrounds. According to analyst firm Canalys, seven million fitness bands were sold in the first quarter of 2015. The data associated with these devices may already be stored in a cloud service either through a third-party application or by the company who sells the device. Smart watches, cell phones, and in-home products have begun tracking and storing personalized data which may be used for assisting in the prognosis or diagnosis of specific health metrics or diseases. However, presently the majority of consumer-grade products are limited to displaying and storage of data with little analysis being done.

The proposed project, *Personal Health and Fitness Analysis Using IoT and Embedded Systems,* will collect tracking data from an existing product and combine this data with ambient data where the embedded system resides in the consumer's residence. Storage and analysis of the data will occur locally on the embedded system. Simple analysis will initially be performed which relates the two data sets; an example of this simple analysis would be comparing the number of steps a person took in a given day with the number of times they left and returned to their residence. Light switch usage, temperature, electronics usage (TV, computer), and other ambient inputs can be collected by the embedded system to be used for combined data analysis. Outputs of this analysis may include the type (active, at-home, out-and-about) and activity level (high, medium, and low electronics/room usage) of a consumer's day.

The combination of a hardware ecosystem, physical monitoring of both the consumer and the consumer's environment, and finally the use of Semantic Web compatible solutions for data analysis present a complete Internet of Things solution for personal health and fitness analysis and provide a platform for easy future integration into the Semantic Web. Combining data from not just a single consumer or single residence, but instead data from a community or the world may enable future health studies and breakthroughs for the health industry but also provide key insights for individual consumers regarding personal habits they have present in their life that may have significant health ramifications in the future.

2 Project Outline

A Raspberry Pi (https://www.raspberrypi.org/) will be utilized as the embedded system for both monitoring ambient conditions and combining then analyzing ambient data with fitness tracking data. The Raspberry Pi was selected as an embedded system platform due to its large and ever-growing community as well as its stable and reliable hardware ecosystem. Drivers, examples, and other support needs are readily available through online communities.

Ambient data will initially consist of residence entry monitoring (door open and close events) and electronics usage (TV and computer use). Timestamps for these events will be collected through either pushbuttons on hardware input pins of the Raspberry Pi device or a simple graphical user interface on the Raspberry Pi's monitor. These timestamps will be categorized by either which pushbutton is pressed or which button of the graphical user interface is pressed. C or another programming language will be utilized for reading pushbutton information, and Java or another programming language will be used for generating a graphical user interface if required.

A Garmin© vívofit® (http://sites.garmin.com/en-US/vivo/vivofit/) will be used to collect daily sleep, steps, step goal, calories, and distance data. This data will then be downloaded to the Raspberry Pi through a Wi-Fi connection and API calls to Garmin© services. Due to utilizing the Garmin© system and interface, data may need to be collected manually or sent over an Ant+ connection via a custom USB dongle instead for development purposes with the Wi-Fi methodology potentially going unutilized in the final project.

Data from the two aforementioned sources (vívofit® and ambient) will then be combined in to a csv file or alternative simple file format for local storage and later conversion to Turtle or alternative RDF format. Python or another programming language will be used to automate the data combination and storage process.

protégé (http://protege.stanford.edu/) will be used to initially create an RDF file to represent the health and fitness data being collected. Jena (https://jena.apache.org/) may be used later in the project for both writing to and reading from the RDF file created for the collected datasets. The RDF file will utilize the Web Ontology Language (OWL) (http://www.w3.org/TR/owl2-overview/), Resource Description Framework (RDF) (http://www.w3.org/TR/rdf11-concepts/), RDF Schema (RDFS) (http://www.w3.org/TR/rdf-schema/), and XML Schema Definition Language (XSD) (http://www.w3.org/TR/xmlschema11-1/) as the base schemas and ontologies. Time Ontology in OWL (http://www.w3.org/TR/owl-time/) is an ontology of temporal concepts and will be used for both timestamps as well as calculated duration data. Built-in reasoners of protégé will be initially utilized for performing analysis of the semantic information.

3 Project Development Flow

3.1 Raspberry Pi

3.1.1 Ambient Activity Timestamp Generation

Timestamp generation for the tracking of ambient activity and conditions will initially be done using a simple graphical user interface. Python will likely be used to generate the graphical user interface on the Raspberry Pi monitor. Timestamps will be stored to a csv file for later integration with fitness data. Durations will be calculated for each timestamp event by the graphical user interface program.

After completing the initial timestamp generation tool, and following the completion of the first implementation of the following steps in the development flow, a physical interface may be implemented for timestamp generation. This would include a set of pushbuttons which can be pressed by the consumer, with having the Raspberry Pi and interrupt handlers generate timestamp data. The hardware interrupt handling code will be written in C, with timestamp data formatting most likely still being written in Python. The reason this feature has been separated into a second phase of the project

development is that the handling of hardware interrupts and corresponding firmware development effort is not fully understood. Ensuring the remainder of the project can be developed is more critical to the overall success of the project than adding the physical interface.

3.1.2 Garmin© vívofit® Data Collection

Collection of Garmin© vívofit® will initially be performed manually by downloading the data from an online service and storing it in a csv file for later integration with ambient data. Loading the data on to the Raspberry Pi will be done over a Wi-Fi connection while using the Raspberry Pi monitor. This method will enable later steps in the development flow to be completed without significant effort in relation to collection of data for a particular fitness tracking platform. The resultant project will then be more adaptable to alternative platforms, with core functionality that is platform agnostic having more development time.

A secondary phase for Garmin© vívofit® data collection will be done through a set of APIs over a Wi-Fi connection to an online service. This will most likely be written in Python, and will handle downloading new data on a regular schedule, and formatting the data for later integration.

3.2 Semantic Web Development

3.2.1 protégé

As outlined above, the initial RDF file for the combined ambient and fitness tracking data will be created using protégé. The graphical environment allows for easier intuition and understanding regarding the generation of an ontology and will facilitate rapid changes that will most likely occur while initially constructing the RDF file. Data will be loaded manually for overall construction of the ontology. Classes will be defined and tested, but will be based on a limited initial dataset due to the manual nature of entering the data. A later, potential step, in the project development flow involving Jena to assist in automating this process may be undertaken with time permitting.

3.2.2 Jena

Following the initial creation of the RDF file, Jena may be used, time permitting, to automate the data entry and reasoned/analysis process for the combined dataset. Due to the currently unknown nature of the impact or time related to this activity, this piece of the project development will be part of a secondary phase. File format will most likely be OWL or Turtle for utilization with Jena with SPARQL providing query capabilities.

3.3 Data Storage and Integration

3.3.1 Data Storage

Data storage will be performed on the SD card of the Raspberry Pi system. One (1) gigabyte of space is available on the formatted Raspberry Pi card for general purpose use. Due to the efficient and condense nature of the csv file format, even the combined dataset is not expected to reach anywhere near the maximum available space. The SD card also provides an easy mechanism for data transfer from both the Raspberry Pi system and monitor as well as a general purpose PC.

3.3.2 Data Integration

Data integration will be performed on the Raspberry Pi system utilizing Python. Separate csv files for both ambient and fitness data will be combined into one dataset for simpler processing and later integration with Jena. The data sets are kept separate at first to allow for incremental steps to be performed on various steps in the project development prior to data integration being complete, such as building an RDF file.

3.3.3 Data Export for Semantic Web

Data export of the final RDF file and potential alternative outputs will initially be handled through manual file transfers over Wi-Fi through the Raspberry Pi monitor. This method would allow for later sharing to public spaces such as GitHub or alternative Semantic Web communities prior to an automated data export being complete. GitHub allows for any individual with an internet connection to access and utilize the final output information.

Automating a GitHub-based sharing method would be performed as a secondary phase. Automation would allow for a continually updating output and enable simpler expansion of the overall output dataset. Improvements and changes could then be worked on collaboratively on a common output.

4 EXPECTED CHALLENGES

Challenges are expected at every phase of the project development flow with varying degrees of severity, and often the unexpected challenges prove to be the most sever. However, the phases which are currently expected to present the most significant challenges have been placed in a secondary phase to ensure a minimum viable product (MVP) can be produced within the given schedule. A primary benefit of the MVP approach is exploited in the secondary phase, as it allows for the shortest possible development time until an iterative approach to improvements can be implemented. Making small changes to improve the already functional product will give more rapid feedback and result in more rapid improvements.

Collection of online fitness data and the automation of input/output generation for the RDF file are the two main components that have a secondary phase. Specifically related to the online fitness data, presently the method for retrieval and level of effort and learning associated with the completion of this task are unclear. The existence of a public API would drastically reduce both the effort and time to completion of this task. Automation of the input/output generation for the RDF file largely depends on the ability to modify the existing example project for Jena to meet the specific needs of ambient and fitness data analysis.