Christopher Livingston and Jasper Bingham
Professor Smith
CS 69: Risks of the Internet of Things
8/11/15

Smart Coffee Table Project Documentation

Overview:

The Smart Coffee Table consists of a networked dashboard built into a coffee table that is used to access useful information in an easy-to-read interface that can adapt to the placement of a beverage or other object. Users can read email, new, weather reports, and time from a centralized location, supplementing their morning routine with important information that can better prepare them for the day ahead. In this project, we wired a Raspberry Pi and a compatible push-button sensor to a LCD TV monitor and mounted this configuration on a wooden table frame that we built. We wrote the driver software for the graphics and API requests to be displayed on the monitor. From there, we covered the monitor with plexiglass to act as a stable surface for coffee cups and other beverage containers.

The Smart Coffee Table addresses the issue of cellular devices and their disturbance to the social atmosphere of a family gathering, important discussion, or any other social event taking place within your home. With the ever-increasing strength and popularity of mobile computing comes a heightened attachment to our smartphones and other cellular devices. This can lessen the value in having a spacious living room as cellular device usage can lead to a tendency to turn inwards, and effectively ignore those around you. This can become a nuisance to others in the room looking to enjoy your company and share in what you are doing and observing.

The solution: **The Smart Coffee Table**.

The Smart Coffee Table offers the informational benefits of a smartphone without the increased distraction and obstruction to social interaction. It also turns a piece of furniture with no other function than being an elevated surface into an information-packed interface. With our device, the user can enjoy the full benefit of a spill-protected surface for the placement of beverages. The touch display of the Smart Coffee Table remains dormant (black screen) until it is prompted to display the Smart Coffee Table Dash. At the touch of a simple activation button on the Raspberry Pi, the Dash, otherwise know as the main home screen display of the Smart Coffee Table, will appear with information on the primary user's email, events, favorite news feeds, and the weather for the entire week. After a customizable about of wait time, the dash screen will return to a dormant (black screen) state until being reactivated by the button on the Raspberry Pi.

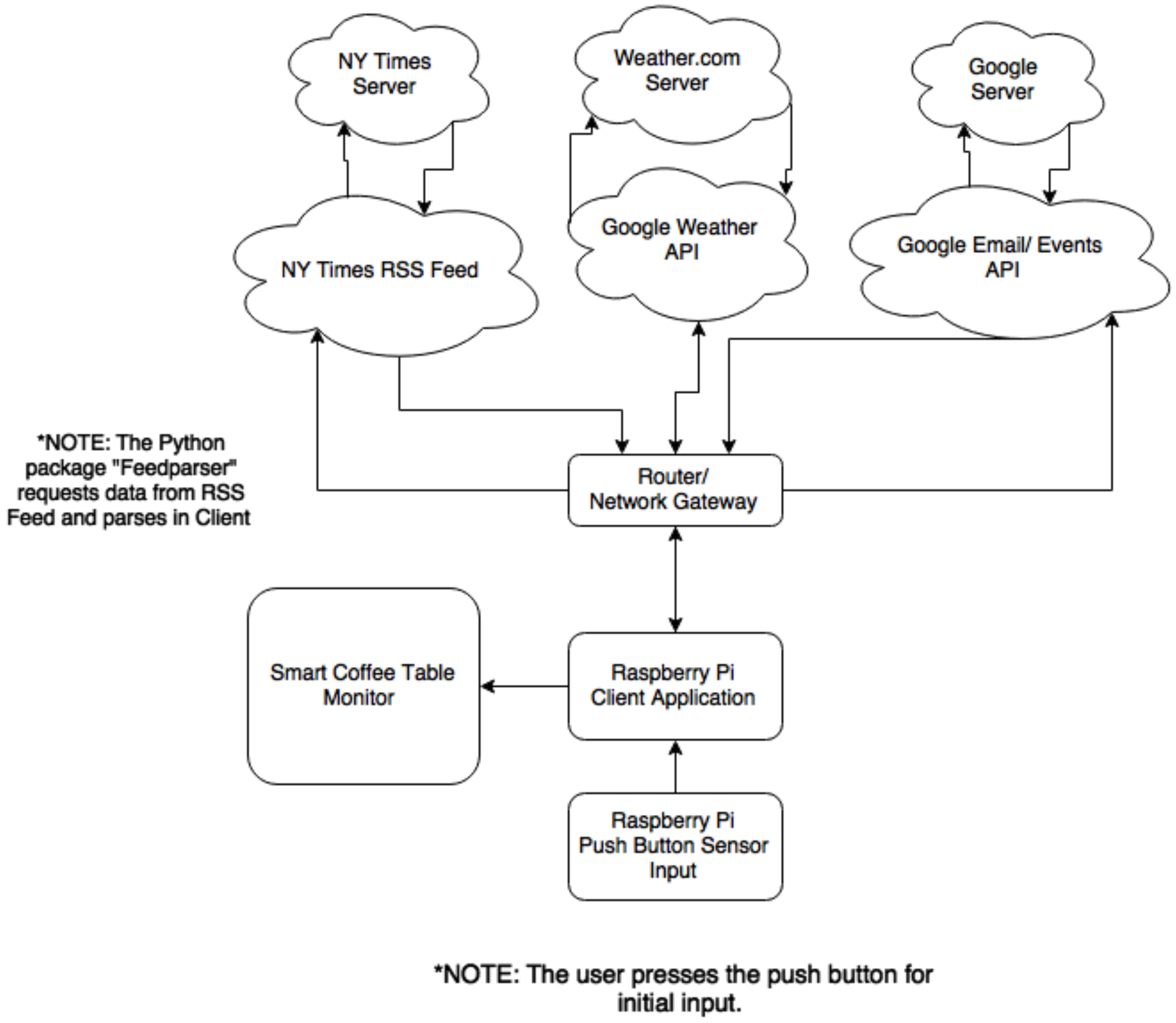
We implemented the hardware components of this project by constructing a wooden frame to act as the supporting table for the monitor and Raspberry Pi device. Metal braces were used to secure the wood components and the wood was lightly sanded to be suitable for monitor placement. We cut a sheet of plexiglass to fit the wooden frame and placed a Samsung LCD TV monitor within the frame and under the plexiglass. This way, we can simulate touch screen functionality and still place actual objects on the surface of the table itself.

It is important to note that, due to restrictions on the cost of a waterproof touch screen that would be responsive to objects such as coffee mugs, we implemented a pseudo-touch functionality to simulate the desired actions that would take place on placement of a beverage container on the screen by a user. We implemented the screen with a Samsung LCD TV Monitor and covered the screen with plexiglass to allow for the placement of coffee mug. To simulate the touch functionality, we recreated coffee mug placement using the “mouse click” event. We have a coffee icon on the dashboard that, when pressed, allows the user to simulate object placement by clicking on a region of the screen where the object would be placed. Depending on which of the four hemispheres the click takes place in, the on-screen text will respond and move accordingly so that it stays in view and is not obstructed by the newly-placed mug.

Core Algorithm:

The core algorithm of our solution relies on the inputs of the user. The user presses an activation push button wired to the breadboard of the Raspberry Pi, and the display activates. A graphics module drives the initial background display. The hard-coded information regarding email addresses for email and event requests, weather queries, and news sources to crawl is used by the main driver module (display.py) to display the information in a formatted text representation on the display. The main module utilizes a web scraping module, “smartCoffeeTable.py”, where all the user-specific information is stored for use in API Requests. We use the Google Email API for the gmail account setup for our Raspberry Pi. Through the same API, events can be pulled from the active Google account’s calendar data. We pull weather information from weather.com using the Google Weather API as in interface. We use the Python package, Feedparser, to parse data from the New York Times World RSS feed for display. We also display the time in the center of the screen. To simulate touch functionality, there is a small coffee icon in the center of the screen. When the user clicks this icon, the user is now in placement mode and can click anywhere on the screen to simulate a placement of coffee in that location. After a prespecified amount of time in display.py, the screen will return to black to await the input of the user to update the user's widgets and redisplay the home screen.

Software Structure Diagram:


NY Times Server
Weather.com Server
Google Server
NY Times RSS Feed
Google Weather API
Google Email/ Events API
*NOTE: The Python package "Feedparser" requests data from RSS Feed and parses in Client
Router/ Network Gateway
Smart Coffee Table Monitor
Raspberry Pi Client Application
Raspberry Pi Push Button Sensor Input
*NOTE: The user presses the push button for initial input.

Testing:

______We tested the Smart Coffee Table by putting through a "simple usage scenario" test case. We mounted the components and setup the monitor, frame, plexiglass, Raspberry Pi, etc. in a well-lit, spacious room for testing. We had our test user sit down at a couch positioned behind the table. The testing of the wooden frame and plexiglass was the first test. We placed the monitor on the frame and the frame held up firmly. We then placed an empty coffee mug on the plexi glass to test the glass for stability (a true hold-your-breath moment). One those tests were successful, we had the test user push the button sensor on the breadboard of the Raspberry Pi (located below the monitor). We ran the display script in the command line with the command “sudo python display.py”. This will produce a black, dormant monitor state where the program will wait for a user input to begin displaying graphics. Upon pressing the button in test case, the graphics displayed as intended. We tested the information by testing the produced information against the source information on the various web interfaces of the respective APIs (Google, NY Times, etc.). They all matched the original requested data. Visually, upon conducting this test the graphical text displays were positioned and styled correctly. To test the touch screen simulation functionality, the test user pressed the coffee cup icon in the center and selected with region (upper left, upper right, lower left, lower right) he wanted to place the cup. The test user started with the top left hemisphere. He clicked in the left hemisphere (to simulate coffee cup placement), and he also placed the coffee cup over the mouse cursor to physically act out the simulation with an actual cup. We tested the sleep functionality (after a specified time, the screen will return to black and “sleep”) by waiting 60 seconds (no user actions). The screen returned to black successfully, and we had the test user re-press the push button. This updated the information on the screen (we sent a new email and added a new event) and redisplayed the graphical display in its original format. We repeated this process for the remaining three regions of the display and they all tested out successfully.

Looking Forward:

_____If we were to continue to expand on this work in the future, we would first look at the cost of purchasing a full-sized, waterproof touch screen to actually respond to a placement of a coffee cup directly and in real-time. The lowest price of available models was too high for this project, but alternatives could be found in the future. We would also like to move forward with the implementation of the fingerprint sensor to act as a security measure and confirm that the primary user is in fact using the Smart Coffee Table. We were unable to do it in this project’s timeline due to some soldering needed to make proper connections with the breadboard wires.