The Report Committee for Andres Humberto Jaramillo
Certifies that this is the approved version of the following report:

Water Heater Smart Control and Management

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor: ______________________________
Ross Baldick

____
Jesse Mee

Modest: Water Heater Smart Control and Management

by

Andres Humberto Jaramillo, M.S.E.; M.S.

Report

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

The University of Texas at Austin

May 2017

Abstract

Modest: Water Heater Smart Control and Management

Andres Humberto Jaramillo, M.S.E.

The University of Texas at Austin, 2016

Supervisor: Ross Baldick

Recent advances in microcontroller technology have given rise to the Internet of Things (IoT), which has created new opportunities in energy automation and optimization. Smart devices and appliances are able to collect operational data, and combine it with data analytics to facilitate optimized control of household electronics. For example, data collected from a smart thermostat can be combined with third-party data sources such as; 15-min interval electricity usage data collected from a smart meter, weather data from numerous sources, utility billing rate data, and data from other smart appliances in the home to optimize the energy usage of residential heating and cooling system.

The goals for this project are: 1) Design a control system that enables collection of temperature, flow rate and power data, 2) implement the control system in an electric water heater for residential use, 3) create a connection between a cloud server and the water heater control system for data collection, and 4) develop an advanced control algorithm for water heater thermal management. The main contribution is meant to be a cloud-based infrastructure for IoT aggregation and control, a kind of artificial intelligence for IoT and energy automation.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

Recent advances in microcontroller technology have given rise to the Internet of Things (IoT), which has created new opportunities in energy automation. Smart devices and appliances are able to collect operational data, and combine it with data analytics to facilitate optimized control of household electronics. The Nest smart thermostat, for instance, is able to report the indoor ambient air temperature, thermostat set point temperature, and the on/off cycles of the HVAC system such as active heating, active cooling, or off. By itself this data is of limited value; however, when combined with other data sources it can become very powerful. For example, the smart thermostat data can be combined with third-party data sources such as 15-min interval electricity usage data collected from a smart meter, weather data from numerous sources, utility billing rate data, and data from other smart appliances in the home to optimize the energy usage of residential heating and cooling system.

Analytics can be performed on this data, yielding various models, from which we can leverage, smart appliances for energy savings, which results in lower utility bills. Rather than regulating a thermostat according to a memorized schedule, for example, we can regulate the temperature based on energy usage patterns in order to meet a monetary budget for the billing cycle. We can do other things as well, such as forecast the cost of operating the appliance and present the user with a price beforehand. Consumers want to save energy but it is difficult for them because most of the energy saving solutions require a substantial capital investment. As such, other creative and more advanced alternatives for saving energy are preferred. Companies often offer energy savings through increased insulation, which is costly and only effective to a certain degree. Furthermore, the water heater set point is not as accessible and user friendly as the air-condition thermostat. Through data and automation consumers can be empowered to manage their consumption by way of simple feedback.

The purpose of this project is to develop a control system that takes a conventional "dumb" electric water heater and makes it "smart," in some respects comparable to a Nest controller but for water heating instead of HVAC. While many smart HVAC thermostats exist on the market, there is no smart water heater technology to speak of despite it being the second largest energy load in an average home. The scope of the project is focused on the embedded aspects of the problem, the circuit design, firmware development, and supporting server based cloud infrastructure in order to facilitate the necessary data acquisition and control. A mobile client is also developed to demonstrate a fully functional system with user interaction.

The long-term aim is not to get in the business of competing with appliance manufacturers, but to provide technology to support those existing manufacturers, this technology would be retrofit to current units to make them Smart, kind of like a Roku can make a dumb TV a Smart TV. The main contribution is a cloud-based infrastructure for IoT data aggregation and appliance control, a kind of artificial intelligence for IoT and energy automation. Existing manufacturers would then integrate it into their appliances, a turnkey solution that allows them to offer "smart" capability with their existing products, and also offer a level of service that would be beyond their in-house capabilities. In this sense, the project also serves as a showcase for appliances manufacturers, in the long-term potentially turning into an embedded development kit that could be provided to engineering teams at those organizations to aid in their development. For example, when displaying appliances in a store, customers can not only see a price tag for the item, but also a tag that displays the average electricity bill or energy consume by the appliance. This will be useful because customers can not only compare the physical and performance features that are normally given but also the potential savings in energy when using them at the same rate.

Chapter 2: Circuit Design

Circuit design progressed through three stages: Requirements, MCU Selection, and Sensor Integration. Requirements and MCU selection are strictly affected by the type of water heater selection (electric vs gas). The sensor integration is done after the sensor selection is obtained and all the data sheets of the former are gathered; therefore, it is important to start the discussion with the water heater sensors and desired control since it directly affects the circuit design process.

2.1 Water Heater Sensors and Control

One of the early design challenges for the project was to decide which water heater type will be the subject for testing. Gas heaters are cheaper in terms of price per energy unit, also they have a higher recovery rate which lead to higher temperature and pressure ratings adding more safety control to the unit itself. On the other hand, electric heaters are more expensive with lower recovery rate leading to a bigger tank and less safety controls. It was decided to do the project design around the electric heater due to the potential for greater energy savings and higher usage among US residential houses. Although the initial design will target an electric water heater, it is noted that many of the components and logic behind the control will be the same for a both the electric and gas heater designs. The primary difference will be that method by which the energy consumption will be measured (flow rate of natural gas compared to electric current). The electric heater design will aim to be flexible and scalable to accommodate a future gas heater design.

Another critical concept in this project is the type of thermostat the water heater is equipped with, and how it will affect the design. In the case of electric water heaters, the unit has two heating elements that can be controlled to provide more effective heat transfer rates.

The non-simultaneous thermostat control is the most common setup found on the water heaters. Here the upper element is turned on first during the beginning of the cycle. Once the water in the upper level has reached the temperature set point, and then the lower heater element turns on providing the desire bulk temperature in the tank. During standby cycles only the lower element comes on. This thermostat logic control is known as the non-simultaneous cycle, which is the most common water heater residential set up. This is an exploitable energy savings control since the hot water by density changes always accumulates in the top. Therefore, it is more efficient to use the upper heater element when energy savings is a priority, which will allow for less energy usage when the lower heater element turns on. This can be seen in Figure 1, when the tank is fully heated only the heater in the bottom part turns on in order to maintain the temperature set point while the water is not being used [1].

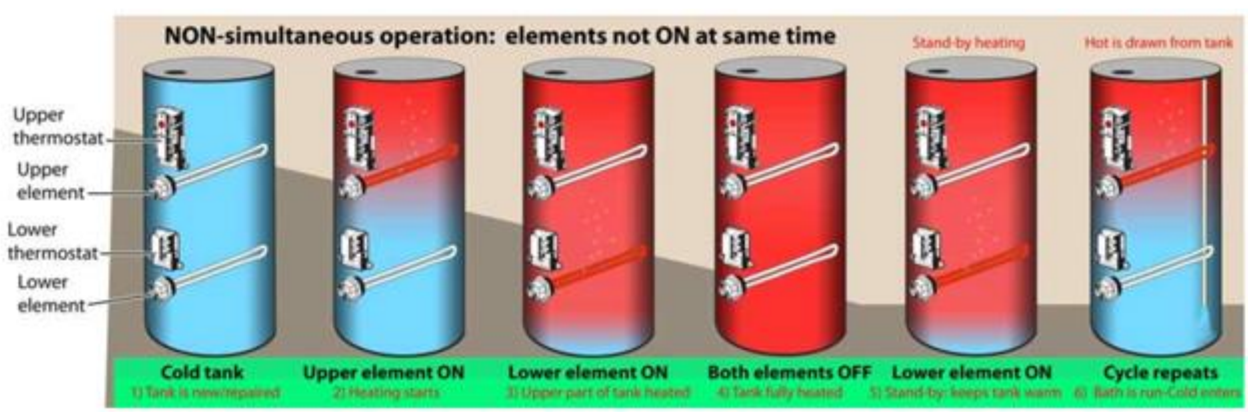

Figure 1. Non-simultaneous Water Heater Operation

For the sensor placement and selection some designs considerations were accounted for. For example, at first a combined flow/temperature sensor was desired due to its simplicity, less circuitry work and easier data collection; however, these sensors are fairly expensive due to the temperature compensation control for the flow meter. As mentioned before, a water heater is a simple concept where hot water is stored in a tank and filled at the same rate that it is being used; therefore, the use flow rate is directly related to the source flow rate. This assumption allows for a cheaper flow rate sensor. An electromagnetic type Hall-effect sensor was selected for the flow rate meter because it provides accurate readings, low voltage pulses and low price.

Accurate readings are not necessary for temperature since the temperature range scale is fairly big and the changes are far from instantaneous; therefore, a thermocouple is best suited for temperature measurements. A J-calibration thermocouple was selected because it better suits the temperature range of the water heater (0 °F to 180 °F). The other calibration types (K, E, and T) are for more extreme temperatures.

Current transformers (CT) are sensors used to linearly step down the current passing through the sensor to a lower level compatible with measurement instrumentation. The power meter was selected to be a CT device because CT devices are inexpensive and very effective at measuring power consumption through drain current rates. Lastly, an electromagnetic single pole double throw (SPDT) relay was selected for the thermostat on and off control.. The exact following sensors were selected:

● Temperature sensor is a Jcalibration thermocouple 12" ungrounded probe, type J, 304 SS sheath 1/4

Figure 2. J Omega Thermocouple

● Flow meter is a Hall effect sensor type with accuracy up to 0.1 gallon per pulse

Figure 3. Hall effect flow meter

● CT energy meter is a 120/240 Volt Pass-Through kWh Meter, 3-wire, 100A, 60Hz

Figure 4. CT Energy Meter

● Relay is a miniature heavy duty DC electromagnetic SPDT

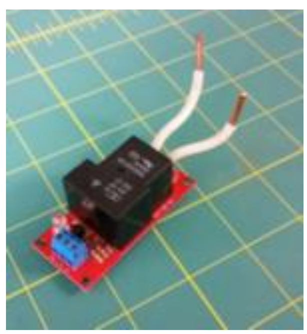

Figure 5. SPDT Electromagnetic Relay

Once the sensors arrived they were tested to ensure they would provide the feedback needed for the microcontroller circuitry design. For example, the flow meter was installed to a water hose for flow water simulation and the output was connected to an oscilloscope that captured the output pulses given by the instrument. In the flow meter case, the flow display was compared to the amount of pulses received to verify correctness of the sensor. The thermocouple was tested in the same way; however, this test revealed low voltage output from the sensor. This had to do with the big range of the thermocouple compared to the low operation range of the water heater (0 °F to 170 °F). This led to obtaining another piece of hardware for amplifying these signals, which is described in greater detail later. The energy meter and the relay do not need any testing since their outputs and inputs are very straightforward to verify onceinstalled. The power meter will show power measurmenet when the heater is on which can be verified when the the temperature readings are below the set point. The relay will be tested sucesfully when it turns off the thermostate.

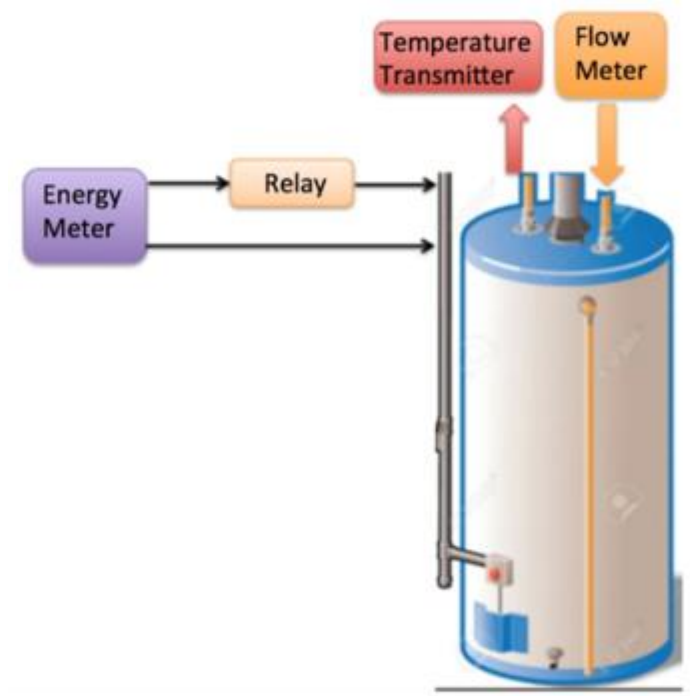

Figure 6. Sensor Layout

The sensor installation locations, as described above, were selected based on the location they will be placed. The thermocouple was installed in the discharge piping side of the water tank so a T coupling was added to allow the sensor be in contact with the water leaving the water heater. The flow meter was installed in series with the suction or upstream piping (cold water flows into the water heater) of the water tank. The power meter was connected directly to the thermostat feed power and SPDT relay in series. For the energy meter, special circuitry design was needed due to its high voltage, which could damage the microcontroller. This design is explained in greater detail in later sections. The experimental set up was done in Energy Bill's lab and then it was set up in a residential house with 2 water heaters. One in the roof of the house exclusively dedicated to the 2 bathrooms in the second floor and the other on the first floor for a single bathroom and the kitchen.

Figure 7. Sensor installation setup

2.2 MCU Requirements

After selecting the sensors, the research began for an appropriate microcontroller to base the control circuit design on. The first step of this process was to develop the following list of requirements:

● Power meter (EKM Metering brand) generates 800 pulses per kWh. It has a max rating of 240V @ 100A, which is 19,200 pulses/hour at max rating, or 320 pulses/min, 5.33 pulses/second. Rounding up, at max power this is potentially an interrupt rate of 6Hz.

● The flow meter generates 10 pulses per gallon. A liberal high-end estimate for max flow is 10 gpm, which would be 100 pulses/min, 1.67 pulses/second, so max interrupts of 2Hz.

● Thermocouple A/D converter for temperature readings requires SPI interface.

● Relay control circuit requires standard digital pin for actuation.

● Internet connectivity over Wi-Fi (802.11g), TCP client, and ability to communicate in HTTP.

● Internet connectivity can be unreliable; there is no guarantee that the controller will always have access to the Internet. Due to this we need onboard storage ability to save data readings locally. Usage of a micro SD card for this, which also requires an SPI interface.

● Highly accurate real-time clock. MCU needs to be fast enough so that the drift is minimal and won't affect our timekeeping when collecting data, and also fast enough so that interrupts are not out of sync with the clock.

● Additional IO pins for peripherals we may wish to add down the road, like an LCD screen or buttons for human interface.

● At this time algorithms are not being executed locally on the controller, however, in the future after algorithms for smart controllers have been developed, it may need to deploy some light-weight versions of those models locally, so that the controller can continue to operate "smartly" offline if it loses internet connectivity.

2.3 MCU Selection

These requirements led to the selection of the CC3200 microcontroller by Texas Instruments. It provides ample processing power with an ARM Cortex-M4 running at 80MHz with plenty of room to grow. It easily supports the interrupt rates that are required for data acquisition and has plenty of IO for other necessary peripherals like SPI. The biggest selling point was that the CC3200 has built in Wi-Fi capability, a single chip solution with an ARM Cortex and Wi-Fi in one IC. For the purpose of this project the CC3200 Launchpad was selected as the main control board to avoid prototyping a new PCB.

	P1		P3	
Ref	Signal	Dev Pin#	Dev Pin#	Signal
1	3.3V			5V
2	ADC_CH1	58		GND
3	UART0_RX	4	57	ADC_CH0
4	UART0_TX	3	60	ADC_CH3
5	GPIO	61	58*	ADC_CH1
6	ADC_CH2	59	59*	ADC_CH2
7	SPI_CLK	5	63	AUD_SYNC
8	GPIO	62	53	AUD_CLK
9	I2C_SCL	1	64	AUD_DOUT
10	I2C_SDA	2	50	AUD_DIN

P4		P2		
Signal	Dev Pin#	Dev Pin#	Signal	Ref
PWM	2*		GND	1
PWM	1*	18	GPIO	2
PWM	17*	8	SPI_CS	3
PWM	64*	45	GPIO	4
CCAP/GPIO	21*		RESET_OUT	5
CCAP/GPIO	18*	7	SPI_DOUT	6
GPIO	62*	6	SPI_DIN	7
GPIO	60*	21	GPIO	8
GPIO	16	55	GPIO	9
GPIO	17	15	GPIO	10

Figure 8. CC3200 Launchpad and IO Mappings

2.4 Sensor Integration

The final phase of the circuit design was to integrate our sensors and other peripherals to the MCU circuit. This required additional circuit design for many components.

● Power meter: An additional design was added due to the high voltage output from the meter in order to protect the circuit against any faults. The opto-coupling circuit was designed to decouple the power meter from the control circuit. High voltage DC signals around ~27V are converted to a 3.3V digital logic signal. When this happens the signal is inverted, with it being normally high. A function generator and oscilloscope were used to test the circuit and validate its intended behavior. The circuit worked perfectly over the necessary frequency range (0 Hz – 10Hz) for our ~6Hz power pulses.

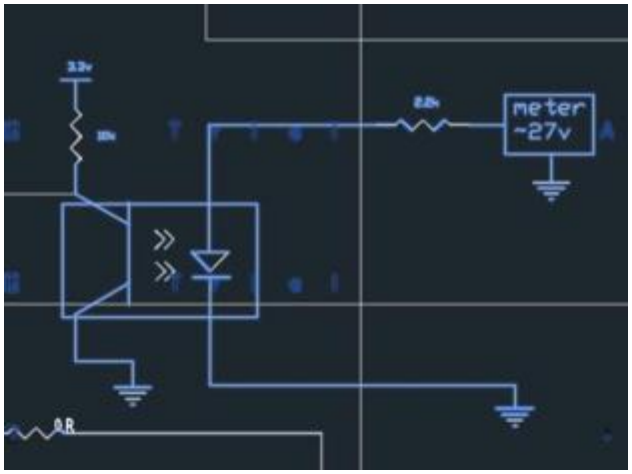

Figure 9. Optical Coupling Circuit

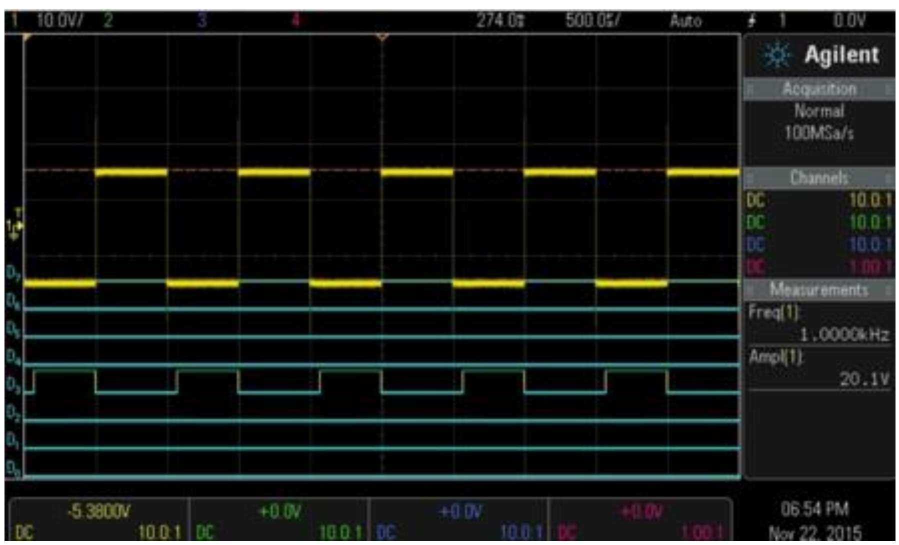

Figure 10. Oscilloscope Pulse Experimental Testing

● Relay: The following transistor based circuit was used to interface the relay with the MCU. An NPN transistor is driven by a digital input at the gate, which causes current to flow through an inductor in the relay, which then actuates the load electromagnetically. A bench top power supply was used to test the performance of this circuit. A breakout board was used to for soldering this circuit.

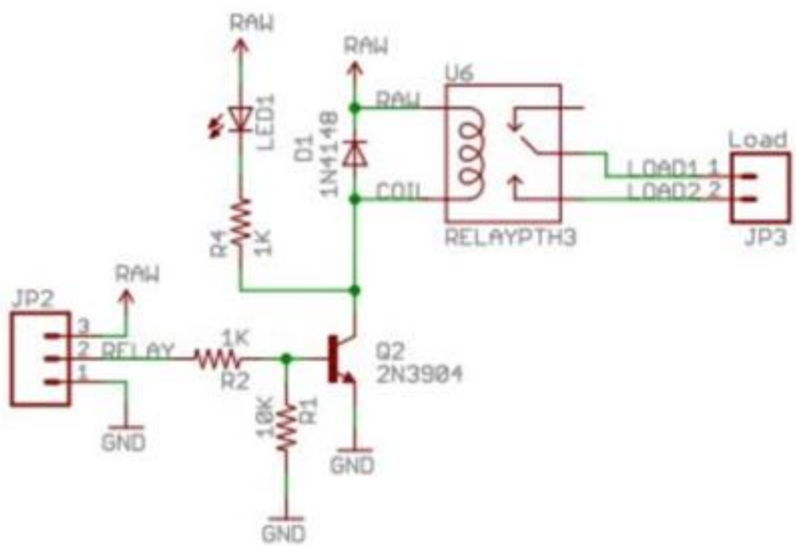

Figure 11. Relay Control Circuit

● Water meter: The water meter was a very simple integration as a Hall-effect switch. It was simply wired straight to the 3.3V power plane with a pull-down resistor to GND at the output end.

● Thermocouple and SD Card: The SD card and MAX Thermocouple-to-Digital converter are both SPI devices. They were connected to the CC3200's SPI peripheral with separate chip select pins being wired to different IO ports.

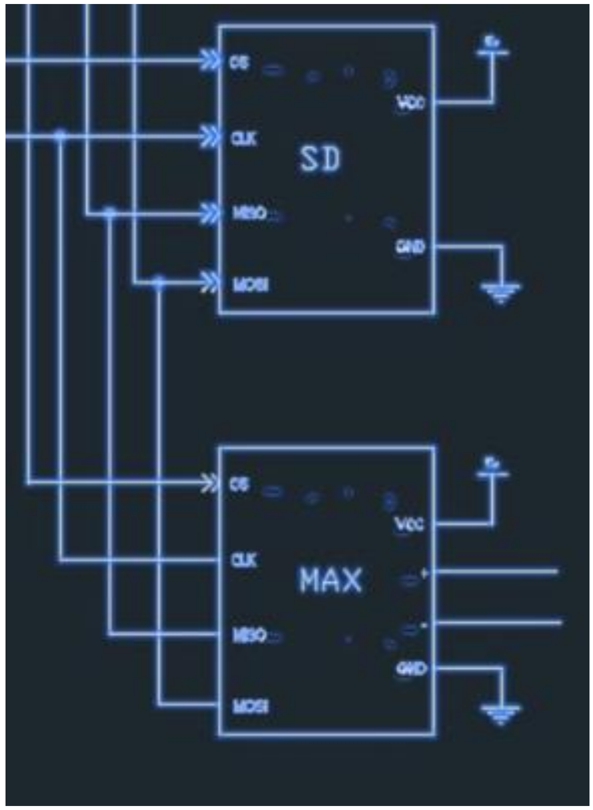

Figure 12. MicroSD card and Max Thermocouple SPI Peripherals

● Power Regulation, Signals Board, and Testing: A 5-35V to 5V DC linear regulator was used to provide power for the circuit. This circuit, along with the opto-coupler and all other signals pins were integrated to a signals board to aggregate all the inputs for the circuit and bring them to the controller board. The final circuit was then constructed and tested. Sensor signals were simulated with a function generator and tested with a logic analyzer and oscilloscope.

Figure 13. Oscilloscope Pulse Function Feeding

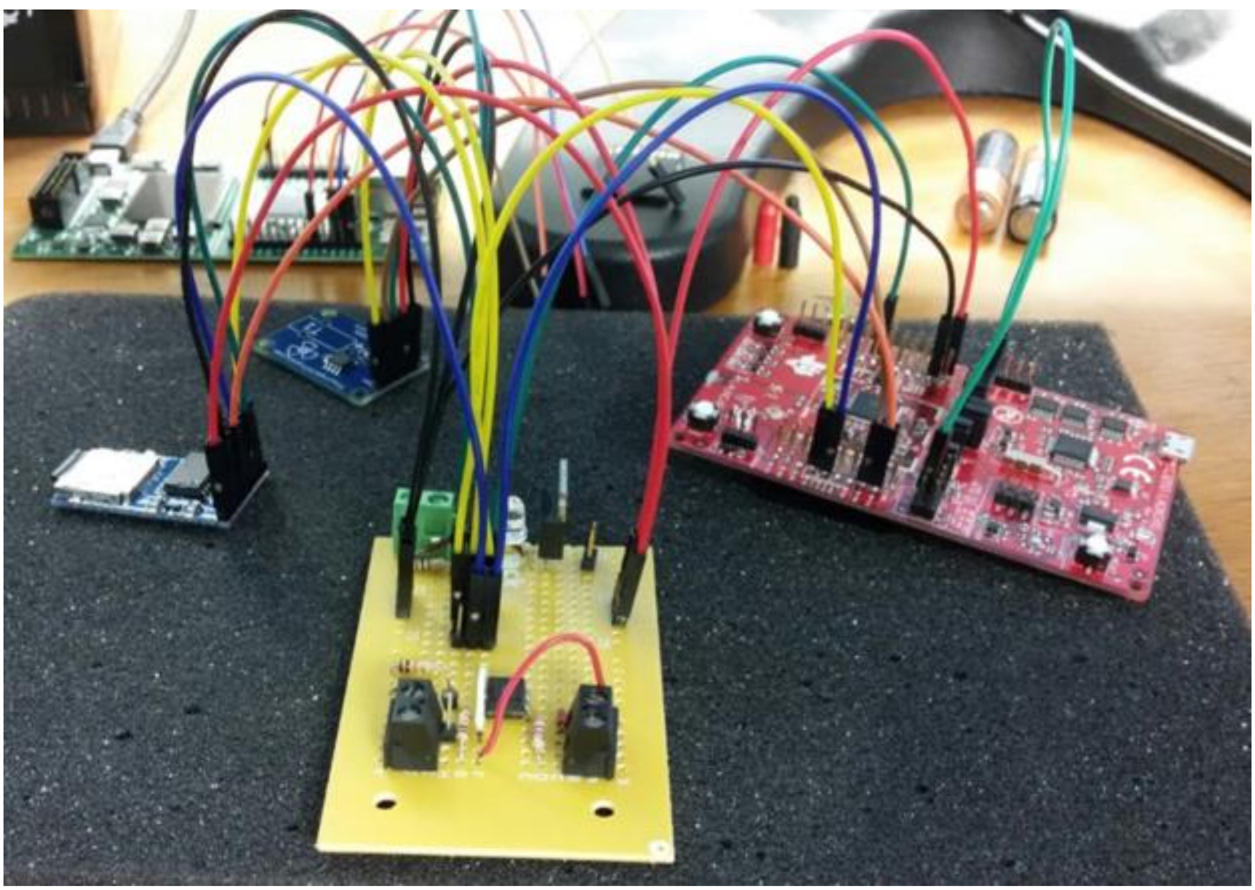

Figure 14. Full control circuit wired (minus relay in photo)

Chapter 3: Firmware and Software Design

Figure 15 shows the high level architecture of the overall system and control circuit. There are three applications that comprise the system: Firmware / Smart Water Heater Device, Server, and Mobile Client.

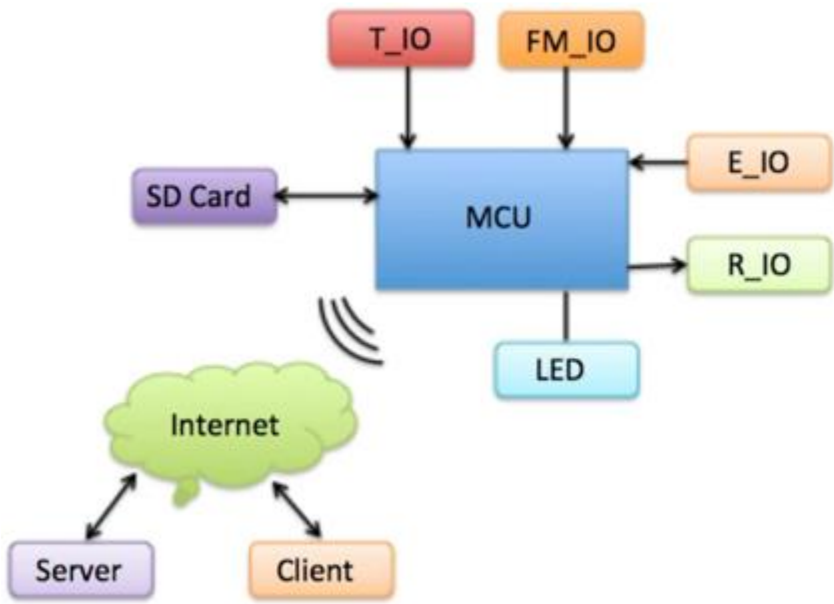

Figure 15. Circuit Architecture

3.1 Firmware Architecture

The following diagram in Figure 16 shows the circuit and system architecture at a high level. T = Thermocouple, FM = Flow Meter, E = Energy, R = Relay, and so on. The MCU connects to the Internet through Wi-Fi, where it will communicate with a server. A separate mobile client will then also communicate with the same server and through it be able to communicate with the MCU.

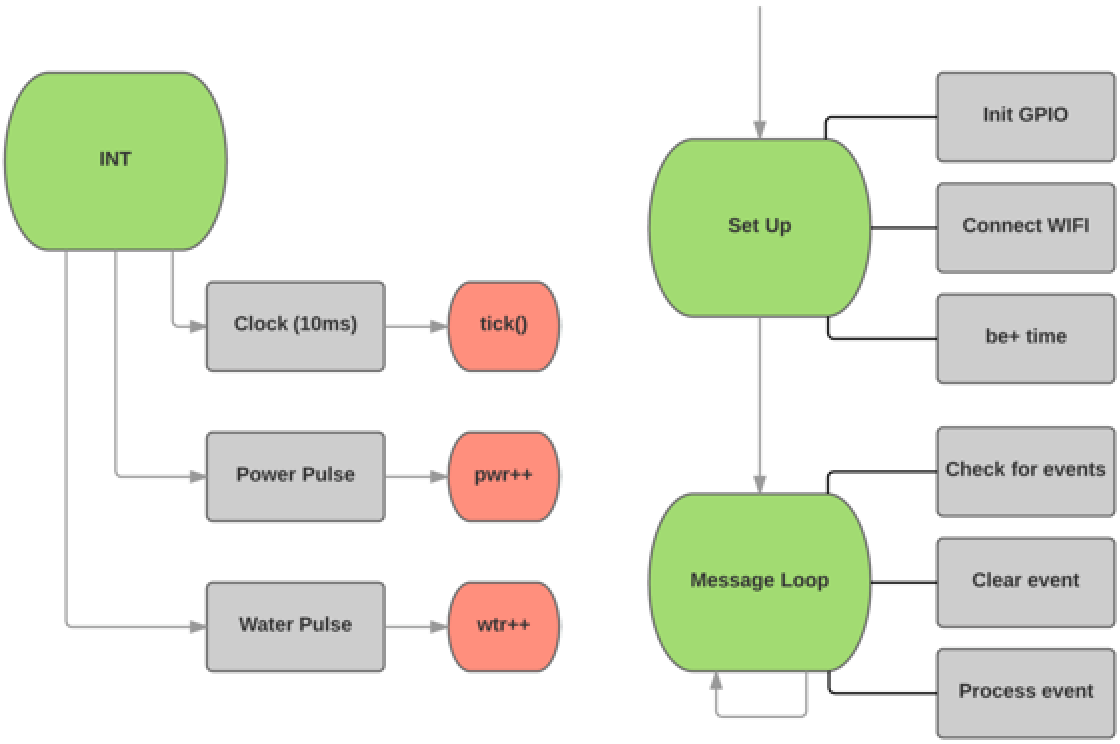

Figure 16. Firmware Architecture

The firmware application is a real-time operating system (RTOS) called WH1 ("Water Heater One"). Its primary and most important job is to collect accurate sensor data and maintain accurate timestamps for that data. It has three hardware interrupts, a timer interrupt that fires every 10ms for maintaining an accurate clock, and two interrupts for the power meter and water meter. These interrupts halt the main program loop to ensure that data is never lost. The timer interrupt executes a WaterTime, tick() routine, which is part of a date/time structure that represents our system clock. As ticks happen the WaterTime object increments variables for tenth, second, minute, hour, day, month, and year.

The WaterTime object then raises timekeeping events, which can be processed by the main program loop. WH1 follows an event driven design pattern. The main program loop continuously looks for event messages and then processes them if they exist. The clock object or the context of the main WH1 program can raise events. The following code snippet is an example of an event being caught and then processed.

```
//minute
 if(wc.Flags & EVENT_MINUTE)
 {
  //clear the flag
  wc.Flags &= ~(EVENT_MINUTE);
  //fire interval once/min
  g_ulFlags |= EVENT_INTERVAL;
 }
```

Figure 17. Code Event Sample

WH1 has the following events defined, in part. Some other events exist as well, but this constitutes the primary business logic of the WH1 operating system.

Event	Description
EVENT_KWH	A power pulse has been received, used to trigger a de bouncing chain to ignore superfluous interrupts.
EVENT_H2O	A water pulse has been received, used to trigger a de bouncing chain to ignore superfluous interrupts.
EVENT_SETPOINT	Application goes to the server and downloads the current set point temperature. This event runs once/min.
EVENT_REGULATE	Checks the current temperature vs the set point and actuates the relay to increase or decrease the water temperature.
EVENT_UPDATETEMP	Uploads the current water and air temperature to the server, this is an instantaneous real-time measurement as opposed to the 1-min average used with intervals.
EVENT_INTERVAL	Fires once/min on M=0, this event saves the 1-min running average for water and air temperature, as well as 1-min totals for kWh energy and gallons of water usage. Information is saved locally to the SD card, and then EVENT_UPLOAD is fired.
EVENT_TEMPERATURE	Fires once/sec and samples the instantaneous water and air temperature from the thermocouple. If water temperature has changed >= 1 degree EVENT_REGULATE is fired.
EVENT_UPLOAD	This event is fired by the INTERVAL routine and also fires on its own once/min. Scans SD card for pending data intervals that need to be pushed to the server. If they exist it uploads them to the server and clears the local cache upon successful confirmation of receipt from the server.

Table 1. Firmware Events

The boot-up procedure for WH1 follows the following pseudo code / steps:

1. Initialize TIMER interrupt or system clock

2. Setup LED IO pins

3. Setup Relay IO pin

4. Start UART for serial port interface and debugging

5. Set SPI and Chip Select IO pins

6. Setup Power Meter Hardware Interrupt

7. Setup Water Meter Hardware Interrupt

8. Initialize Wi-Fi Adapter

9. Connect to Wi-Fi Router, obtain IP address

10. Connect to server and get calendar date and time

11. Connect to server and get set point temperature

12. Connect to SD card and obtain client unique ID number

After the successful completion of the setup routine the program goes into the main program loop indefinitely and starts processing messages. Sensor data is collected, intervals created, data intervals uploaded to server, control settings download from server and water heater controlled through relay.

3.2 Server

The server is developed as a basic REST service in C# using the .Net framework and ASP.Net Web API. It is called Water Cloud and resides at http://water.energybill.com/api/ [22] with the following service endpoints exposed.

Endpoint	Method	Description
/time	GET	Firmware polls for current system time
/setpoint	GET	Firmware polls for set point temp setting
/setpoint	POST	Mobile client sets set point temp setting
/interval	POST	Firmware uploads single data interval
/dump	GET	Dump of all interval data, for public/web browser client
/temperature	GET	Mobile client fetches current temperature readings
/temperature	POST	Firmware uploads instantaneous temperature readings
/settings	GET	Mobile app gets current device settings

Table 2. JSON Parsing Methods

3.3 Mobile Client

A mobile app was developed for Android that allows a client end-user to remotely see the current water and air temperature, and manually adjust the set point thermostat temperature. This app is initially very simple and meant to be a framework for additional features in the future. The intention for the App is to evolve in such a way that it can display usage patterns for the water heater and other devices, including usage forecasting for the future based on those patterns, and provide real time pricing information, like a price tag, which changes depending on device settings like set point temperature. It is envisioned that Energy Bill App would regulate the water heater set point automatically to save energy, and the app would allow the user to provide feedback such as requesting an increase or decrease in hot water.

Figure 18. Water Heater Mobile App UI

3.4 Data Format

All data passed within the system is in JSON format. This is an example of a record obtained from the server and an interval record prepared by the firmware:

{"Time":"2015-12-03T03:56:12.403556-06:00"}

{"Id":"d95f6681-b99a-4aef-aa16-6d41c566cb5c","ClientId":2,"StampStart":"
2015-12-03T03:55:00","StampEnd":"2015-12-03T03:56:00","kWh":0.0,"Gall
ons":0.0,"TempWater":59.03302,"TempAir":17.781839,"TempSetpoint":65.
559998,"PowerState":60"}

Interval data consists of the following:

● ClientId = This field differentiate between the downstairs water heater (1) and the one upstairs (2)

● StampStart = Start time for data interval

● StampEnd = End time for data interval

● kWh = Energy in kilowatt hours

● Gallons = Water volume in gallons

● TempWater = Water temperature from thermocouple in Celsius

● TempAir = Room temperature from MAX thermocouple

● PowerState = Number of minutes relay is at HIGH state

● TempSetPoint = The set point temperature

3.5 Circuit Installation

Once the circuit components were tested and verified to be collecting data properly the circuit was installed in a plastic box to protect the circuitry from any water contact. This allowed for a safer installation of the microcontroller and secured it from children and the heating elements.

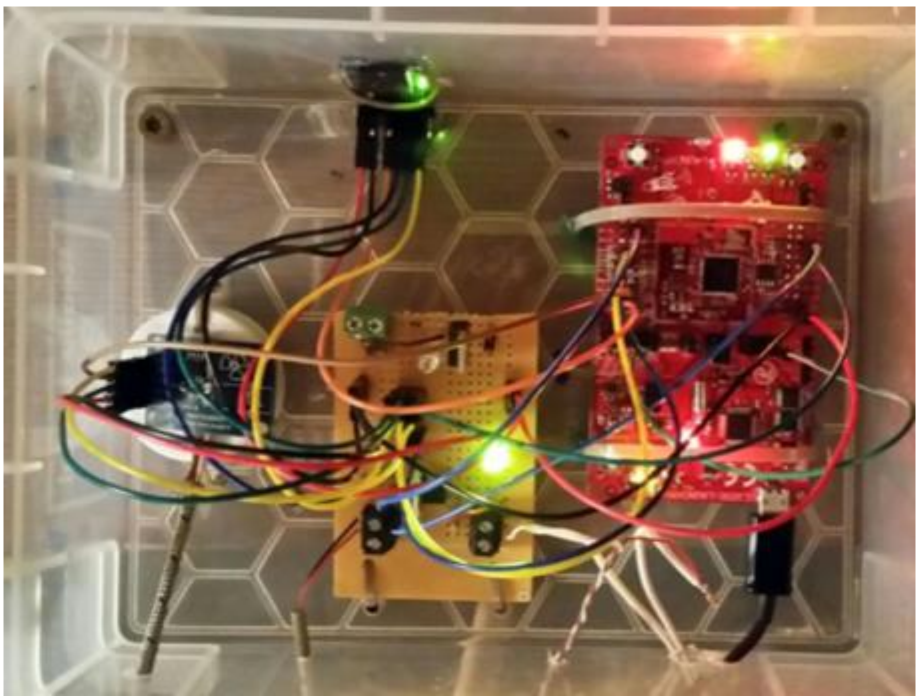

Figure 19. Circuit Container Design

Figure 20. Final Circuitry Installation (Left: Upstairs water heater, Right: Downstairs water heater)

3.6 Data Collection and Verification

Data has been successfully collected by the control system and pushed to the server. The system performs reliably, with both smart water heater controllers running for days without any time drift or loss of data intervals. No data discrepancies have been discovered as of yet. The mobile app also works and allows for monitoring and control of the water heater remotely. An analysis of the data initially collected follows.

After the server started collecting data it was important to perform some data analysis to ensure the data was being collected properly and at the right time. Figure 21 shows the relationship between water temperature (grey line) and ambient air temperature (yellow line) as captured by the monitoring system over a four-day window from November 28th to December 2nd. The plot below confirms that the data behaves as expected and provides great insight for the water savings algorithm.

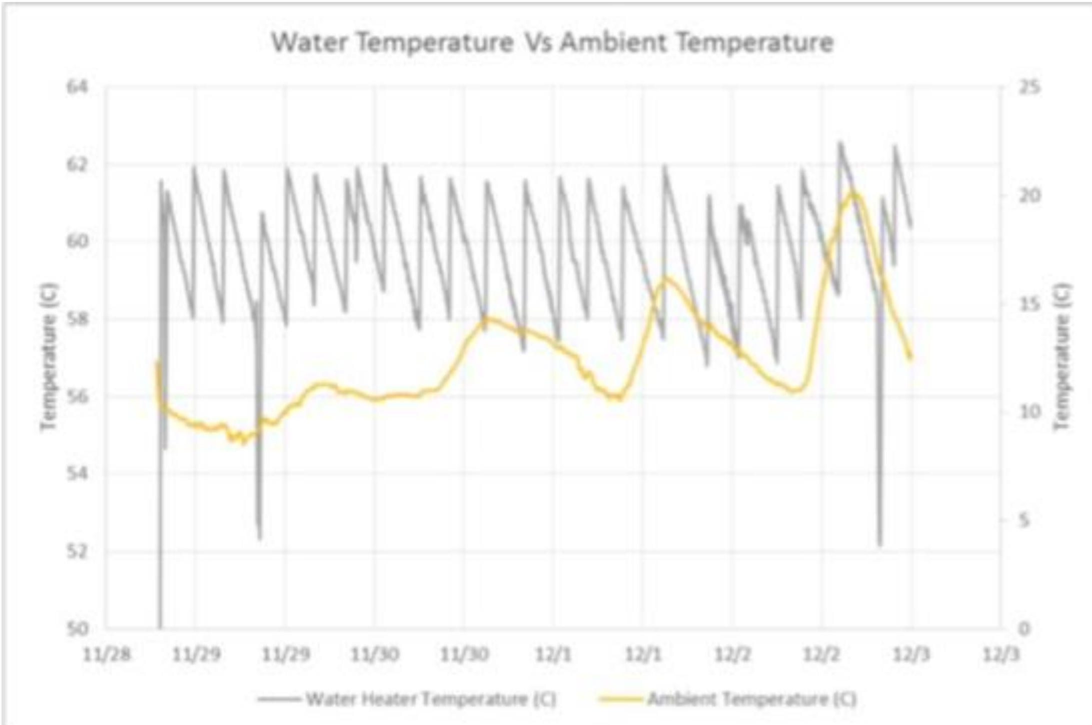

Figure 21. Downstairs water heater temperature relationship

There is no clear relationship between ambient temperature and water heater temperature. Data mining methods like Principal Component Analysis (PCA) will need to be applied to determine the real effects of the temperature outside. From the energy balance heat is lost to environment even through the water heater insulation; however, it is suspected these losses are minor compared to poor control losses. As demonstrated in Andrew's work [19] for HVAC losses, energy savings are far greater when the temperature is maintained tightly around a set point instead of just letting the area to heat up to ambient temperature and then have the air conditioning to work hard to cool down the air again. This is the same methodology for control that is used in the water heater (not PID control), which can be exploited because, in contrast to HVAC, cooling the water inside the tank does not need to be warm all times, but just when it is needed!

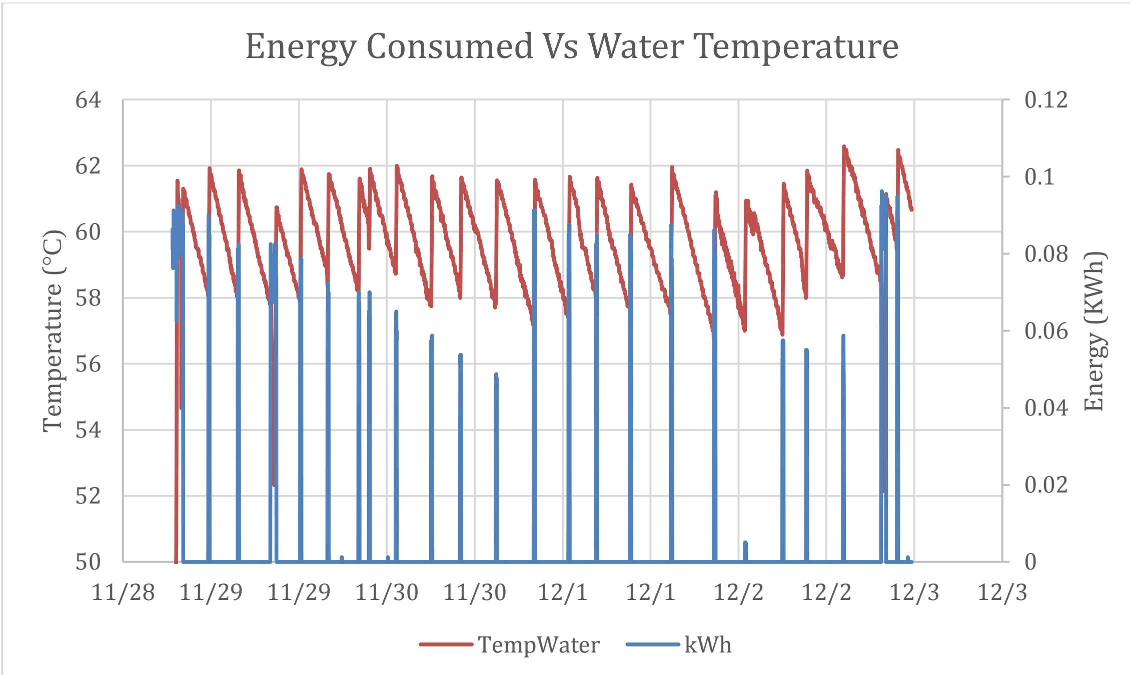

Figure 22. Downstairs water heater energy relationship

Figure 22 shows the relationship between the temperature of the water and the energy consumed. This relationship between energy consumed and water heater temperature is very striking. This shows that the microcontroller is doing its job in collecting the data from the sensors in a coordinated manner. Every time the temperature goes below the cycle set point, the heater comes up and the power meter senses the power drain which exactly correlates with the water heater temperature increase.

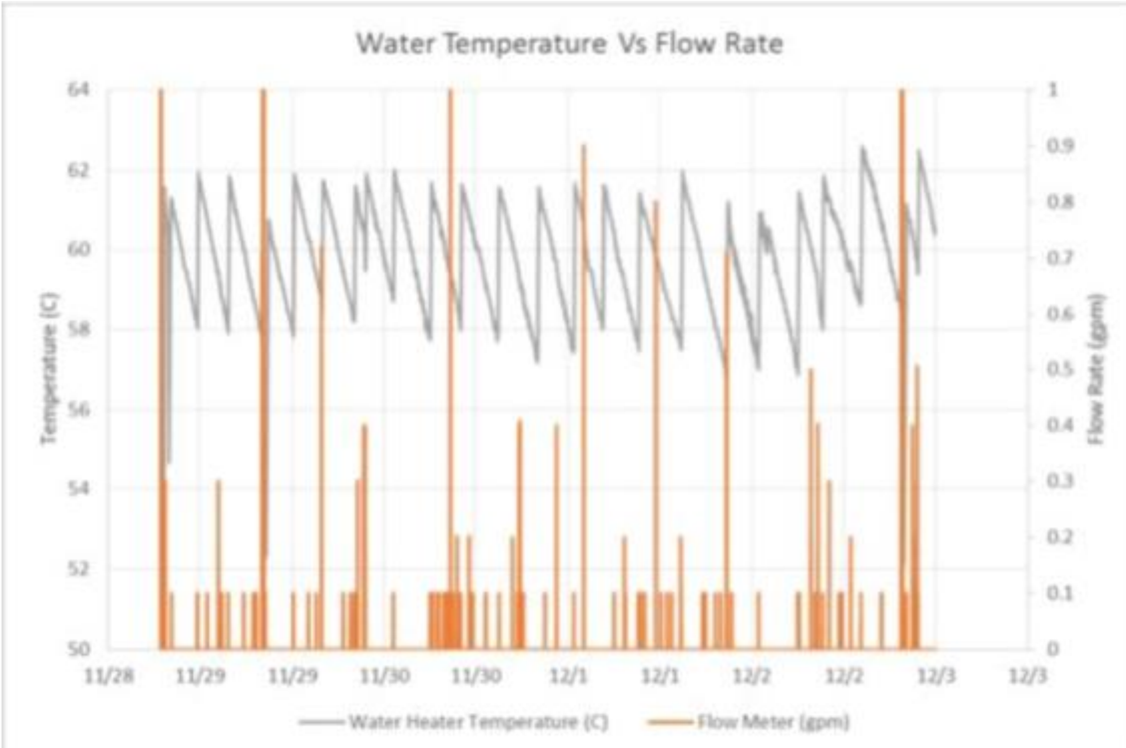

Figure 23. Downstairs water heater flow rate relationship

Figure 23 shows the relationship between the flow rate and the water temperature. Flow rate data is a little bit harder to correlate since this water heater, in particular, is used frequently. In general, it is shown that during high consumptions times the temperature slope is steeper, driving the temperature of the water heater faster to the lower cycle set point. This has to do with the new cool water that comes into the tank, which drives the hot water to a lower temperature.

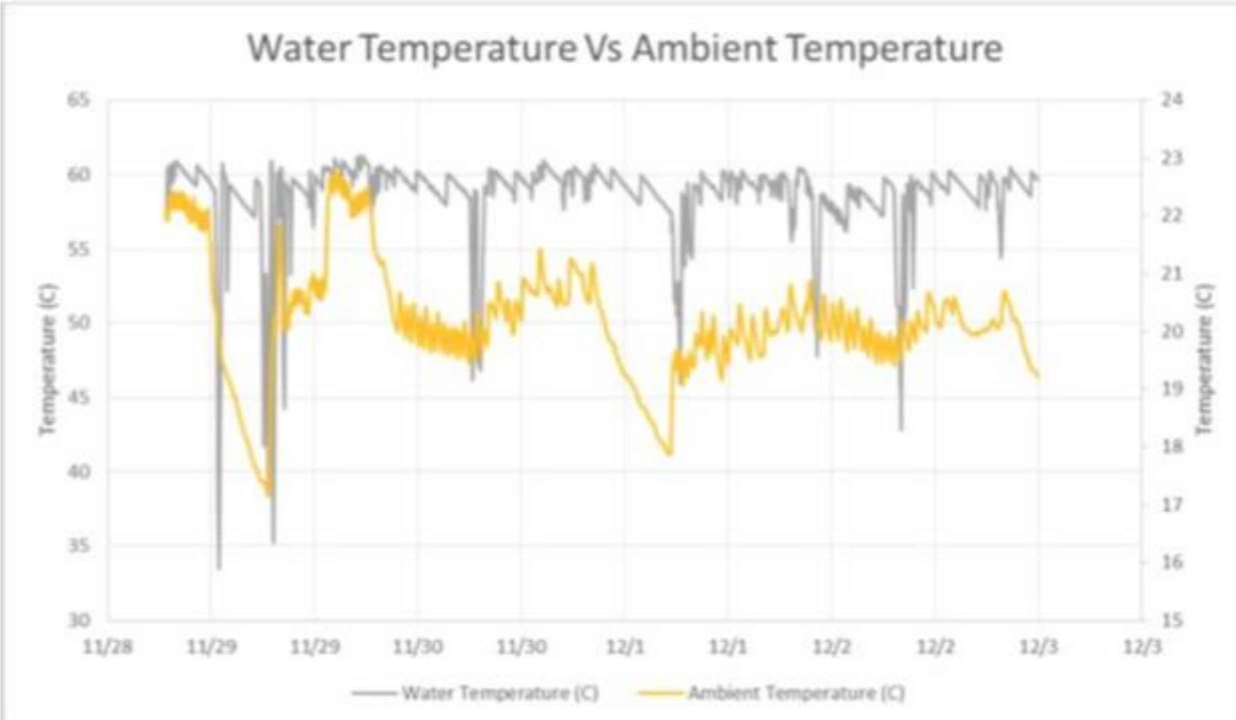

Figure 24. Upstairs water heater temperature relationship

The relationship of the upstairs water heater temperature to the ambient temperature, shown in Figure 24, is different from the downstairs results and shows a strong correlation with the ambient temperature. This completely makes sense taking into account the fact that the upstairs unit is located in the attic, which is not well insulated and therefore the unit the ambient conditions heavily impact it. This shows promising savings for the algorithm implementation because the water does not need to be warm all the time but only when is needed; therefore, when usage behavior patterns are established, the savings will be noticeable since the thermostat will be on only during water usage, especially during winter months. Also, the upstairs water heater is simpler since it only feeds the bathroom, which normally shows a distinguishable usage behavior.

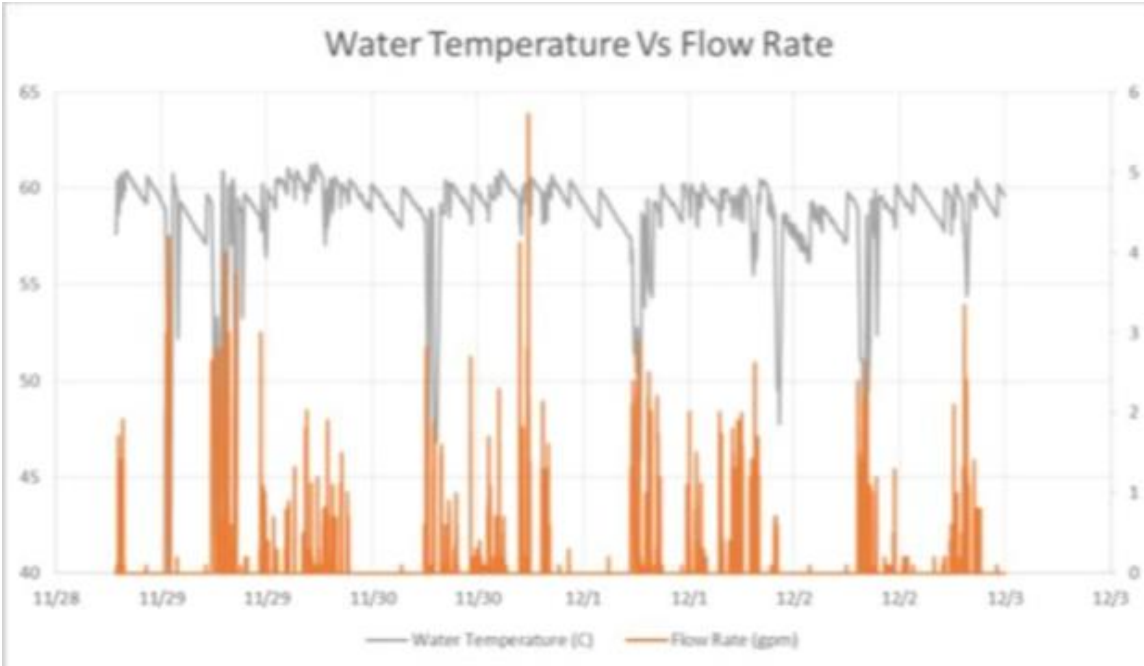

Figure 25. Upstairs water heater flow rate relationship

Similar to the downstairs unit, the temperature is directly affected by the usage rate since new cool water impacts the bulk temperature in the existing water within the unit, this is shown in Figure 25. The high usage intervals can be appreciated in this unit. This has to do with the fact that the upstairs unit is exclusively dedicated to supply the bathroom in the second floor of the house. Here a usage pattern can be appreciated since high consumptions demands are noticed during morning and night times revealing useful ways to save energy using the collected data. As explained earlier, the thermostat will be turned on some time before the usage pattern (morning and afternoon) so it give plenty of time for the water to reach the set point temperature and be ready for usage. The thermostat can be clearly turned off at nights and during the day when the water is not utilized in the house.

Chapter 4: Data Analysis

The main goal of this chapter is to analyze the data collected from the water heater, and create a model that can predict behavior patterns that can be utilized successfully for energy savings. Machine learning algorithms are able to address some of the most challenging problems in engineering, such as computer vision and data mining like this one. Due to the nonlinear and quantitative nature of the data, two different nonlinear models have been chosen, the k-nearest neighbor algorithm (KNN) and the support vector machines (SVM).

4.1 The k-nearest neighbor algorithm (KNN)

The KNN algorithm is a very standard nonlinear model used for classification and regression. It consists of the k closest training examples in the feature space. For regression, we average the values of the k nearest neighbors. In Figure 25, k = 1 is used as an example. In this case the input would be classified under category 1. Another possible approach would be to use k = 3 to get the top 3 matches, and choose the category, which has the majority of the top 3 results.

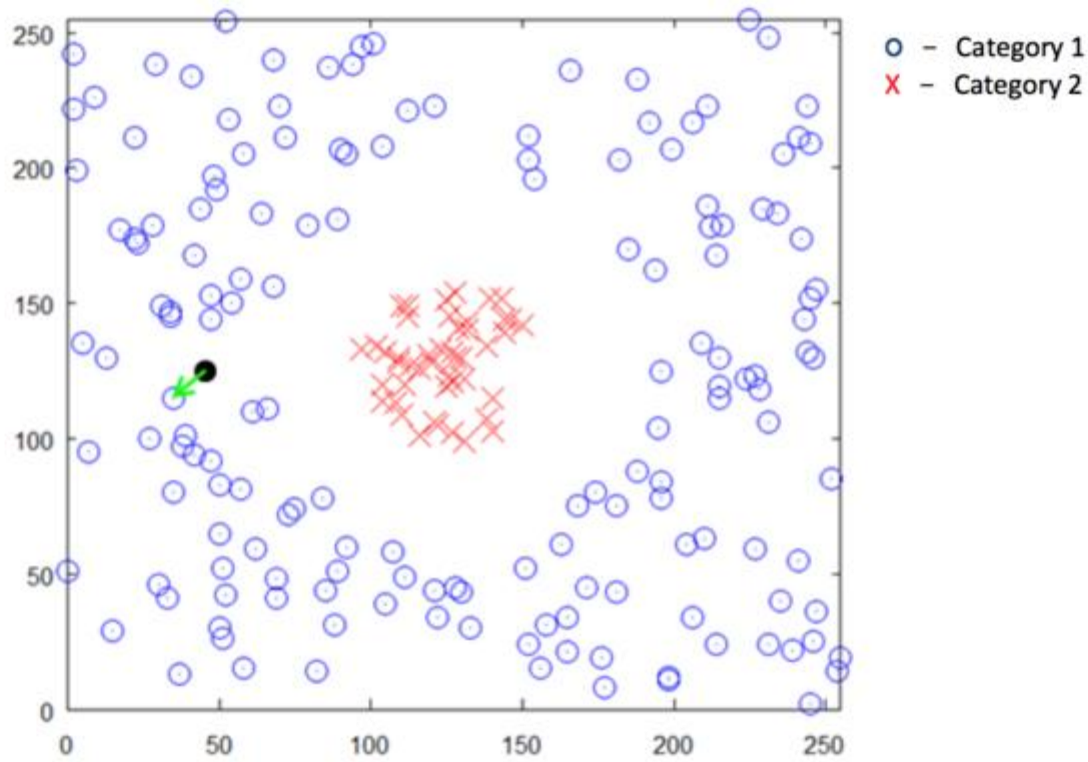

Figure 26. Example of KNN being used for classification

4.2 The Support Vector Machines (SVM)

Firstly, working with neural networks for supervised learning specifically Radial Basis Function (RBF) showed good results. RBF uses feed forward and recurrent networks. Multilayer perceptron (MLP) properties include universal approximation of continuous nonlinear functions and include learning with input-output patterns. RBF simulation proved to be difficult due to finding the numbers of neurons needed for the task making very sensitive to the input parameters. The other issue found is that even if the neural network solutions used tends to converge, this is not always a unique solution [17]. SVM has an advantage over neural network type algorithms because among all possible solutions (hyper planes) there is one that achieves the maximum separations between classes in the case of classifiers. Having the maximum margin (best solution) provides a small error in the location of the boundary, which provides with a least chance of causing misclassification. This makes SVM less sensitive than RBF by avoiding local minima and giving better classification [18]. The goal of SVM is to find the best hyper plane, introduce an alternate loss function for regression, and perform a non-linear mapping to the data into a high dimensional feature space by the kernel approach.

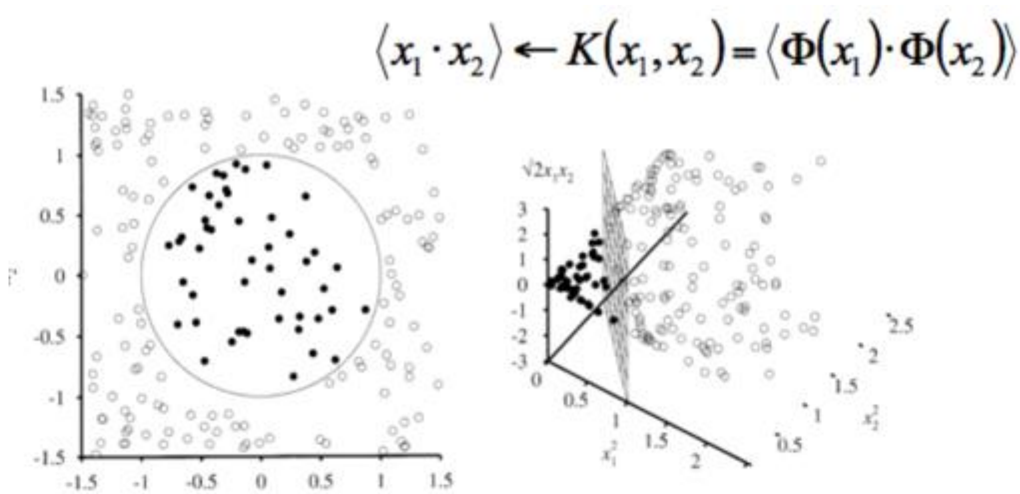

Figure 27. Example of SVM Kernel Transformation

Figure 27 shows data transformation into feature space by the kernel trick in order to form non-linear boundaries.

4.3 The Cross-Validation

In most machine learning algorithms, the goal of the learning algorithm is to build a model, which makes accurate predictions on the training set. Because of this, machine-learning classifiers tend to perform very well on the data they were trained on.

Training set accuracy is not a good indication of how well the classifier will perform, however, when classifying new data outside of the training set is the real indication of the model accuracy. The cross-validation process provides a much more accurate picture of the model's true accuracy. In cross-validation, the data is divided into a large training set and a smaller validation set, then train on the training set and use the validation set to measure our accuracy.

In this approach data is randomly selected, 80% of the existing data to use for training and 20% to use for validation. Since there is some risks in selecting skew data, k-fold cross validation is performed. In this procedure, the data is randomly sorted, and then the data is divided into k folds. Cross validation is run using one of the folds for validation, and the remaining (k-1) folds for training. This is repeated k times, each time selecting a different one of the folds to use for validation. The accuracy over the k folds is averaged to get a final cross-validation accuracy.

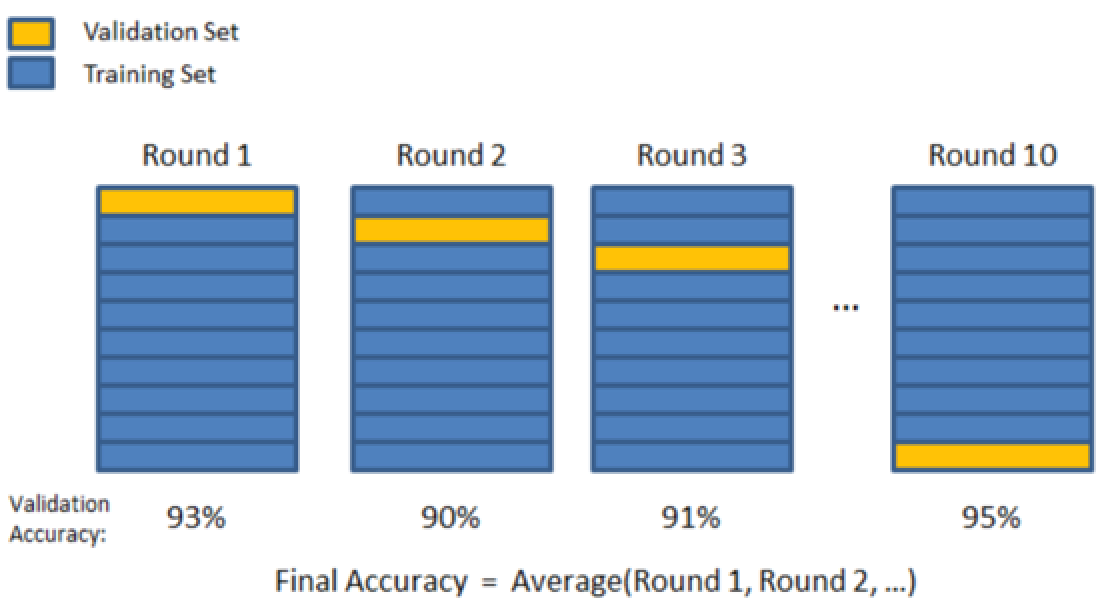

Figure 28. Example for k=10 folds in Cross-validation

4.4 Data Post-Processing

In this section it is described how we calculate our error and measure the results. As mentioned before, a type of error between the actual results and the predictions need to be calculated in order to assess the accuracy of the model.

The Mean Absolute Percentage Error (MAPE) is a measure of accuracy of a method for constructing fitted time series values. Mathematically it is defined in the following equation [ref]:

$$M = \frac{1}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|$$

Where:

$A_t = actual\ value$

$F_t = forecast\ value$

The aggregate error (AE) is a percent difference between the integral of the forecast and integral of the actual.

$$E = \frac{\sum_{t=1}^{n} |F_t| - \sum_{t=1}^{n} |A_t|}{\sum_{t=1}^{n} |A_t|}$$

Where:

$A_t = actual\ value$

$F_t = forecast\ value$

Both the MAPE and the AE errors should be enough metrics to measure the accuracy of the different models presented next.

4.5 Results

Results are presented for both KNN and SVM models for the water heater dataset collected from the sensors. The majority of time was spent with KNN, which had the greatest number of attempts in order to find the best k; therefore, avoiding both over-fitting and under-fitting. All attempts were done only with the water heater data set for both locations upstairs and downstairs. In all cases the data was split in 5 equal parts, 1 fold is used for validation and the other 4 folds were used for training. This same methodology was tried out 5 different times in all possible combinations between the training folds set and the validation folds set (k-fold = 5). The global accuracy is the average of the individual accuracies in each fold combination. In the SVM case, time was spent to find the best kernel for the non-linear transformation. SVM proved to be much faster than KNN due to less data needed to define the forecast prediction after the hyper plane was built. For example, in the upstairs water heater case, the KNN algorithm took over 4 hours to predict the results for a k =6. On the other hand, the SVM algorithm took less than an hour to provide similar results.

- KNN Results

A	Dataset	K	MAPE	AE
1	Down	1	38.6096	-0.08615
2	Down	100	148.23	-0.01506
3	Down	20	61.1501	-0.03793
4	Down	15	23.8114	-0.0372
5	Down	7	18.94916	-0.02611
6	Down	5	7.6086	-0.02622
7	Down	3	4.9674	-0.01387

Table 3. KNN Results for Downstairs Water Heater

Key for the table: A=Attempt, Data=Dataset {Up=Upstairs water heater, DOWN=Downstairs water heater}, K=KNN k-Value, MAPE= Mean Absolute Percentage Error, AE=Aggregate Error.

A	Dataset	K	MAPE	AE
1	Up	1	1.1394	-0.02005
2	Up	40	0.4194	-0.01353
3	Up	20	0.4033	-0.05351
4	Up	15	0.3256	-0.0035
5	Up	4	0.5222	-0.00598
6	Up	9	0.2179	-0.0414
7	Up	6	0.1248	-0.00038

Table 4. KNN Results for Upstairs Water Heater

Key for the table: A=Attempt, Data=Dataset {Up=Upstairs water heater, DOWN=Downstairs water heater}, K=KNN k-Value, MAPE= Mean Absolute Percentage Error, AE=Aggregate Error.

Tables 3 and 4 show the results for the KNN algorithm for both water heaters. It is clear that the aggregate error is very low and the MAPE values are low especially in the upstairs water heater. The aggregate error is much lower in the upstairs water heater compared to the one downstairs. This has to do with the smaller variation in water usage that occurs in the upstairs water heater. The usage pattern in the upstairs water heater is very consistent with the exception of specific events such as family or friends staying over. On the other hand, the downstairs usage pattern is very erratic during the day and only consistent at nights. Also, it is important to mention that the upstairs water heater is more affected by outside weather conditions than the downstairs one due to its location in the attic. Further correlations with ambient temperature should be found with the power consumption of the upstairs water heater. The next step in the future work is to study this correlation further for finding better predictions. The large MAPE values and variation in the downstairs heater suggests that the models are as consistent as the model found for the upstairs water heater. In the downstairs water heater the best k value found for the dataset was 3, which is similar to the value found in the upstairs setup of 6.

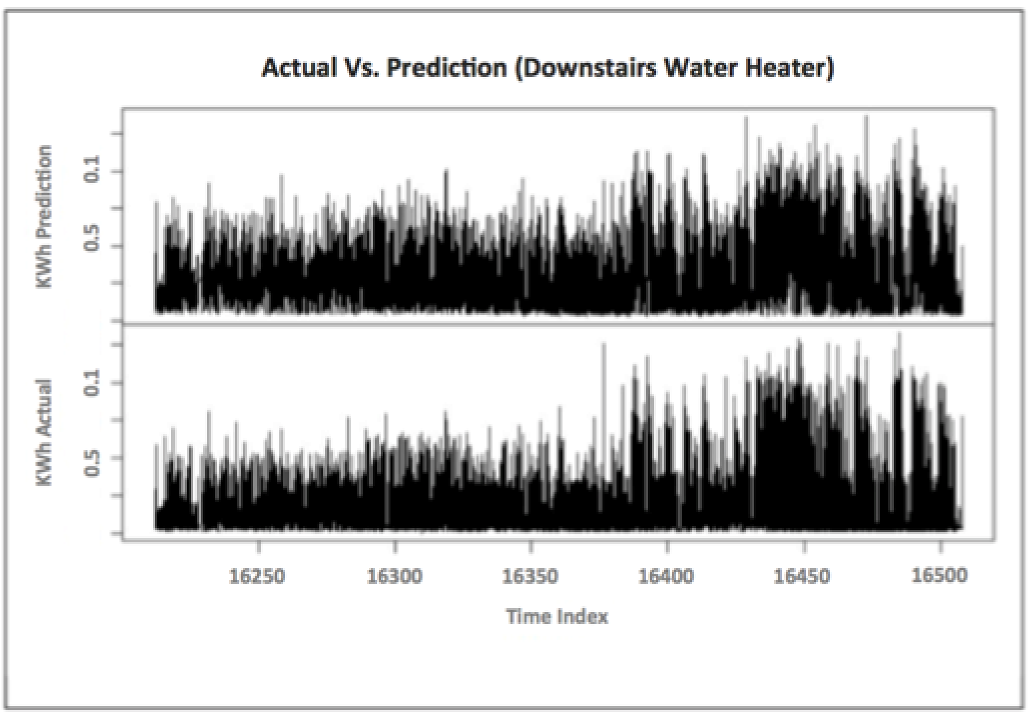

Figure 29. Actual Vs. Prediction Model Downstairs Water Heater

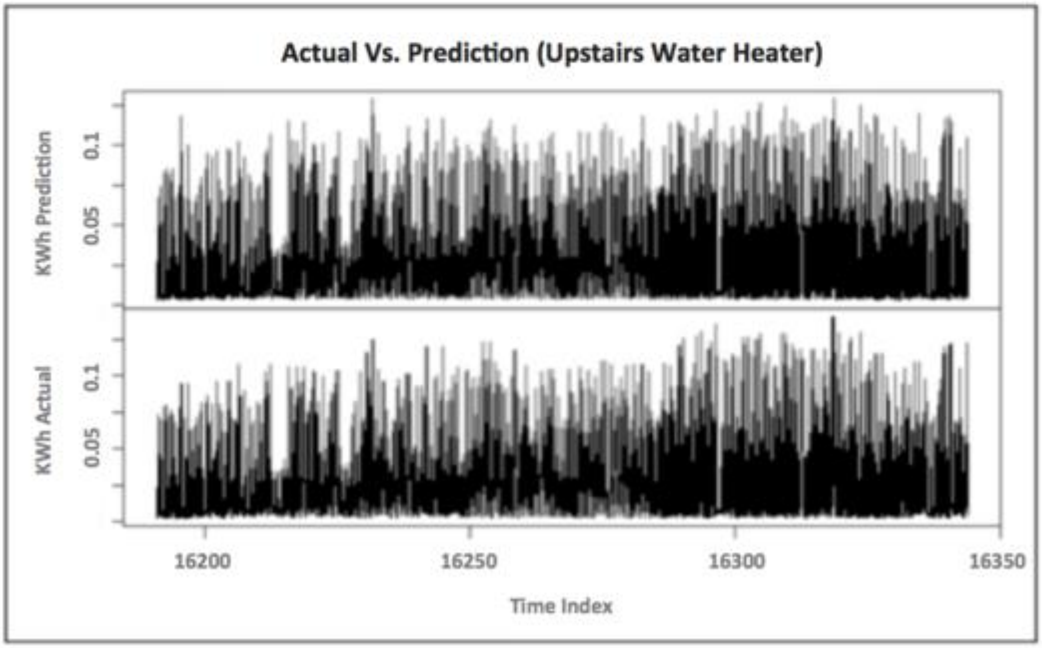

Figure 30. Actual Vs. Prediction Model Upstairs Water Heater

The actual and predicted energy usage values for the downstairs and upstairs water heaters are shown in Figures 29 and 30, respectively. The prediction in Figure 30 looks very similar to the actual energy value and it is believed that adding weather correlation to this model will further increase its accuracy [19]. Including weather data to the water heater data set will be a more complicated problem since it will need further preprocessing techniques such as wavelength decomposition [19] and feature extraction in order to correlate both datasets properly. In the other hand, Figure 29 does not show as much similarity between the actual and predicted model, but it is still a very good approximation.

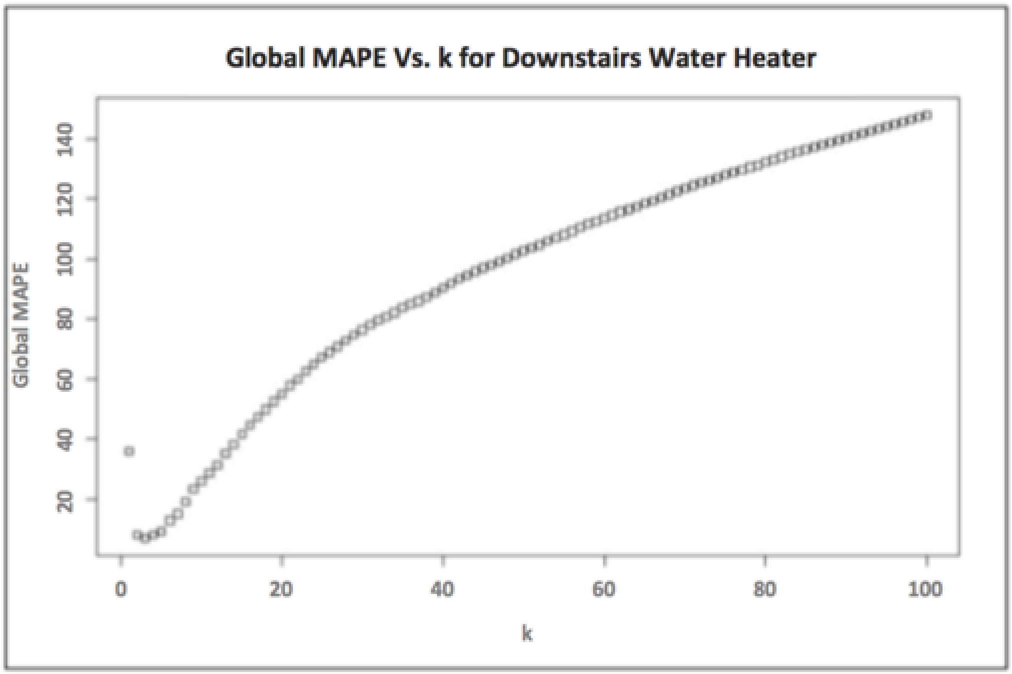

Figure 31. Global MAPE Vs. k for Downstairs Water Heater

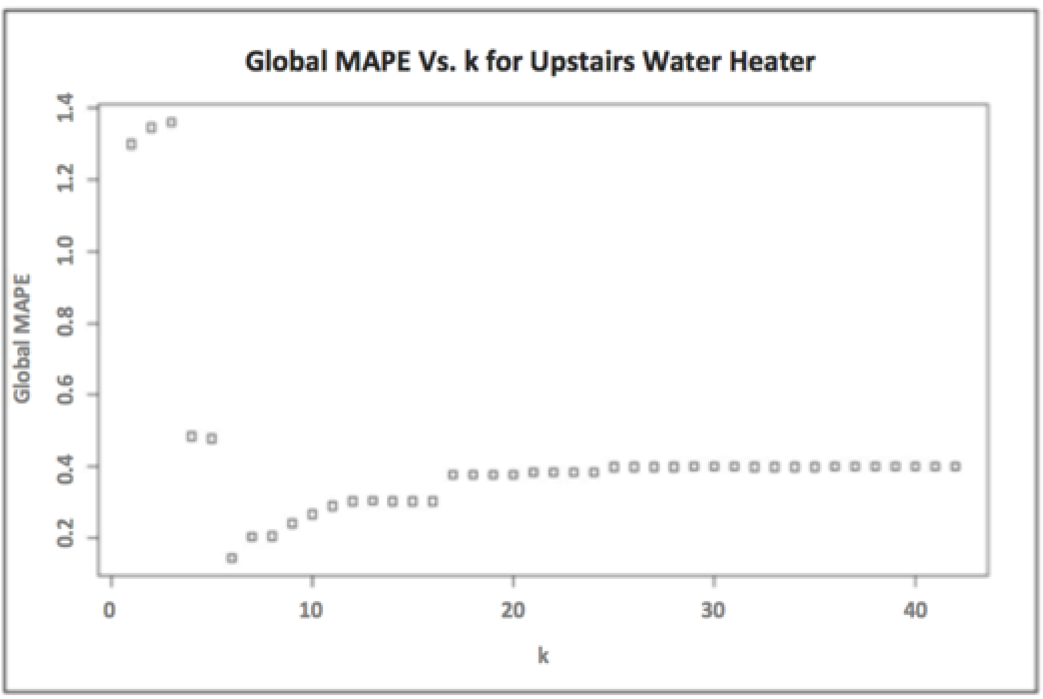

Figure 32. Global MAPE Vs. k for Upstairs Water Heater

Figure 31 and Figure 32 were created with the algorithms to loop and find the best k for a given training set. These results show the analysis previously described in Tables 3 and 4. The variance in values for best k compared with MAPE is interesting. Based on early results from upstairs water heater one might expect a high k to result in over-fitting, but that is not the case for upstairs water heater MAPE.

● SVM Results

SVM can be applied to regression problems by the introduction of an alternative loss function [20]. The loss function must be modified to include a distance measure. The regression can be linear and non-linear. Linear models mainly consist of the e-intensive loss functions, quadratic and Huber loss function. Similarly to classification problems, a non-linear model is usually required to adequately model data. When the data is linear, a separating hyper plane could be use to separate the data, however; the water heater energy usage (KWh) is non-linear, as a result, the datasets are inseparable. A non-linear mapping can be used to map the data into a high dimensional feature space where linear regression is performed. The kernel approach is employed to address the curse of dimensionality. Two different kernels were utilized in order to enable operations to be performed in the input space rather than the high dimensional feature space due to the non-linearity of the dataset. Exponential radial basis function (RBF) kernel provides good results when discontinuities are acceptable [20], on the other hand; multi-layer perceptron (MLP) kernel shows good results when the datasets are non-linear and have wavelength behavior [21]. Figures 33 and 34 show the actual water heater energy usage versus the SVM predicted behavior of the downstairs water heater for the RBF and MLP kernels, respectively.

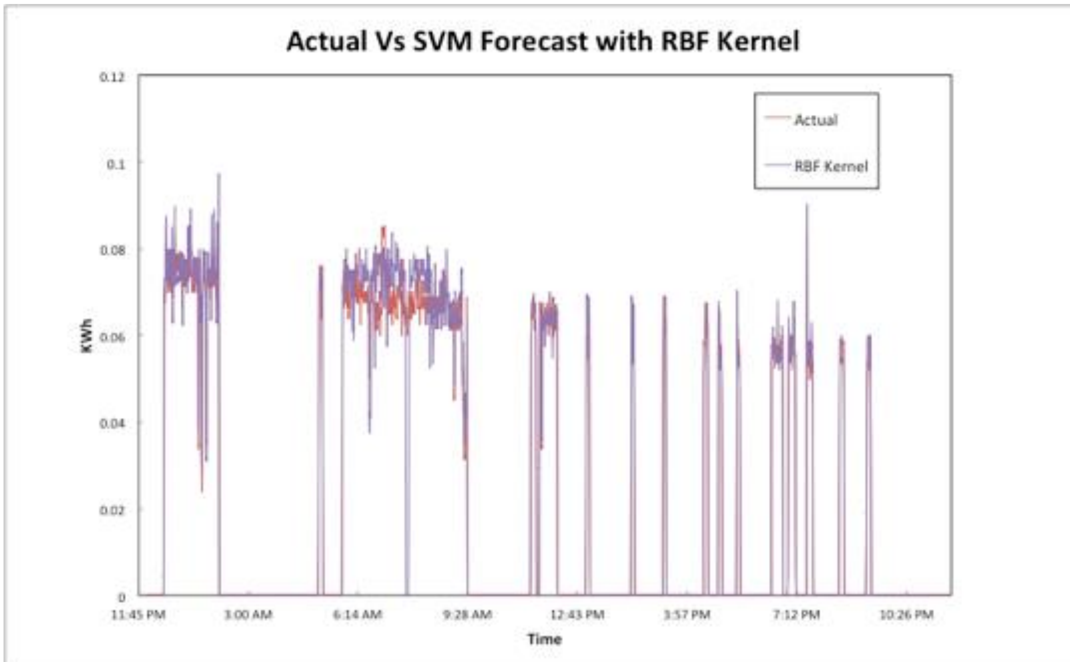

Figure 33. Actual Vs. SVM Forecast with RBF Kernel for Downstairs Water Heater

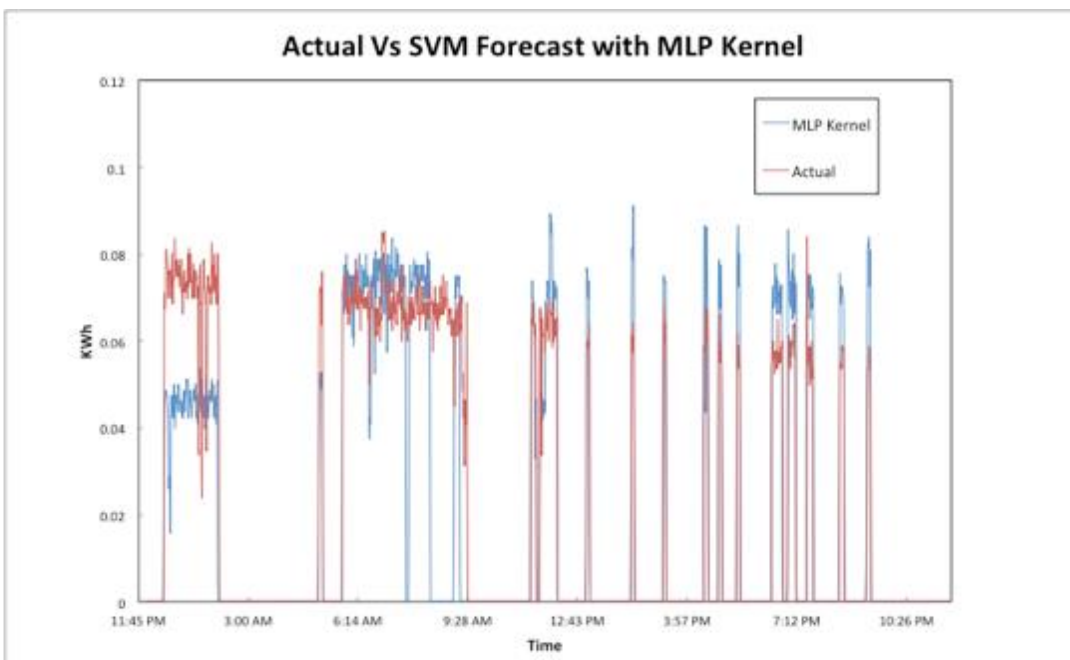

Figure 34. Actual Vs. SVM Forecast with MLP Kernel for Downstairs Water Heater

The SVM algorithm proved to be less expensive in terms of computer time and higher accuracy than KNN algorithm. Even when the numbers of neighbors (k=3) required for the algorithm was reasonable lower, the KNN models still utilize all the points from the training dataset in comparison to SVM which only utilize the points around the hyper plane once the dataset have been transformed into a different space and it has been divided. Furthermore, SVM was shown to be very stable and not as sensitive compared to other neural network models such as KNN [19]. The RBF kernel was shown to be ~45% more accurate than the MLP kernel. One single hyperbolic transformation layer was chosen for the MLP case, so adding more layers with a Fourier or B-splines transformation could make the model more accurate; however, the main challenge of SVM is the kernel selection and having to increase the difficulty of this by adding more layers would not be make sense since RBF kernel is highly accurate and much simpler in comparison to the MLP kernel case. Additionally, Figures 33 and 34 show the water usage behavior of the downstairs water heater is much more inconsistent in comparison to that of the upstairs water heater, shown in Figures 35 and 36, which is utilized more consistently in the morning and night hours.

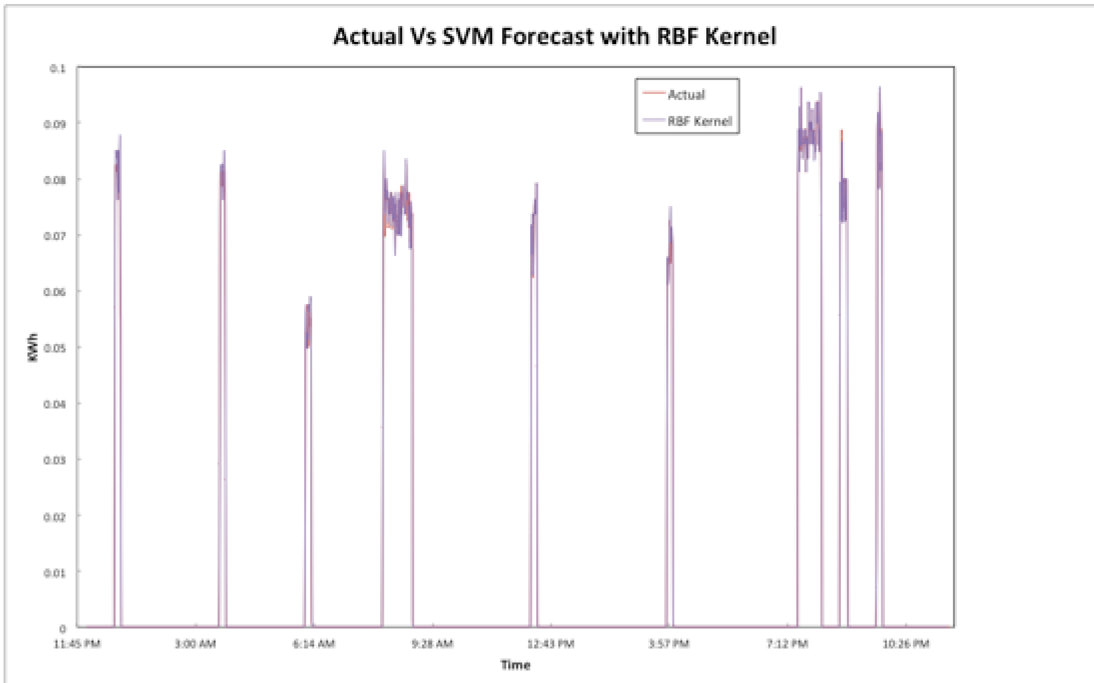

Figure 35. Actual Vs. SVM Forecast with RBF Kernel for Upstairs Water Heater

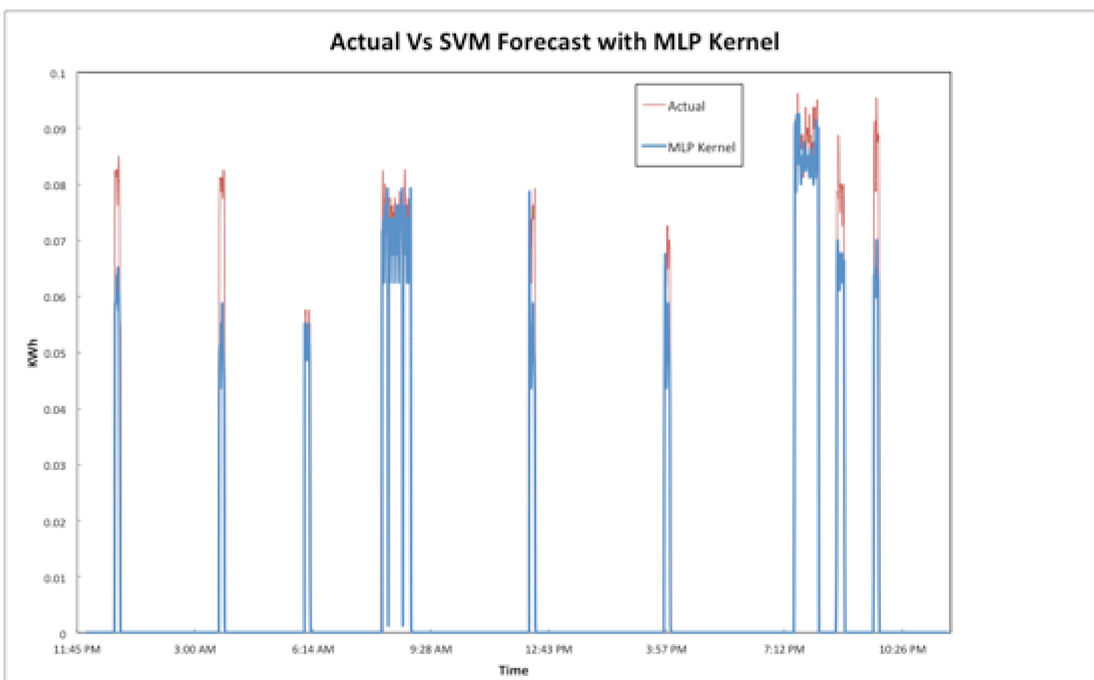

Figure 36. Actual Vs. SVM Forecast with MLP Kernel for Upstairs Water Heater

As for the downstairs water heater the RBF kernel is more accurate than the MLP one. Finding the best kernel is one of the complications in the SVM algorithm and in this case the usage of RBF and MLP kernels were based previous experience [20]. The SVM model seems to suit better the upstairs water heater dataset in comparison to the downstairs data, this is also believed to be associated with the usage behavior between both water heaters and their respective locations. The same observation was found for the KNN model.

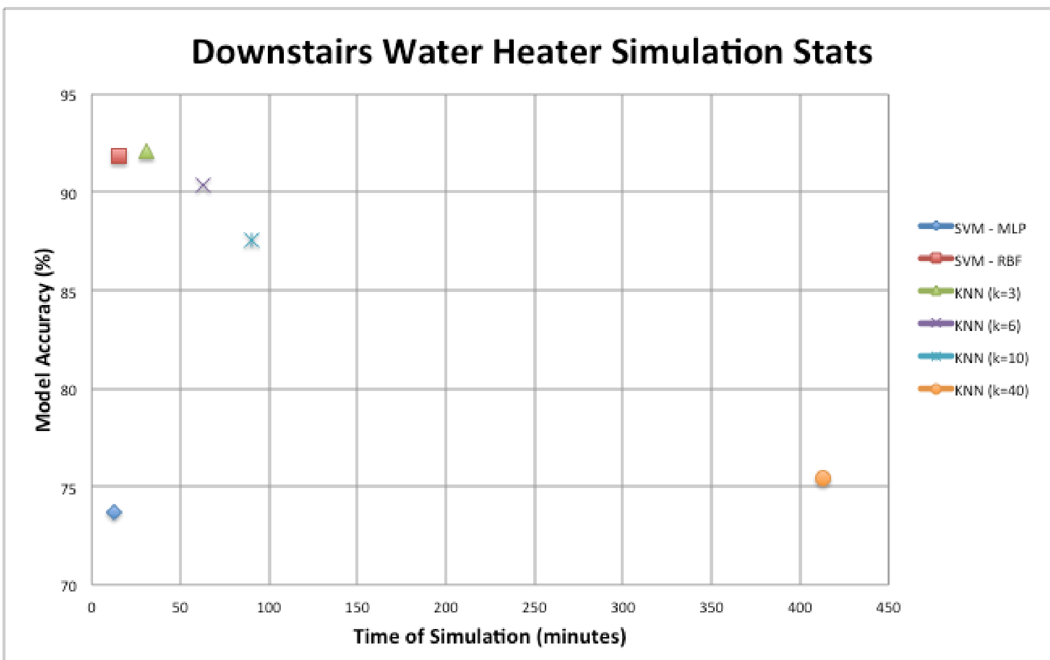

Figure 37. Downstairs Water Heater Simulation Stats

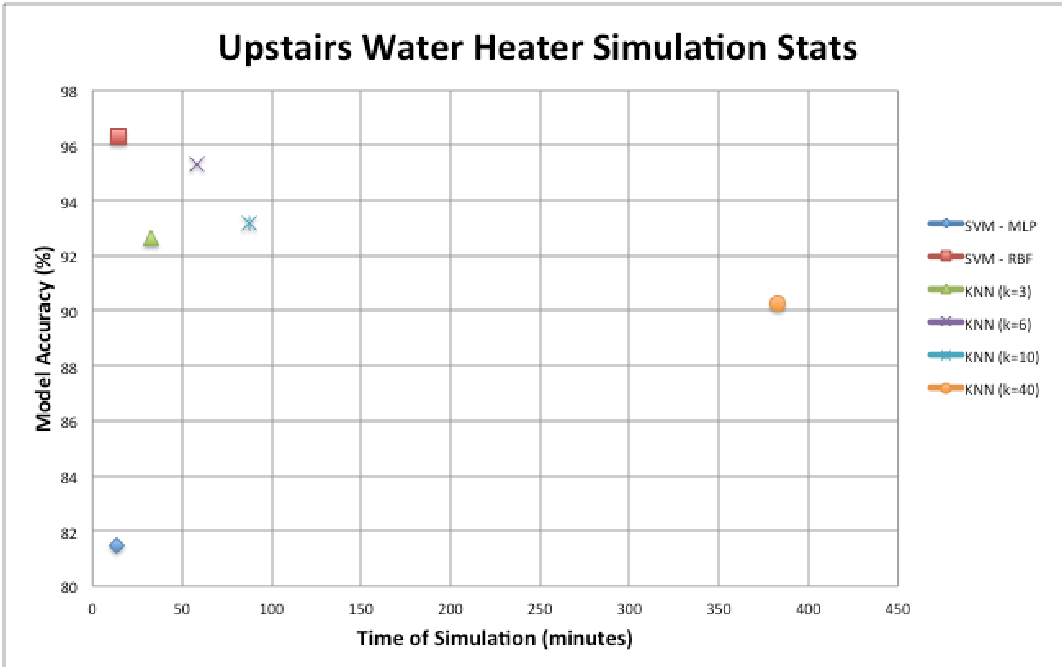

Figure 38. Upstairs Water Heater Simulation Stats

It is important to analyze the accuracy and simulation times for the different algorithms and specific conditions. Figures 37 and 38 plot the accuracy of the KNN and SVM models against the simulation time in minutes, and demonstrate two important points discussed above. First of all, the SVM algorithm is at least ~50% faster than the KNN algorithm in general. When k=3, the KNN algorithm runs the fastest with high accuracy but it is still slower than the SVM case. When k=40, the KNN algorithm is ~35 times slower than the SVM algorithm and provides less accuracy. Second of all, the SVM algorithm for the RBF kernel is still more accurate than the best case in the KNN algorithm. This difference is 0.3% and 1% for the downstairs and upstairs cases, respectively. Lastly, the results show that for both cases (downstairs and upstairs) the worst KNN simulation still performs better than the SVM algorithm for the MLP kernel.

Chapter 5: Related Work

Understanding water heater power consumption is not a novel idea; however, designing a circuit to collect data from the water heater itself and successfully collect data to create some models for energy prediction is something worth trying. There are several energy prediction approaches for the residence as a whole, but none of these focus exclusively on water heater power consumption. When the electric bill for a residence is studied, around 60% to 80% of the bill comes from the HVAC power consumption. Because of the fact that a water heater is so cheap (~$2,000), and it last so long (10 to 20 years), it difficult to find related work. On the software side, with the mobile revolution and IoT technologies, there is some work related to obtaining information of the power consumption behavior and its relationship to the air conditioning thermostat [15]. For the HVAC case, it is justifiable to have an advanced thermostat that can communicate via WI-FI or Bluetooth to other devices due to the high percentage of electricity usage and the capital cost of the air conditioning system as a whole; however, the water heater thermostat is very rudimentary and improvements are not foreseen in the future. All the previous reason motivated this study, and focus the attention on the sensor selection. It is known that a company exists (Aquanta) [14] that provides hardware for measuring water heater related metrics. This hardware was surprisingly cheap, so technical information for the sensors as the datasheet was requested from the company to understand the price difference. The company didn't provide any datasheets for the sensors, so the circuit design followed the data reliability path as explained in Chapter 2. Most of the work that was related to this Thesis came from HVAC power consumption savings solutions [2, 6] since the problem definition and challenges are very similar in terms of model prediction and data preprocessing. The estimated cost for the microcontroller integration for this study is ~ $1,500, it is believed that with better circuit integration and more testing time for different types of sensors the price could be reduced to ~$300. Aliabad offers hundreds of different sensors but no datasheet is provided in order to verify its integration and to obtain the proper circuit design.

Chapter 6: Conclusion

After collecting more data the intention was to advance the model development further and from there devise a control algorithm. The algorithm envisioned for future work will combine the descriptive physical model with machine learning and combinatorial optimization. At this time, a predictive model was developed to forecast usage based on data collected from the microcontroller circuit integration.

The circuit design successfully collected data from 2 different water heaters (upstairs and downstairs) and sent it to the server. The sensor selection was appropriate since the microcontroller processor correctly acquired the data. The WI-FI set up and firmware creation also proved to be a good choice since the server successfully collected the data. Furthermore, the communication between the mobile app and the microcontroller was demonstrated to be functional when the mobile temperature application feature varied with the changing water heater conditions.

Experimental results were very positive. Successful creation of two predictive models (KNN and SVM) was achieved and accurate forecast of energy load based on water heater data was demonstrated. Results varied by location, but for both water heater locations, an average error rate less than 18% and an average aggregate error of 1.1% was achieved for KNN model.

SVM was a more challenging model in the beginning due to the mathematical background needed to understand its application. Applying the Support Vector approach to a particular practical problem involved resolving a number of questions based on the problem definition and the design involved with it. One of the major challenges was that of choosing an appropriate kernel for the given application [4]. There are standard choices such as a Gaussian or polynomial kernels that are the default options, but for non-linear datasets, more elaborate kernels were needed. Therefore, RBF and MLP kernels were used to define the feature space in order to provide the description needed by the machine for viewing the data. SVM demonstrated to be a powerful algorithm to approximate the training data and generalizes better to the water heater dataset. The results showed that the complexity of the kernel affects the performance on the new datasets; however, SVM supports parameters for controlling the complexity which were validated through Cross-Validation on the given datasets.

Significant time was spent understanding the circuit design problem and especially in the sensor selection. It was critical to select the right sensor that could easily capture the system behavior and communicate with the microcontroller in an efficient manner. There are many factors that contribute to water heater energy usage, but the primary factors are usage behavior and weather conditions. Energy usage contains both linear and nonlinear components, and a lot of randomness with human behavior and weather influences. Related to this, one of the issues faced was combining the data in ways that were meaningful. Since data collection began in November, the only data known to date, is affected by winter weather. Texas winter is relatively mild so it would be interesting to see the water heater behavior during Summer time, especially since Texas summers can be extreme.

Potential savings are mainly due to the usage pattern behavior. If the downstairs water heater, which behavior was hard to predict, is turned off at night during winter time, the potential savings is between \$300 and \$500 annually. In the case of the upstairs water heater, the potential savings is greater since its usage predictability is more accurate and it could potentially lead to \$400 to \$800 savings annually.

For future work, the algorithm needs to evolve with weather and electric bill data in order to better predict the energy consumption during the whole year in any location of the house. It is envisioned that the algorithm will combine the descriptive physical model with machine learning and combinatorial optimization. The model will to forecast usage as well as create a decision space that can be searched to find the cheapest control path from the mobile application. Lastly, the sensor design can be optimized so the price of the whole package, including the algorithm, can be easily installed in any electric residential water heater and provide some savings despite its initial cost.

Lastly, the experience obtained in Andrew's work on thermostat data preprocessing and weather correlation provided an insight and better guidance for this work. The results obtained in neural networks provided a foundation for comparison with the SVM results and a better understanding on kernel selection.

Appendix

Heat Transfer Balance

After analyzing the initial results of the data an initial model was developed to describe the physical behavior of the water heater and compare it with the data we are seeing. This is a very rough initial model that is meant to evolve with future work.

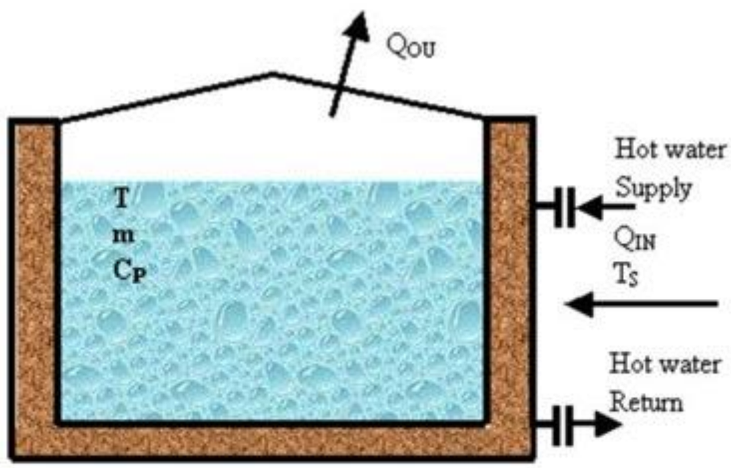

Figure 39. Water Heater Energy Balance

Water heater tanks are devices for storing thermal energy in water. Typical water heater applications are for domestic hot water heating. The input object water heater mixed provides a model that simulates a well-mixed water tank for simulating residential water heaters.

● Energy balance

The well-mixed assumption implies that all water in the tank is at the same temperature even when cold water enters at the bottom of the tank when hot water is being used. Since temperature varies in time as when the water is being heated or the water is losing energy through insulation, this model is considered unsteady state. Boundary conditions in the 4 walls of the tank are considered. To calculate the water temperature, the model analytically solves the differential equation governing the energy balance of the water tank:

Energy balance for control volume is:

1

$$cp\frac{\partial\mu}{\partial t} + \nabla.\emptyset - Q = 0$$

The Fourier's Law of Heat Conduction is:

$$\emptyset = -K\nabla_{\mu}$$

Where K is called the thermal diffusivity. Applying Law of Heat into Energy Balance we have:

$$\frac{\partial\mu}{\partial t} = k\nabla^2\mu + \frac{Q}{cp}$$

Conduction is neglected in any direction and for only one dimension the equation is reduced to:

$$\rho V c_p \frac{dT}{dt} = q_{net}$$

Where:

$\rho = density\ of\ the\ water$

$V = volume\ of\ the\ tank$

$c_p = specific\ heat\ of\ the\ water\ at\ constant\ pressure$

$T = temperature\ of\ the\ water$

$t = time$

$q_{net} = net\ heat\ transfer\ rate\ to\ the\ tank\ water$

The density and volume can be replaced with the total mass m of water in the tank to get:

$$mc_p \frac{dT}{dt} = q_{net}$$

The net heat transfer rate q_{net} is the sum of gains and losses due to multiple heat transfer pathways.

$$q_{net} = q_{heater} + q_{oncycpara} + q_{offcycpara} + q_{oncycloss} + q_{offcycloss} + q_{use} + q_{source}$$

Where:

$q_{heater} = heat\ added\ by\ the\ heating\ element$

$q_{oncycpara} = heat\ added\ due\ to\ on\ cycle\ parasitic\ loads\ (zero\ when\ off)$

$q_{offcycpara} = heat\ added\ due\ to\ off\ cycle\ parasitic\ loads\ (zero\ when\ on)$

$q_{oncycloss} = heat\ transfer\ due\ to\ the\ ambient\ enviroment\ (zero\ when\ off)$

$q_{offcycloss} = heat\ transfer\ from\ the\ ambient\ enviroment\ (zero\ when\ on)$

$q_{use} = heat\ transfer\ to\ the\ house$

$q_{source} = heat\ transfer\ from\ the\ water\ source$

For the sake of this first control model attempt the heat added by parasitic loads can be neglected; therefore, $q_{oncycpara}$ and $q_{offcycpara}$ are zero.

Also $q_{oncycloss}$ and $q_{offcycloss}$ can be approximated to q_{amb}, which are the heat losses through the tank due to the environment and insulation

Heat capacity of the tank can be expressed as:

$$mc_p = C$$

Rearranging the equation we have:

$$C\frac{dT}{dt} = q_{heater} + q_{source} + q_{use} + q_{amb}$$

$$q_{source} = \dot{m}c_p\Delta T$$

Where:

$\dot{m} = source\ flow\ rate$

$\Delta T = T_{source} - T$

$T_{source} \cong T_{amb}$

$$q_{use} = \dot{m}c_p\Delta T$$

Where:

$\dot{m} = use\ flow\ rate \cong source\ flow\ rate$

$\Delta T = T_{use} - T$

$T_{use} \cong control\ output\ for\ the\ algorithm$

$$q_{amb} = \Delta T / R$$

Where:

$\Delta T = T_{amb} - T$

$R = thermal\ resistance\ of\ insulation$

Substituting we have:

$$\frac{dT}{dt} = \frac{1}{C}\left(q_{heater} + \dot{m}c_p T_{source} - \dot{m}c_p T + q_{heater} + \dot{m}c_p T_{source} - \dot{m}c_p T + \frac{T_{amb}}{R} - \frac{T}{R}\right)$$

$$\frac{dT}{dt} = \left[\frac{1}{C}\left(q_{heater} + \dot{m}c_p T_{source} + q_{heater} + \dot{m}c_p T_{source} + \frac{T_{amb}}{R}\right)\right] + \left[-\frac{1}{C}\left(2\dot{m}c_p + \frac{1}{R}\right)\right]T$$

$$\frac{dT}{dt} = a + bT$$

Where:

$$a = \frac{1}{C}\left(q_{heater} + \dot{m}c_p T_{source} + q_{heater} + \dot{m}c_p T_{source} + \frac{T_{amb}}{R}\right)$$

$$b = -\frac{1}{C}\left(2\dot{m}c_p + \frac{1}{R}\right)$$

Solution for linear separable differential equation is:

$$T(t) = \left(\frac{a}{b} + T_i\right)e^{bt} - \frac{a}{b}$$

Where:

$T(t) = temperature\ of\ the\ tank\ at\ time\ t$

$T_i = temperature\ of\ the\ tank\ at\ t = 0$

The control algorithm to control set point in the water heater microcontroller is governed by the following differential equation solution for temperature and time:

$$\boldsymbol{t = \frac{1}{b} ln \left[\frac{\frac{a}{b} + T_f}{\frac{a}{b} + T_i} \right]}$$

References

[1] How water heater thermostats works
http://waterheatertimer.org/How-water-heater-thermostat-works.html

[2] Aggarwal, Sanjeev Kumar, Lalit Mohan Saini, and Ashwani Kumar. "Electricity price forecasting in deregulated markets: A review and evaluation. "International Journal of Electrical Power & Energy Systems 31.1 (2009): 13-22.

[3] *Thermocouple-to-Digital converter*
. https://www.maximintegrated.com/en/products/analog/sensors-and-sensor-interface/MAX31855.html
.

[4] Kraskov, Alexander, Harald Stögbauer, and Peter Grassberger. "Estimating mutual information." Physical review E 69.6 (2004): 066138.

[5] Sorjamaa, Antti, Jin Hao, and Amaury Lendasse. "Mutual information and k-nearest neighbors approximator for time series prediction." Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005. Springer Berlin Heidelberg, 2005. 553-558.

[6] Pandey, Ajay Shekhar, Devender Singh, and Sunil Kumar Sinha. "Intelligent hybrid wavelet models for short-term load forecasting." Power Systems, IEEE Transactions on 25.3 (2010): 1266-1273.

[7] Cao, Shuanghua, and Jiacong Cao. "Forecast of solar irradiance using recurrent neural networks combined with wavelet analysis." Applied Thermal Engineering 25.2 (2005): 161-172.

[8] Park, Dong-Chul, Dong-Min Woo, and Seung-Soo Han. "Electric Load Forecasting Using Adaptive Multiresolution-Based Bilinear Recurrent Neural Network." Image and Signal Processing, 2008. CISP'08. Congress on. Vol. 4. IEEE, 2008.

[9] Lukoše vičius, Mantas. "A practical guide to applying echo state networks. "Neural Networks: Tricks of the Trade. Springer Berlin Heidelberg, 2012. 659-686.

[10] ViSiDiA: Visualization and Simulation of Distributed Algorithms
http://visidia.labri.fr//html/home.html

[11] DAJ: Distributed Algorithms in Java
https://www.risc.jku.at/software/daj/

[12] A Simulator for Self-Stabilizing Distributed Algorithms
https://www.cs.bgu.ac.il/~projects/projects/odedha/html/

[13] The GridSim Toolkit
https://www.cct.lsu.edu/~dsk/eScience2007Posters/sulistio.html

[14] The Aquanta smart water heater
https://aquanta.io/

[15] Dong Chen, David Irwin, Prashant Shenoy, and Jeannie Albrech. "Combined Heat and Privacy: Preventing Occupancy Detection from Smart Meters" IEEE International Conference on Pervasive Computing and Communications (PerCom), 2014.

[16] Applications of Data Mining Techniques to
Electric Load Profiling
https://www2.ee.washington.edu/research/real/Library/Thesis/Barnaby_PITT.pdf

[17] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, "Introduction to Data Mining" Addison-Wesley, First Edition, 2005.

[18] Daniela Witten, Gareth James, Robert Tibshirani, and Trevor Hastie, " An Introduction to Statistical Learning with Applications in R" Springer, 6th printing 2016 Edition.

[19] Andrew Davis, Ross Baldick, " EnergyBill - Energy Consumption Predictive Modeling" The University of Texas at Austin, 2017.

[20] Osuna E., Freund R., and Girosi F., “Support Vector Machines: Training and Applications”, A.I. Memo No. 1602, Artificial Intelligence Laboratory, MIT, 1997.

[21] Veropoulos K., Cristianini N., and Campbell C., "The Application of Support Vector Machines to Medical Decision Support: A Case Study", ACAI99

[22] Data base collection cloud http://water.energybill.com/api/