Household Device Ecosystem

Team 33 - Ian Goodwin, Sam Atac, John Armgardt
ECE 445 Project Proposal - Spring 21
TA: Ali Kourani

1 Introduction

1.1 Objective

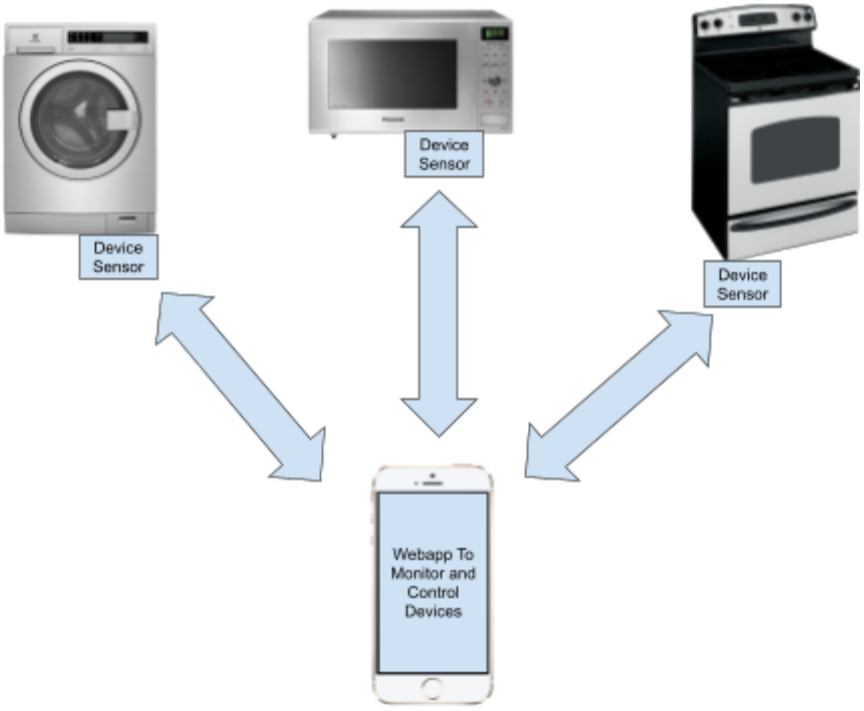
In the near future, as 5G networks roll out, IoT devices are predicted to become an ever more important facet of modern life. It is predicted that by 2025, there will be 75 billion IoT devices in use across the globe, an increase from just 7 billion in 2018[1]. As this rapid transition unfolds, consumers will begin to expect to monitor and control all of their household devices remotely. While there is no doubt that many top end future household devices will come with IoT functionality baked in, the vast majority of people will not own one of these devices for many years to come. Microwaves can last up to 10 years, Washing machines up to 16 years, and Stoves up to 23 years[2]. Clearly these often expensive items will last far into the IoT revolution, and thus would benefit greatly from external IoT upgrades. One other issue of the IoT revolution is device compatibility. As IoT devices have exploded in popularity, the number of options for smart device ecosystems have increased drastically. Most of these ecosystems are “walled gardens”, only allowing devices from their own brands to be added to their network[3]. This limits potential options for the consumer, and only allows connection to the device types offered by that particular brand.

Our goal to address these issues is to develop a modular IoT ecosystem with easy to use sensors. These modules would allow for monitoring/control of existing household devices, without the need to buy dedicated smart devices or replace existing household appliances. Our ecosystem would also be open source and as modular as possible, in order to allow the user to accommodate addition of IoT devices not designed by us. This open source philosophy would help to make the best use of any customers current IoT device setup.

1.2 Background

The current IoT device landscape is very fractured, with certain brands tending to focus on specific sectors. The security sector is led by brands like Ring and Nest[4]. The smart electronic sector is the most mature, with major tech brands such as Google and Apple leading the way[5] and The appliance sector is currently very split, with offerings from some major brands like Samsung[6]. These devices, however, are generally stand alone, and no major ecosystem exists to connect them. Our device offerings as a platform thus have no direct competition.

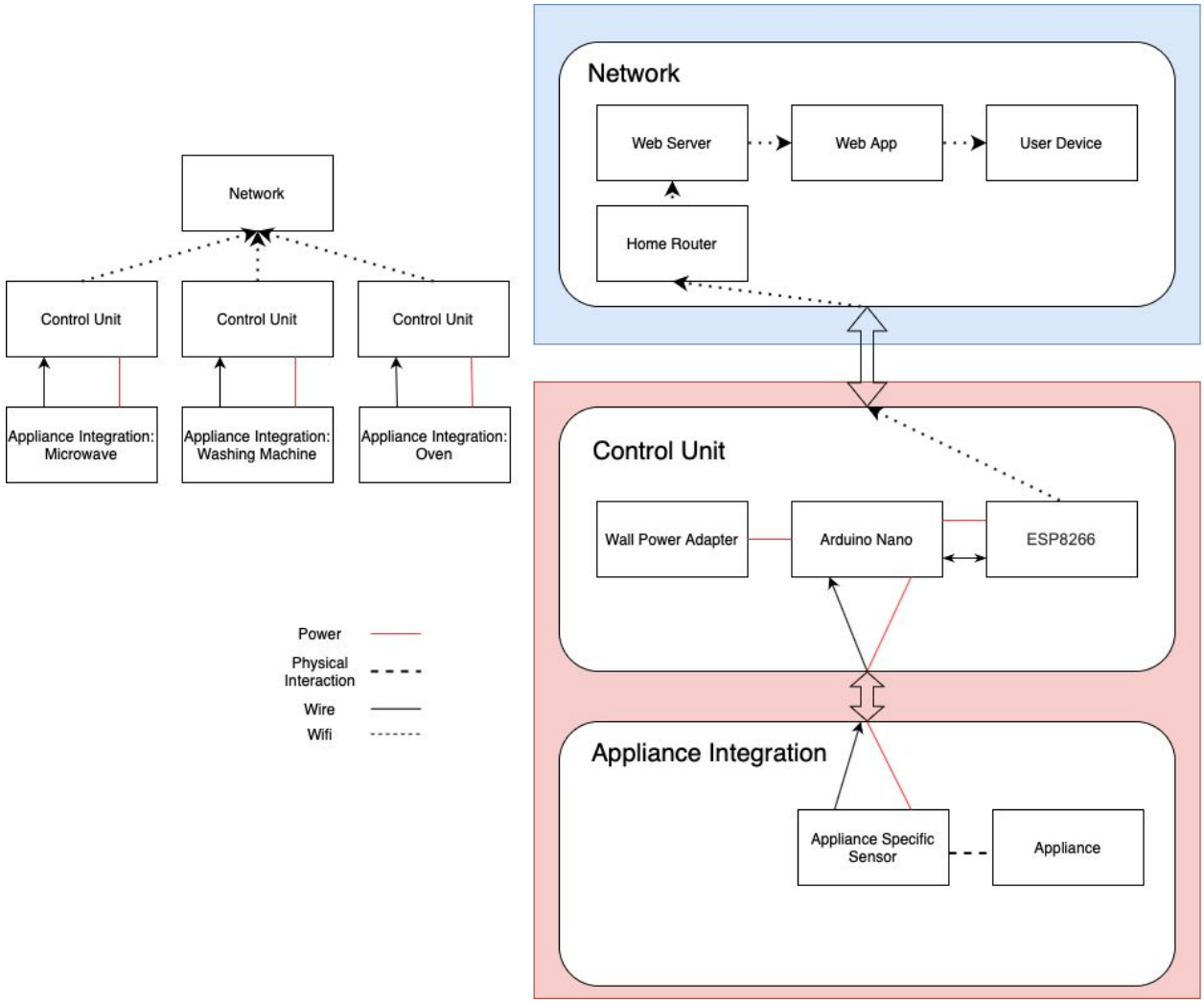
1.3 Physical Design



1.4 High-Level Requirements List

- Interface with at least 3 appliances
- A case to protect the hardware
- Web App to read the data remotely

2 Design



2.1 Network

The network unit controls how the user interfaces with the sensor data. When a sensor's state changes, the control unit will send a packet to the web server, and the app will handle and display the data accordingly on the user's device.

2.1.1 Home Router

The home router handles the transmission of data between the microcontrollers and web server. We will be using our router already at home, a TP-Link Deco.

Requirement: the home router can transfer data over WiFi at a minimum of 5 Mbps

2.1.2 Web Server

The web server is a simple server that handles sending and receiving requests between the microcontrollers and the web app. We will set it up as an AWS EC 2 server, and run a Node.js web server on it.

Requirement: the web server can hold up to a minimum of 500 MB of data

2.1.3 Web App

The web app is a simple browser application designed using HTML and JavaScript. It will display the status of the existing sensors, allow a user to add/remove sensors, and interact with the sensors themselves if the device allows.

Requirement: the web app runs the latest version of JavaScript to properly retrieve and handle data

2.1.4 User Device

The user device is the computer the user accesses the web app on. We will be testing our application on a Windows 10 PC.

Requirement: the user device has an updated browser capable of running the latest versions of HTML and JavaScript

2.2 Control Unit

The Control Unit consists of the modules that power the local devices and accept data from the sensors, packages that data and sending it out to the home router over wifi.

2.2.1 Wall Power Adapter

We will be powering each of our Control Unit subsystems off a 5v wall power adapter that will plug into our Arduino Nano.

Requirement: the wall power adapter can output a minimum of 5V

2.2.2 Arduino Nano

Each of the arduino nanos will power their ESP8266 and Appliance-Specific Sensor as well as accept data from the Appliance-Specific Sensor, format it and then provide it to the ESP8266 for transmission.

Requirement: The Arduino Nano can accept data from application specific sensors
Requirement: The arduino Nano can communicate with and provide data to the ESP8266

2.2.3 ESP8266

The ESP8266 boards will accept data from the Arduino and then transmit it to the server over wifi.

Requirement: the ESP8266 can send requests to web server

2.3 Appliance Integration

The appliance integration block consists of the physical components that are attached to devices we monitor, and the actual devices themselves. Our initial goal is to create a sensor for an oven, a washing/drying machine, and a microwave.

2.3.1 Appliance

The Appliance is a device that the consumer already has that they wish to have more data from remotely. We will be initially supporting a washing machine, microwave and oven.

2.3.1.1 Microwave

The microwave we build a sensor for will be a standard microwave oven. A user can place food inside the microwave and run it for a desired amount of time.

Requirement: the microwave is capable of outputting a minimum of 800W

2.3.1.2 Oven

The oven we build a sensor for will be a standard kitchen oven. A user can place food inside of it and have it heated to a set temperature. The oven will have a preheating feature to allow the user to set the desired temperature.

Requirement: the oven is capable of outputting a minimum of 400°F inside the oven.

2.3.1.3 Washing/Drying Machine

The washing/drying machine setup we build a sensor for will be a standard version of these appliances. The washing machine will have a door lifted from the top end of the appliance, and the dryer will have a door opened from the front.

Requirement: the appliances are able to run for at least 20 minutes each

2.3.2 Appliance-Specific Sensor

The appliance specific sensor is the electronic component that is attached to the monitored device. It will be wired into its respective Arduino Nano for power and for data transmission.

2.3.1.1 Microwave Microphone

The microphone will be placed outside the microwave, but close by so that it can pick up on the microwave's noises. It will monitor the microwave's interior motor noises to determine if the microwave is currently in use, and send this data to the control unit. It will also monitor for louder alerts used to indicate the microwave's completion.

Requirement: the microwave is sensitive enough to monitor a microwave's noises

2.3.1.2 Oven Thermometer

The thermometer will be placed inside the oven, and be wired to the outside of the oven into the control unit. The thermometer will monitor the oven's temperature, and send this data to the control unit.

Requirement: the thermometer is accurate enough to monitor internal oven temperature
Requirement: the thermometer is durable enough to withstand at least 500°F

2.3.1.3 Washing/Drying Machine Accelerometer

The accelerometer will be placed on a side of the washing/drying machine and sense the machine's movement. It will monitor how much the machine is moving to determine if it is still in use, and send this data into the control unit.

Requirement: the accelerometer is sensitive enough to monitor a washing/drying machine's movement

2.4 Risk Analysis

Our primary source of risk comes from the appliance-specific sensors. They must be sensitive enough to report the data we are tracking, be module enough to be wired into the control unit, and be durable enough to maintain working status after a healthy lifespan. For example, we may obtain an inaccurate accelerometer that always outputs movement, and constantly says the washing/drying machines are in use. Furthermore, we may obtain a flimsy thermometer that cannot withstand the internal temperatures of the oven, and may break after short use. As such, we will need to properly research quality sensors for our project, and handle them with care. The requirements listed above will be sufficient to ensure our sensors will be of high enough quality to function properly.

Our secondary source of risk comes from the network block. Due to the imperfections of wireless internet, we may experience issues in getting our control unit and our router to connect. Certain issues include packet loss, impedance, improper setup, etc. As such, we will choose WiFi-capable microcontrollers with large range to ensure connectivity is maintained. We will also initially test our devices at a short range, and increase it to test the boundaries the devices can reach. If necessary, we will look into obtaining network extensions to maintain a higher quality signal.

3 Ethics and Safety

Our primary safety issue lies within the handling of electronic components. We will be plugging in various components into a microcontroller, and that into a wall outlet. There is a risk of electrocution present, although unlikely. As such, we will practice standard safety procedures when handling electricity, such as powering off the device when tinkering, using grounded equipment, etc. Once the device is fully built, we will construct a protective case to ensure the device cannot be easily tampered with, and users do not come in contact with electricity. As such, we see no safety concerns outside those of a standard electrical appliance upon completion.
Our primary ethical concern lies within security. ACM Code of Ethics 2.9 states that we are to “Design and implement systems that are robustly and usably secure”. [7] This requires that systems have security features to ensure misuse and modification don’t occur. As such, we will implement a key system for our web server API. This will ensure that the web server only sends requests to and receives requests from verified devices. If the web server attempts to connect to a bad key, it will ignore that device. This protocol will ensure that only the device itself is communicating with the server, and the data is not altered in transmission.

References

[1]
https://securitytoday.com/Articles/2020/01/13/The-IoT-Rundown-for-2020.aspx?Page=2
[2]
https://www.mrappliance.com/expert-tips/appliance-life-guide/

[3]
https://www.concannonbc.com/walled-gardens-are-killing-the-internet-of-things/

[4]

https://www.pcmag.com/picks/the-best-smart-home-security-systems

[5]
https://www.pcmag.com/news/the-best-smart-home-devices

[6]
https://www.pcmag.com/picks/the-best-smart-kitchen-appliances

[7]
https://www.acm.org/code-of-ethics