Scalable Smart Transducers Network Using Power-Over-Ethernet

Towards smarter, safer and more controllable buildings

by

Ivan Lobachev

B.A.Sc., The University of British Columbia, 2015

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Applied Science

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES
(Electrical and Computer Engineering)

The University of British Columbia
(Vancouver)

December 2016

Abstract

In the age of rapid technological advancement, many telecommunication applications are being integrated into our lives, including smart phones and Internet of Things (IoT). Smart buildings (and houses) use these technologies to reduce energy consumption and increase safety. The need for these buildings is growing as urbanization continues and resources dwindle. According to a report by the United Nations, by 2050 66% of the worlds population will live in cities. This will cause the size and number of megacities to expand drastically in the near future. Complex communication networks, controls and other services will allow us to build smart cities to manage and improve the publics quality of life. The necessity of smart buildings and, eventually, cities, have stimulated growth of sensor networks for these purposes. This thesis discusses the research on creating a smart transducer network architecture concept that uses Power over Ethernet (PoE) as a method for transferring data and power over a single medium, together with principles of decentralized and remote computing for data processing. The concept prototype had its power supplied by a Cisco Catalyst 4507R+E switch and utilized cloud computing to provide an easily scalable and adaptable architecture, able to easily adapt to a wide array of applications and fit the demands of new trends or integration into other systems. The set-up was tested on RaspberryPi and BeagleBone microcontroller boards as sensor hubs, and used DigitalOcean as the cloud computing service of choice. The server in this implementation acts as user interface, front end, and as the console unit back end. The architecture has demonstrated the feasibility of the concept of uniting PoE and IoT to create a flexible architecture. The system has also demonstrated fast communication times, below 200ms, in a cross-continental setting and the ability to provide fast processing times of under 1s. This

shows particular promise for the use of the architecture within the context of green and smart housing equipped with a low-cost sensor network, where local power is partially provided via renewable energy harvesting, and the majority of short-range power transmission is DC-centered.

Preface

A version of chapters 2-5 is published as part of the IEEE Annual Information Technology, Electronics and Mobile Communication Conference 2016 (I. Lobachev, and E.Cretu, ”Smart Sensor Network for Smart Buildings”, IEEE Annual Information Technology, Electronics and Mobile Communication Conference 2016) [1], where the publication received the *Best Paper Award*. I was the lead investigator and responsible for the composition of majority of the manuscript.

Table of Contents

List of Tables

List of Figures

Glossary

This glossary contains the acronyms that are used in this work and their definitions.

CAT5E Category 5 Enhanced Ethernet Cable

CAT5 Category 5 Ethernet Cable

CAT6 Category 6 Ethernet Cable

DARPA Defense Advanced Research Projects Agency of the United States of America

DC Direct Current

FTP File Transfer Protocol-used to transfer files that are hosted remotely

HDF5 Hierarchical Data Format - version 5

HVAC Heating, Ventilation and Air Conditioning system - a centralized system used to perform climate control within buildings

ISM Industrial, Scientific, and Medical radio band

ISP Internet Service Provider

IoT Internet of Things

JSON JavaScript Object Notation

MCB Microcontroller Board - a miniaturized, self sufficient computing device able to perform a certain array of actions

NI National Instruments

PoE Power over Ethernet

RF Radio Frequency

RTE Real Time Ethernet

SDN Software Defined Networking - an approach where a routing scheme of a network can be implemented by using software rather than hardware

SHF Super High Frequency radio band

SQL Structured Query Language

SSH Secure Shell Script

TCL Tool Command Language

TDMS Technical Data Management and Streaming

UHF Ultra High Frequency

UPOE Universal Power Over Ethernet

XML Extensible Markup Language - a file format used for configuration files

Acknowledgments

I would like to thank my supervisor Edmond Cretu, for his support and guidance during my research. The freedom Edmond gave me to explore tangential topics allowed me to broaden my knowledge of the field as well as come up with innovative ways to approach the presented problem. His suggestions to explore green technology and nano materials have lead me to discover technology and research areas I was previously not fully aware of. As a result I believe my broadened view of the industry and the associated fields as well as the state of the art technology that is being developed will greatly benefit me in the future, be it in the area of IT software and hardware development or continuing on as a researcher.

I would also like to extend a special and heartfelt thank you to my family and my beloved girlfriend who are a major reason why I am where I am today. Their support and advice throughout my undergraduate and graduate careers has allowed me to come through some fairly difficult situations and accomplish the goals that I set for myself. I wish to especially thank my father Michael, who has been my teacher and my mentor throughout my life, for his wisdom and counsel, experience that he has passed down to me and most importantly for giving me courage and confidence to set the bar high, and the motivation to surpass my own limits, even when I thought I had reached my wit's end.

In addition I would like to thank Hoschule Augsburg for collaborating with me, in order to test the system in a cross-continental deployment. This partnership provided me with a wonderful experience in addition to the well executed tests.

Lastly I would like to acknowledge Dr. Paul Lusina who initially with a group of co-op students worked with my supervisor Dr. Edmond Cretu, to look into similar tangents of technology and investigated Power over Ethernet.

Chapter 1

Introduction

1.1 Motivation of the Research

Sensors have become indispensable elements in today's society, being utilized in nearly every electronic device that is being used. Due to this situation there has been a great advancement in the field, and today there is an innumerable variety of sensors for every application imaginable. As technology moves forward, so do the trends and user demands. One of such trends is smart buildings, Internet of Things (IoT), as well as the new ”mega buildings” [5] we are seeing more and more of, along with the need to gather data on the state of the building and its affairs. Another trend is ”ecological buildings” or tall wooden buildings which have been an object of interest particularly in areas with a high supply of lumber such as British Columbia. Furthermore as noted in a report by the United Nations, by 2050 over 65 percent of the world's population will live in urban areas according to their projections [6]. This means that the cities will continue to evolve into smart cities, expressing a need for smart houses and buildings, green and environmentally friendly houses which, along with infrastructure operations such as traffic and environment control and oversight, will need to be controlled and monitored by a non-disjointed system. Such large scale systems are bound to be infrastructure dependent, and potentially very costly both in instillation and maintenance.

By utilizing the technology already available today, it is possible to develop a system that, by joining the strengths of the individual elements, would be able to overcome their individual weak points and disadvantages of implementation. For example Power over Ethernet (PoE) has the ability to unify data communication and power transmission, therefore reducing the amount of wiring necessary and reducing power consumption. Devising a system that is able to offer the aforementioned joining of the concepts in order to improve scalability and deployability, among other things, brings large benefits to the field of engineering in general, and to the structural industry in particular. The need to have better control of the building's resources used, track usage statistics, alert concerned parties in cases of emergency, and the overall need to improve safety and comfort, are all strong motivators behind the system described in this work.

1.2 Research Objectives

The main objectives of this work are to develop and implement as a case study a flexible and scalable system architecture that would be able to unify the benefits provided by IoT, along with power and installation efficiency of PoE. PoE has demonstrated the possibility to reduce installation costs for certain types of applications, such as LAN, IP and VoIP [4]. Furthermore, the necessary infrastructure for a PoE-enabled system is already in place for any building that has internet access thorough Ethernet ports, as the same Ethernet connections can be used for both power and data transmission via PoE standards. The flexibility of the system architecture needs to adapt to the vast variety of sensor types and configurations, to allow various sensors to be connected or disconnected to an existing operating network. One of objectives includes investigating the applicability and usability of some of the off-the-shelf smart sensors applicable to structural monitoring applications. In addition, the research aims to base the new concept on the lessons learned from previous iterations, in order to improve aspects such as scalability, adjustability, and ease of deployment and exploitation. Once the prototype of the architecture concept has been developed, it was tested to verify the feasibility of use of such a system, and its possible deployment schemes. In particular the testing attempted to answer whether the new design concept is able to adhere to the paradigm of high degrees of scalability and adjustability. The prototype system passed the testing stage, the details of which can be found in Chapter 4. As a result the developed concept can be used to later create a commercial system, a hybrid sensor network (as enabled by the designed system architecture), combining the abilities and properties of both wireless sensor networks as well as hard-wired sensor networks and their respective sensor modules.

1.3 Organization of the Thesis

The thesis is divided into five chapters.

Chapter 1 Introduction: The motivation and objectives of the research work are discussed in this chapter.

Chapter 2 Background and Related Work: The chapter discusses previously developed systems with similar goals or functionality, as well as the previous work in our research group that has contributed to shape the current system perspective.

Chapter 3 Concept and Implementation: The developed system architecture and an example of its practical implementation are discussed in this chapter. The chapter also looks closely at the inner workings of the developed algorithms, the visual representation and user interface.

Chapter 4 Testing and Applications: The chapter discusses a testing framework, its implementation and the results of the testing, together with their interpretation. Overall the design has demonstrated favorable results with regards to scalability, as it operated within acceptable margins, and has demonstrated that using a cloud-computing controller client is a positive improvement over a hard-wired connection to a local, physical machine.

Chapter 5 Conclusions and Future Work: The conclusions of this work, as well as recommendations for future work and some proposals on the current design improvements, are given in this chapter.

Some additional representative figures of the system, along with additional information and diagrams regarding the principles of operation of PoE can be found in the appendix section of this work.

Chapter 2

Background and Related Work

Small scale sensor networks have been previously investigated in various contexts and application areas, along with smart sensors and monitoring systems. This chapter will discuss some of the previously conducted work in this field, outline some of the shortcomings of the discussed systems and the lessons learned that were used when constructing the currently implemented concept discussed in Chapter 3.

2.1 Wireless Sensor Networks

There is a large number of wireless sensor network configurations that has been previously implemented, which vary by their intended application and architecture design. This list includes:

- **Smart Wireless Sensor Networks** - where all or the majority of the data transfer is done wirelessly. The system utilizes smart sensors for its network - an example from everyday life can be a Wi-Fi (IEEE 802.15.4 standard) or Bluetooth network of devices. The primary criteria to assign this label to a system is that sensors communication needs to occur wirelessly.

- **Large-scale Sensor Networks** - this label can be given to sensor networks that span large areas. For example the BC Hydro *Smart Grid* [1] would be an appropriate everyday life representative.

[1] https://www.bchydro.com/accounts-billing/rates-energy-use/electricity-meters.html

- **Mimicking Sensor Networks** - sensor networks that try to mimic the behavior of a certain system set-up while using a fundamentally different principle can be placed into this category. An example can be a wireless sensor network within a room, which would use the same positioning scheme as well as routing principles as the wired equivalent.

- **Small-scale Sensor Networks** - Sensor networks which possess only a few sensors in their network can be placed in this category. An example can be Bluetooth enabled, small scale wearable devices, such as the FitBit[2], or sensors in a small house. Such sensors can be light intensity or air quality monitoring sensors, which use wireless communication to transfer the data.

Each one of these is constrained by design decisions made for their implementation, and are often limited to the particular application that was originally intended. A certain architecture may have several attributes, depending on the components used and on the possible or intended applications.

2.1.1 Smart Sensors for Wireless Sensor Networks

A good example of a smart sensor that has been previously developed is the Imote2 smart sensor platform developed by Jennifer Rice et.al. [7]. Their work focused on a wireless sensor network setup, differentiated from other solutions through the approach used. Primarily, they have employed a decentralized computing scheme to reduce the amount of data communication within the network; as a result, the architecture has reduced the amount of power consumed by the system through limiting the power usage of individual elements. However, while the system does offer benefits over traditional wireless setups (where the design would attempt to mimic a wired configuration using wireless transmission), it still has its flaws - While the group managed to reduce the overall power consumption, the main problem becomes the maintenance of the energy sources, as the sources of power in these cases are local batteries, regardless of whether they are primary or secondary cells[3]. Furthermore, the possible issue of interference and lost or deteriorated signals is

[2] https://www.fitbit.com/ca

[3] ***Primary cells*** are batteries that are non-rechargeable, while ***Secondary cells*** are batteries that can be recharged, such as the lithium ion batteries in smartphones, for example.

still not fully eliminated, as it is in the nature of wireless transmission. A related Bluetooth wireless network concept was proposed by Oliver Kasten et. al. in [8], where they have also implemented the idea of on-board data processing and sensing. Their primary focus was the development of a concept called *Smart-It*, where one could add ”smartness” to an object in an post-hoc[4] way. The authors utilized Bluetooth technology for the communication needs of their system.

Other attempts at sensor networks in the recent years focus primarily on wireless sensors and data transmission[9]. Nevertheless, many of the investigators tackle the issues of power consumption and scalability via routing protocol manipulation and various kinds of scheduling application layers, such as the tree structure architecture proposed in [10] or the approaches in [11]. A number of recently developed solutions target more specific applications, such as wireless sensor networks for home lighting systems[12], industrial wireless sensor network data transmission scheme in [13] or the case study on Greenorbs in [14]. Which looked into the issues of scalability and system dependence on individual nodes, an issue discussed in Chapter 3 and which our architecture adresses.

2.1.2 Semi-wireless Sensor Networks

Another approach for monitoring a building, in particular for structural health monitoring, has been implemented by Xu et. al. with their system named *Wisden* [15]. The system was then later improved and tested again by Chintalapudi et. al. who published their results in [16]. However the system cannot be called a truly wireless one, as there is a certain wiring setup involved in the system. The system uses wireless sensors, which send the data to a local hub, or node as titled by the authors, which is then connected to a local PC via serial port, a hard-wired connection. The computer would then handle the bulk of the data processing and can be connected as a network to other computers employing a similar set-up. This design shows a number of drawbacks, such as the need for the wiring, which defeats the purpose of using a wireless set up. In addition, the serial communication as implemented in this work, as well as the semi-centralization of the data processing may, on a larger

[4]**Post-hoc** in this context would refer to an installation scenario where the installation of the additional device happens after the manufacturing process, possibly by non-professionals in an ”at-home” setting

scale, greatly affect the speed and performance of the system as a whole [17]. Furthermore, the wiring that is required for the set up, as well as the need of separate power supplies for the nodes, in addition to the power required by the sensors, make this a good intermediate step; it is however by far not the optimal solution for the purposes of large scale building monitoring with large sensor clusters.

One possible approach to resolving the issue of power deficiency in semi-wireless sensor networks is wireless power transmission. Research done by Visser et.al. on the use of Radio Frequency (RF) to harvest energy locally and transmit it to low-power wireless sensors [18] makes the idea of using a semi-wireless or hybrid configuration solution a lot more appealing, as it adds another alternative to supply the necessary power while retaining the flexibility of node placement offered by wireless data transmission.

Lastly none of the solutions that have been previously mentioned provide a hybrid integration of sensor and actuators, for complex control systems, which was also pointed out by the authors. This is due to a number of reasons - one of them is the fact that the system would not have enough power to actuate anything of importance, since one of the primary goals highlighted by all of the authors is to minimize power consumption. The other factor that reinforces this fact is that in order to provide power to the actuating device, a wired connection will be required, which may defeat the purpose of setting up a wireless system in the first place. However the lack of actuation capabilities is understandable as the concept was outside of the original scope decided by the authors and not their primary focus.

2.1.3 Commonplace Wireless Networks

A certain number of wireless sensor networks have already integrated themselves into our everyday life. While we may not realize it consciously, we use wireless sensor networks for a wide variety of tasks. A Wi-Fi network, for example, is a wireless sensor network which monitors and exchanges data with the devices connected to it via RF signal transmission[5]. Another common sensor network, which many not immediately come to mind is the GSM communications network, as well

[5]Wi-Fi communicates mainly using the 2.4 gigahertz (12 cm) Ultra High Frequency (UHF) and 5 gigahertz (6 cm) Super High Frequency (SHF) Industrial, Scientific, and Medical (ISM) radio bands

as the BC Hydro *Smart Grid* as mentioned previously. In addition, many of the new gadgets, electronic wrist bands, arm bands, smart bracelets etc. that allow a person to monitor their heart rate, steps taken and calories burned can also be considered a wireless sensor network, typically using Bluetooth communication standard for data exchange. In the medical field we have for instance pacemakers and blood sugar meters, which can again provide relevant data when wirelessly communicating with a network. All of these are often centered around a smartphone, which compiles all of the data for the user to see, as well as take other actions, such as send alerts to a local hospital or notify the user to alter their current activity in order to circumvent the risk. A great overview of the state and direction of this technology is provided by Pantelopoulos et. al [19], which discusses current technology of wireless sensing for biomedical applications and the trends they believe to be present.

2.2 Wired Sensor Networks

As seen previously, there are a number of approaches that can be taken in wireless sensor network systems. However, it is important to remember their basics and predecessors when constructing a design, which are the wired sensor networks. Such sensor networks have been used since the early 1950s, as part of the national power grid in the US or the radar network used for air traffic control [20]. The sensor networks as we see and define them today, have been more intensively researched and used during the cold war by the Defense Advanced Research Projects Agency (DARPA), as shown in their investigative work of the history of sensor networks by Chong et. al. [20], and onward by many institutions and enterprises. The primary concept of wired sensor networks is to have an array or cluster of sensors physically connected to a processing hub, which will output some results based on the data received. A good reference table that can be used when considering the design aspects and requirements of a sensor network can be found below in Table 2.1. It lists the major aspects that should be considered when making the design decisions and classifying the system, it will also be referenced throughout this work when consulting certain guidelines.

Table 2.1: Aspects and Attributes of Sensor Networks to be Considered During the Design and Evaluation Stages

Aspect	Considerations and Variations
Sensors	***Size:*** small (for example micro electromechanical systems (MEMS), large (for example radar or satellite network, city grid)) ***Type:*** passive (for example strain gauges, piezoelectric, magnetic or acoustic sensors), or active (for example radar, sonar, Bluetooth) ***Composition:*** homogeneous (where all of the sensors are of the same type) or heterogeneous (where different types of sensors are employed) ***Spatial Coverage:*** Dense (sensors are closely packed together for example on a CPU or a small wearable/portable device), sparse (for example temperature sensors scattered across a forest or a city) ***Dynamics:*** stationary (for example building monitoring) or mobile (for example on a vehicle or a robot)
Sensing Entities of Interest	Organization: Distributed (for example environmental monitoring, building monitoring), localized (for example target tracking, individual location monitoring (e.g. room)) ***Mobility:*** static, dynamic ***Nature:*** cooperative (for example air traffic control, HVAC systems), non-cooperative(for example military targets, animal tracking etc.)

Continued on next page

Table 2.1 – continued from previous page

Aspect	Considerations and Variations
Operating Environment	Benign (for example interior of rooms of a building, within machines that shield them (ex: computers) etc.), adverse (for example aggressive outdoor environment such as a mine, ocean waters, places of military action etc.)
Communication	***Networking:*** wired (using wired connections between nodes and sensors), wireless (using wireless communication between nodes and sensors, for example Bluetooth or RF) ***Bandwidth Availability:*** high, low ***Bandwidth Usage:*** heavy (constant big loads of data transferred ex: images, videos etc.), Light (small continuous transmission of data ex: text or binary data)
Processing Architecture	Centralized (all of the data is sent to and processed at one single central location), distributed (the processing and data management is done at the sensor level and is distributed between the different locations), hybrid (a hybrid of the previous two approaches)
Energy Consumption	High (big power drain due to frequent wireless transmissions, high density sensor arrays, inefficient power management etc.), low (small scale design, low energy design, small sensor arrays with few sensors, efficient power management)

Continued on next page

Table 2.1 – continued from previous page

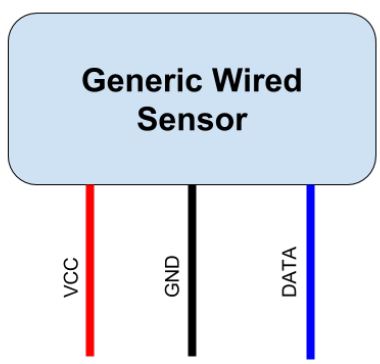
Aspect	Considerations and Variations
Energy Availability	Readily available (power sockets, available power lines, relevant infrastructure to be connected to), scarcely available (remote locations or places fare from infrastructure that would require battery and/or renewable energy sources for power supply)
Deployment Infrastructure	Fixed and Planned (for example a factory or a smart house where the infrastructure for the sensors is built into the design of the structure) or ad hoc (post-construction installation for example during a building upgrade, or random deployment through other means, no designated infrastructure would be present)
Ease of Modification	High (components can be added or removed with ease, there is no need for a full recompile or reassembly of the system or source code, modifications are simple and localized), Low (need for additional infrastructural to integrate the modification, major parts or whole assembly needs to be adjusted and/or recompiled, modifications are spread across the whole system due to high tangibility and dependencies)

The following subsections will discuss the varying approaches to wired sensor networks. We have briefly analyzed their evolution afterwards focusing on one of the primary aspects that is of importance to this work which is Power over Ethernet (PoE) and the technology around it.

2.2.1 Wired Sensor Networks With Dedicated Cabling

A typical approach to physically implement a wired sensor network is to have separate, dedicated cabling for power and data. This sort of setup would be used when either AC or DC power is easily attainable locally (without the use of batteries), and where the speed of the data transfer is important and preferred to be real-time, or near real time. Usually, this is accomplished through bus connections[6], although other wiring approaches, such as completely separated wiring, have also been used in the past. As seen in Figure 2.1, a generic set up would have a minimum of three wires for power and data transfer. The power can be drawn either from the controller board that serves as a node, a central controller or form a separate source. In any of those cases a certain degree of voltage control may be required to fit the needs of the sensor, the latter case may occur when the sensor will require more current than can be provided by the board. Such setup is often used in thermostats as well as Heating, Ventilation and Air Conditioning system (HVAC).

Figure 2.1: A graphic representation of a generic wired sensor



2.2.2 Power Over Ethernet (PoE) and its Applications

The concept of Power over Ethernet (PoE) has evolved from the Ethernet data cable, with the goal to integrate both power and data communication over the same bus wiring. Since the data component is handled by the Ethernet protocol portion of the design, the concept was to use some of the wires within the Category 5 Ethernet

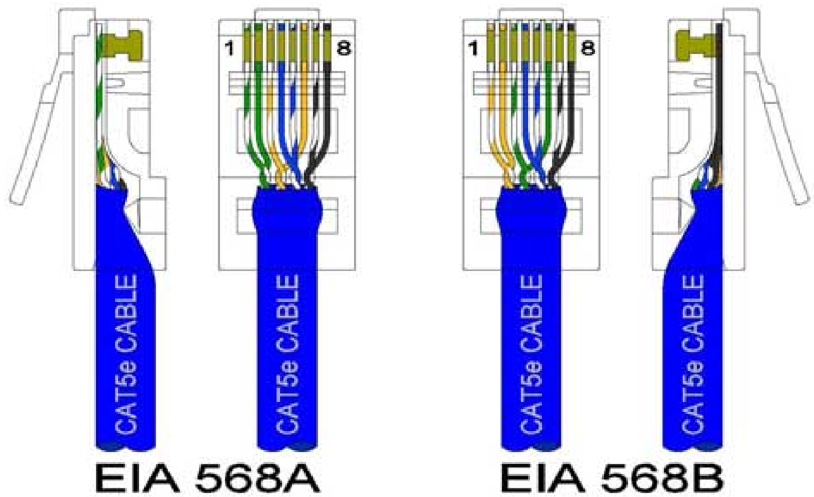
[6] In this context a **bus** refers to parallel electrical wires with multiple connections

Cable (CAT5) to transmit the power. Cisco had initially developed the technology for VoIP applications, with only 13W-15W of power being supplied. However, as the scope of applications grew and the technology developed, so did the amount of power that could be supplied, with the latest performance achieving 98W of power delivery, as will be discussed later. To briefly explain the principle of PoE and how it works with a standard Category 5 Enhanced Ethernet Cable (CAT5E) cable version, a short introduction will be provided here. There are typically eight wires, or four twisted pairs, switches[7] make use of some of these pairs to transmit the power to the connected devices. A diagram showing the pairs on the Ethernet cable can be see in Figure 2.2, where the pairs of cables 4,5 and 7,8 are used for power transmission in PoE. The two standards EIA568A and EIA568B are essentially identical in functionality and performance, their major difference being rather the used color scheme. As the details of the wiring are outside of the main scope of interest of this work, we will not go into further detail on this topic.

Once the switch capable of delivering PoE is available, the other aspect that needs to be considered is the array of devices that will be connected to it. The device on the other end of the cable, would need to be able to receive power over Ethernet, and communicate back. The common requirements for devices desiring to utilize PoE features are outlined in the Cisco specification documents[21], and further details on the specific operations of PoE can also be found at [22] - this paper goes into more detail on the protocols that can be used with PoE systems, along with some of the history of development and current standards. Currently the most widespread used cable is CAT5E, which is often shortened to its deprecated predecessor CAT5; in the following chapters, when referring to Category 5 Enhanced Ethernet Cable the CAT5 acronym will be used, for the purpose of simplicity. It is important to note that there is a raising popularity of PoE, as it is now used for wireless routers, phones, security cameras, and in some cases even lighting[23][24], all due to the increase of power that can be supplied through a single port. The new IEEE 802.3bt standard allows to supply up to 60W (level 3) using CAT5E, which is the power supply that was used in this work, and level 4

[7] A network switch (also called switching hub, bridging hub, officially MAC bridge) is a computer networking device that connects devices together on a computer network, by using packet switching to receive, process and forward data to the destination device.

Figure 2.2: A diagram showing the the cable wiring and pairs within an Ethernet CAT5E cable[2]



EIA 568A		
Pin #	Wire Color Legend	Signal
1	White/Green	TX+
2	Green	TX-
3	White/Orange	RX+
4	Blue	TRD2+
5	White/Blue	TRD2-
6	Orange	RX-
7	White/Brown	TRD3+
8	Brown	TRD3-

EIA 568B		
Pin #	Wire Color Legend	Signal
1	White/Orange	TX+
2	Orange	TX-
3	White/Greeage	RX+
4	Blue	TRD2+
5	White/Blue	TRD2-
6	Green	RX-
7	White/Brown	TRD3+
8	Brown	TRD3-

power of 90-100W which can be used with a Category 6 Ethernet Cable (CAT6) cable for higher power application [25][26]. If even more power is required, it can be delivered by combining some of the ports, therefore doubling or tripling the power supply; this in turn complicates the setup process and the hardware required, but broadens the possible applications. All of these aspects, as well as the research being performed right now for Direct Current (DC) nanogrids [27], and renewable energy sources, makes PoE an attractive method for delivering and distributing power now, and even more so in the future. In the context of smart houses, the power delivery capabilities through PoE-enabled networks allows an electrical infrastructure where the DC power and data are physically distributed through the same physical cables throughout the entire building.

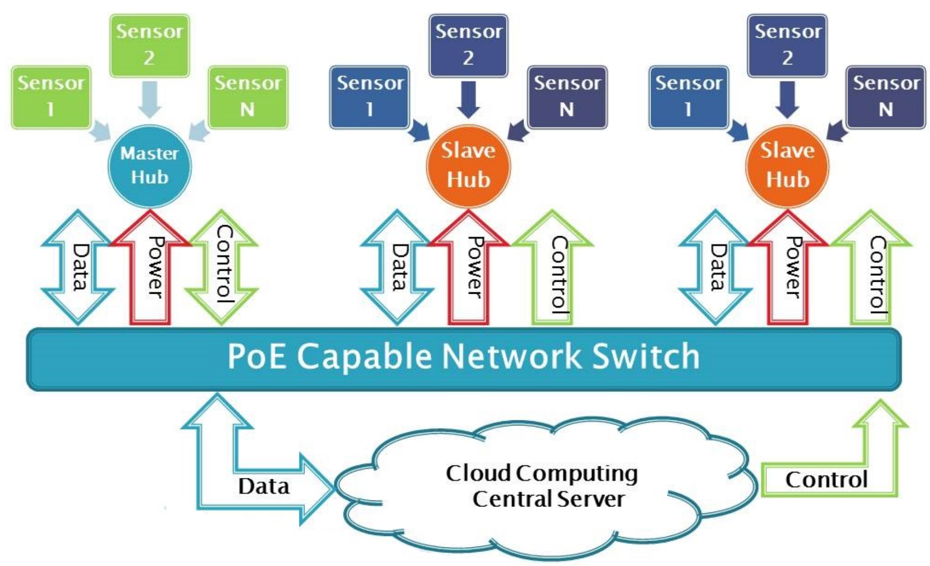
Chapter 3

Concept and Implementation

The concept began with the inception of the system architecture. It needed to have the ability to be scalable, which means that the system had to have the ability to be expanded without major changes, alterations or disruptions. The sensor hubs, with the bigger picture in mind, needed to be easily controlled and configured, and the system itself should be reasonably power efficient and deployable. The power efficiency and deployability were achieved by using PoE, varying sensor hub classes, with the corresponding varying operation schedules, and semi-distributed data processing with a heavy emphasis on cloud computing. The general overview of the architecture can be seen in Figure 3.1. The inter-system communication and linkage, in addition to providing the possibility of hierarchical and remote control, allows for system to be optimized with regards to power consumption and processing cycles, by only powering on the necessary hubs on a per-case basis - the role of a hub is to act as a local sensor module interface, with some additional configuration and local signal processing tasks. Further details on the operation models of operation and classes of the hubs will be discussed in the sections below, and the principle behind the case-dependent activation can be observed in Figure 3.5 and Figure 3.6. The use of PoE allowed for both the data, power and control commands to be sent via a single cable, which is already present in almost any building that has an internet connection infrastructure.

The setup of the prototype used to test the validity and feasibility of the concept proposed in this work has used a Cisco Catalyst 4507R+E switch [28] to handle

Figure 3.1: General architecture view

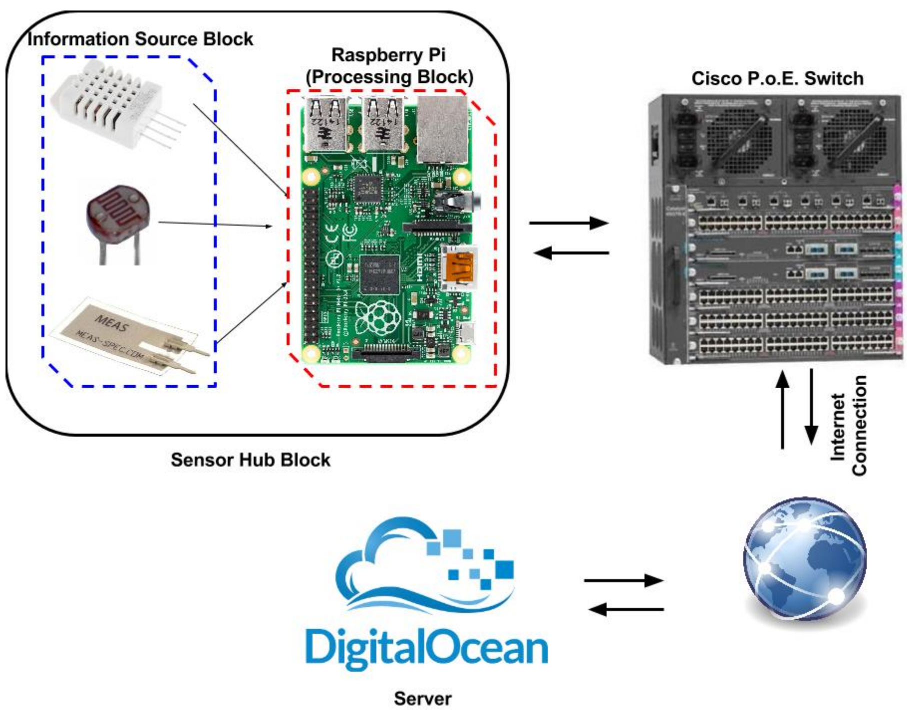


the routing and supply the power to the sensor hub network [29]. The Catalyst switches of these series are capable of delivering up to 60W per port using Universal Power Over Ethernet (UPOE), which essentially doubles the per-port power output specified by the PoE standard [30] . In this setup a hub is a Microcontroller Board (MCB) capable of receiving PoE, either by means of a shield[1] or an integrated adapter, and is capable of housing the Linux kernel. While the current prototype used an Ubuntu distribution, a lighter operating system, such as Puppy Linux[2] or Archive Linux, ArchLinux [3], can be used on MCBs with less resources, as well as scenarios where the size of the hub has higher priority over the amount of local processing that is necessary. A good example of such a scenario would be the slave class hubs, which will be discussed later in this chapter. Due to their tasks

[1]https://www.pi-supply.com/product/pi-poe-switch-hat-power-over-ethernet-for-raspberry-pi/
[2]http://puppylinux.org/
[3]https://www.archlinux.org/

being non-resource-demanding, the proposed scheme of optimization can be used to reduce costs. The booting sequence of the operating system on the MCBs used in the experiment was modified to reduce booting time, by preventing programs and services unnecessary for the functionality of the system from loading. A high level overview diagram of the system concept can be found in the Figure 3.2 below. The figure outlines the overall set-up that was assembled, while later a lower level description will be provided to accompany the discussed details.

Figure 3.2: A high level diagram of the implemented system set-up



3.1 Design Considerations and Decisions

For the prototype that was used to test the concept, sensors which can give relevant data about the state of a building or a room were used. In particular such attributes as the temperature, humidity, vibration on the walls as well as illumination were measured. These attributes were chosen in order to simulate real deployment, where the system may be used either to make a room or building ”smart”, monitor structural integrity, or both. A strain gauge sensor was considered but not used, as the one necessary for building monitoring would need to be of wide are of effect, to measure things such as settlement of a building. Due to the fact that such a sensor is not currently readily available, within economical constraints that would be feasible for a prototype test, this portion of the testing can be conducted at a later stage of development. Hardware components, aside from the Cisco switch and sensors, consisted of a Raspberry Pi microcontroller board, which acted as an MCB, and a Cisco UPoE power splitter[30]. The power splitter was used to separate the data and the power supply, as Raspberry Pi does not natively support PoE, and to lower the voltage supplied to the board over the Ethernet cable (48V) down to 12V DC, which is one of the operational voltages of this particular MCB. Raspberry Pi was chosen as the MCB for the project due to its economical benefits, wide range of capabilities and features, as well as a large support community and low level access to the core of the board. The low level access was a very desired attribute, as it allowed to alter and modify performance and operational routines. Overall this resulted in Raspberry Pi being the optimal choice in terms of price-to-performance ratio.

Complementary to the hardware architecture, there were a number of considerations and design decisions that needed to be made regarding the software component; in particular, the following aspects have shaped the project:

1. The format to be used for the configuration files - the format had to be adaptive in terms of operating environment, be able to store metadata[4] about the components, and be easily convertible to other formats if necessary, including manual inspection. Such configuration file formats as the INI[5] file format, Extensible Markup Language (XML) and JavaScript Object Notation (JSON).

2. The database structure and format to be used, as the winning contestant would need to house large amounts of scientific data, and be able to quickly

[4]**Metadata** - a set of data that describes and gives information about other data.
[5]**INI** file format is an informal standard for configuration files for some platforms or software.

and efficiently traverse and possibly process it. The operation would also occur online, as such the constraints imposed with that need to be heeded. The considered options were the Hierarchical Data Format (Hierarchical Data Format - version 5 (HDF5)), an open standard originally developed at the National Center for Supercomputing Applications and supported by Mathworks, the Technical Data Management and Streaming (TDMS) file format [31] by National Instruments (National Instruments (NI)), an open format widely used for LabView applications, and as such could potentially be a good solution. Another open format that was considered was the ROOT file format[6], developed by CERN[7]. While a ROOT file is a list of consecutive data records, labeled TKeys by the developers, ROOT itself is a modular scientific software framework, which is written primarily in C++, but certain instances are also available for Python[8] and R[9]. Lastly, a more traditional Structured Query Language (SQL) database was considered for storing and processing data.

3. The last component of the system to be decided is the central processor. According to the concept this processor would be virtual and scalable, as well as accessible from any internet connected device. As such a number of cloud computing services were considered, as well as the option of locally hosted server. For cloud computing services the EC2[10] provided by Amazon INC as well as DigitalOcean[11] were investigated as potential solutions.

After careful consideration and investigation, JSON (JavaScript Object Notation) was chosen as the configuration file format. The INI file extension, while convenient on some platforms was not as versatile or adaptable, and it did not provide an easy hierarchy between the various fields. As for the XML format, while possessing similar properties, its use has deviated to other applications, and being a predecessor to JSON, it was decided that it would not be the best candidate

[6]https://root.cern.ch/
[7]**CERN** - The European Organization for Nuclear Research
[8]A programming language - https://www.python.org/
[9]**R** - a programming language for statistical computing and graphics - https://www.r-project.org/
[10]https://aws.amazon.com/ec2/
[11]https://www.digitalocean.com/

for the task. As a result the versatility, convertibility and its wide acceptance, its hierarchical properties, along with the fact that its intended use aligned with our application, have driven the decision to use JSON as file format for the configuration files for the transducer nodes in the network. When choosing the database that will be used for the collected data in this architecture concept, a first candidate was NI TDMS; while possessing some good quality regarding hierarchical organization, data control and efficient read-write access of very large scientific data, it was perceived as too restricted in its use and integration, and widespread use (its primary application was with NI software). This format was considered non-ideal for the intended design, which aims to increase adaptability and scalability of sensor networks with minimal restrictions. The ROOT system, on the other hand, has shown great possibilities, and the tools that come with the platform may make it a suitable candidate for a future improvement of the design. However, at its current stage, the complexity of the data structure as presented by ROOT, along with the associated integration difficulties, caused this format to not be the number one choice for this particular design. Lastly, SQL database format is very widely used for web applications; however, heavy customization or a design scheme would be needed to add metadata or hierarchy to the collected data. While it would provide an ease of deployment and integration with other systems, which satisfies the adaptability requirement, the lack of hierarchy and metadata capabilities caused this variant to be disregarded as well. As such, the HDF5 format, which posses both the metadata capabilities as well as hierarchical data structure, in addition to being lightweight for file transfers and easily used with other scientific software for data analysis (such as MatLab), made it the ideal choice to be used for the data structure. The need for flexible export and import capabilities stems form the fact that many researchers use a variety of different tools to conduct their research and store and evaluate their data. Therefore a file format which can be easily shared between researchers from different institutions, and that has a wide support from the open source community, such as the HDF5, would be ideal for the application of a widely deployed sensor network, and hence was chosen. The locally hosted server was ultimately disregarded, due to a number of reasons. First of all, it would not allow for the system to be adequately tested to verify its scalability, in particular the international deployment, as discussed in Chapter 4, as the results would be too skewed to allow adequate assessment. Secondly, the limitations of a single piece of hardware, since the server would be set up as a local desktop computer, in addition to possible service disruptions (due to hardware, software or local network failure) made it an undesirable solution for housing the developed system. When considering cloud computing, both EC2 and DigitalOcean provided similar services. The primary advantage of EC2 would be the ability of rapid increase in processing power when necessary, which can be very beneficial on a larger scale of deployment, and may be used in the future. However this service provided economic drawbacks which, along with the dependability and security of DigitalOcean, caused the latter to be the cloud computing service of choice. These decisions led to the creation of an architecture concept that was implemented in a prototype system, the basic requirements of which will be discussed below in Section 3.2.

3.2 System Requirements

There is a short list of requirements for the components that can be used to set up the system. In particular the MCB should have:

1. **Sufficient processing capabilities** - The MCB, which will serve as a sensor hub module needs to be able to run either Linux, or some operation system that is capable of connecting to the internet and executing python scripts (in the current iteration). It also needs to able to process and package the data into HDF5 format for sending.

2. **An interface to connect the sensors** - The MCB needs the capability to connect the various types of sensors that are necessary for a given task. The connection and corresponding peripherals can be either wired or wireless by nature, depending on the MCB's capabilities and other requirements and restraints of a given application.

3. **Sufficient storage** - This aspect needs to be considered for MCBs that will operate in semi-autonomous and autonomous modes, as described in Section 3.5. The storage requirement in this case will be dictated by the amount of information to be processed and stored, which in turn is determined by

the application. In the current set-up the dedicated storage for the data was 4Gb.

4. **PoE capable** - last major requirement is the ability to operate with PoE. This can be accomplished either by a built-in circuitry that the MCB has, or by a means of a shield, such as the Pi POE Switch hat for Raspberry Pi, or a Cisco Catalyst Power Splitter for many of the other boards.

Figure 3.3: A snipped of the JSON code that is used to store configuration settings

```
"masters": [{    //the array for MCB
        "menu": [{    //first tab level, is always "menu"
            "submenu": [{  //second tab level and any other level thereafter is "submenu"
                "Name": "X",  //the name of the parameter
                "Active": "true", //whether the parameter is on/off
                "Critical": "true",  //whether the parameter is time critical
                "Pin": "25",    //the pin or port of the paramter on the MCB
                "Mode": "semiauto",   //the mode of operation of the MCB
                "Type": "parameter", //the type of object: paramter, cluster, master, slave
                "IP": "",    //ip address of the MCB
            "Master": "",      //ip address of the master if slave, of the switch if master
                "Email": "XXX@YYY.com",    //email of the person to be notified
                "url": "192.168.2.20/acceleration/x", //path to the parameter in the HDF5 file
                "namesen": "AccX,rate,active",    //names of the headers to be monitored
                "MinNorm": "0.1,1,0.9",  //minimum thresholds for the specified parameters(same order)
                "MaxNorm": "0.6,3,1.1"  //maximum  thresholds for th specified parameters (same order)
            },
            ...]
```

3.3 Sensor Hub Classes

There are two general classes of sensor hubs in this system set-up. A master class sensor hub and a slave class sensor hub. The naming convention was used due to the behavior of the system as described below.

1. **Master** - is a sensor hub class which can operate in either fully autonomous or semi-autonomous modes of operation, which will be discussed in further detail in 3.5. In order for a hub to be assigned in this class, it needs sufficient processing power - the amount is dictated by the exact application and performance constraints. While in possession of this class label, the hub may issue commands to the slave class hubs assigned to it, for example to gather

more data. In addition, it will also be capable of bidirectional communication with the server.

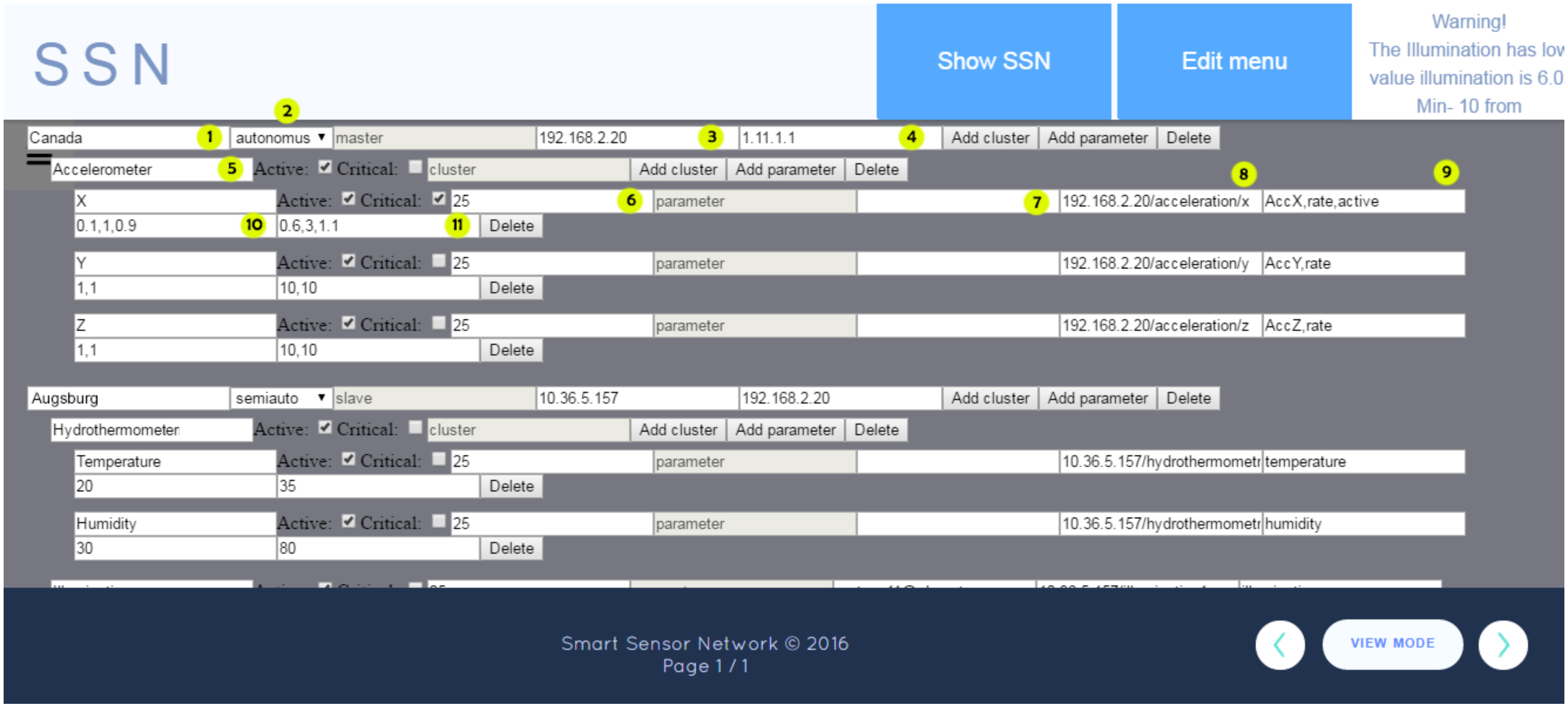
2. **Slave** - is a sensor hub class which is capable only of one mode of operation. However, while it is unable to perform complex or resource demanding tasks locally, it provides the benefit of a lighter build. This in turn means that the hub may have much smaller physical dimensions and may be more economical, which provides a more financially efficient solution on larger scales.

3.4 Configuration of the Hubs

As seen in Figure 3.4 the implemented edit menu provides the user with the ability to configure all of the sensor hub modules via the web interface, remotely. The changes made there will be carried over into the master configuration file, which is stored on the server. The configuration file uses JavaScript Object Notation (JSON) format to store all the settings and relay them to the respective modules. Once any changes have been made the server will issue an update notification to the MCBs that have been affected, once the MCBs receive the notification and have prepared to receive information the new configuration file for them will be created, sent and applied, upon which the changes will have taken effect. The individual configuration file is created by taking a relevant snippet of the JSON master list file that contains the information of all the MCBs, including the ones that were affected. That snippet is then used to create the file that will be sent. An example snippet of the JSON code that is used in the system can be found in Figure 3.3.

If desired, all of the settings can be altered by directly accessing the back end as well, assuming sufficient access privileges are present. This can be done by establishing an Secure Shell Script (SSH) connection to the server, using the aforementioned credentials with sufficient access privileges, and accessing the master list. Following the access and editing, the changes will be propagated throughout the system in the same manner as when the changes were made via the online interface. A derivative of this method would be to use an File Transfer Protocol (FTP) access program, to gain visual access to the files on the server.

Figure 3.4: A screenshot of the web view of the user interface and configuration window



The numbers highlighted in the figure are used to reference the fields they are located at, where: 1 is the name of the Master or Slave module, 2 is the mode of operation, 3 is the IP address of the module, 4 is the IP address of the corresponding switch or master module, 5 is the name of the cluster, 6 is the pin or port that the cluster is connected to on the board module, 7 is the e-mail address of the person to contact in case of anomalies, 8 is the path within the HDF5 file, 9 are the headings in the HDF5 file that will be monitored, 10 are the minimum thresholds for the specified parameters, 11 are the maximum thresholds for the specified parameters.

3.5 Requirements and Modes of Operation

While the MCB acts a sensor hub, it can house a number of logical clusters of sensors, for example a sensor that provides multiple types of data, such as an accelerometer or a humidity and temperature sensor such as DHT22 [32]. The hierarchy of the system takes into account what is considered a parameter, as is seen in Figure 3.4. The MCB, which, as long as it meets the requirements outlined in Section 3.2, is non-discriminant of the type and model used, communicates directly with the server. The server is located on a cloud computing[12] service host - in the case of this implementation DigitalOcean was used. However, depending on the application, the server can be both outsourced to a third party or hosted locally depending on the available resources and performance needs. In addition to the previously implemented functionality of a master/slave status of the sensor hubs, which focused more on power consumption control, the system now also has three modes of operation, designed to allow it to adapt to almost any scenario. These modes are:

1. **Autonomous** The sensor hub functions individually, conducting all of data processing locally, and performing actuation and/or executing scenario scripts as is dictated by its local code. The hub will periodically send its stored information to the server. When operating in this mode the hub will also periodically probe the server to determine whether it is operational and verify the presence of an active internet connection. If either one of these tests fails to produce a positive response, an error will be logged, noting the details of the discovered failure, and brought to the attention of the person in charge once proper operation is restored and confirmed. A diagram outlining the operational architecture of the algorithm can be seen in Figure 3.5.

2. **Semi-autonomous** The most common mode of operation, typically used by the master class sensor hubs. When operating in this mode, local processing of the data will only be applied to sources of information marked as time-critical, once processing is complete the hub will then act based on the

[12] **Cloud Computing** - the practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer.

results. Data that is categorized as non-time-critical will simply be periodically forwarded to the server by the hub. A diagram outlining the algorithm of this mode of operation can be seen in Figure 3.6.

3. **Follower** The most basic mode in which the hub will simply execute the commands it was given by either the server or the master class hub. The primary function of this class is to continuously forward all of the collected data to the server for processing for as long as it was instructed to do so.

3.6 Parameters, Organization and Data Processing

Each parameter has maximum and minimum threshold values, if the current value of the reading exceeds either of the threshold values an alarm is triggered. A parameter in this context is defined as a data source for a specific hub. The alarm routine visually highlights the value on the web page for manual inspection, as can be seen in Figure 3.9 and sends a notification to the corresponding e-mail of the concerned party. In addition the error is kept in a log of alarms which is present on the main header tab, as can be seen in Figures 3.4, 3.8 and 3.9. The alarm will also trigger an additional data request. This request will wake up all of the slave hubs that are under the master hub which recorded the anomaly. Once awake the slave hubs will collect an additional set of data to either confirm or deny the presence of the anomaly that was picked up from the readings provided by the master class hub. This approach shows signs of being very useful for applications in the civil engineering industry, for monitoring civil structures. In particular it can be used to monitor the health of a bridge as well as a building, especially in cases of emergency such as, for example, an earthquake.

The original system had a very limited speed and scalability due to the utilization of RS232[13] for communication with the client/server which was a local Ubuntu machine, physically connected to the switch[17]. This set up in turn also limited the scalability of the system, as it required there to be a separate, constantly operational machine for every switch. These individual machines would in turn

[13] **RS-232** is a standard for serial communication transmission of data

Figure 3.5: A diagram displaying the high level algorithm of autonomous mode of operation

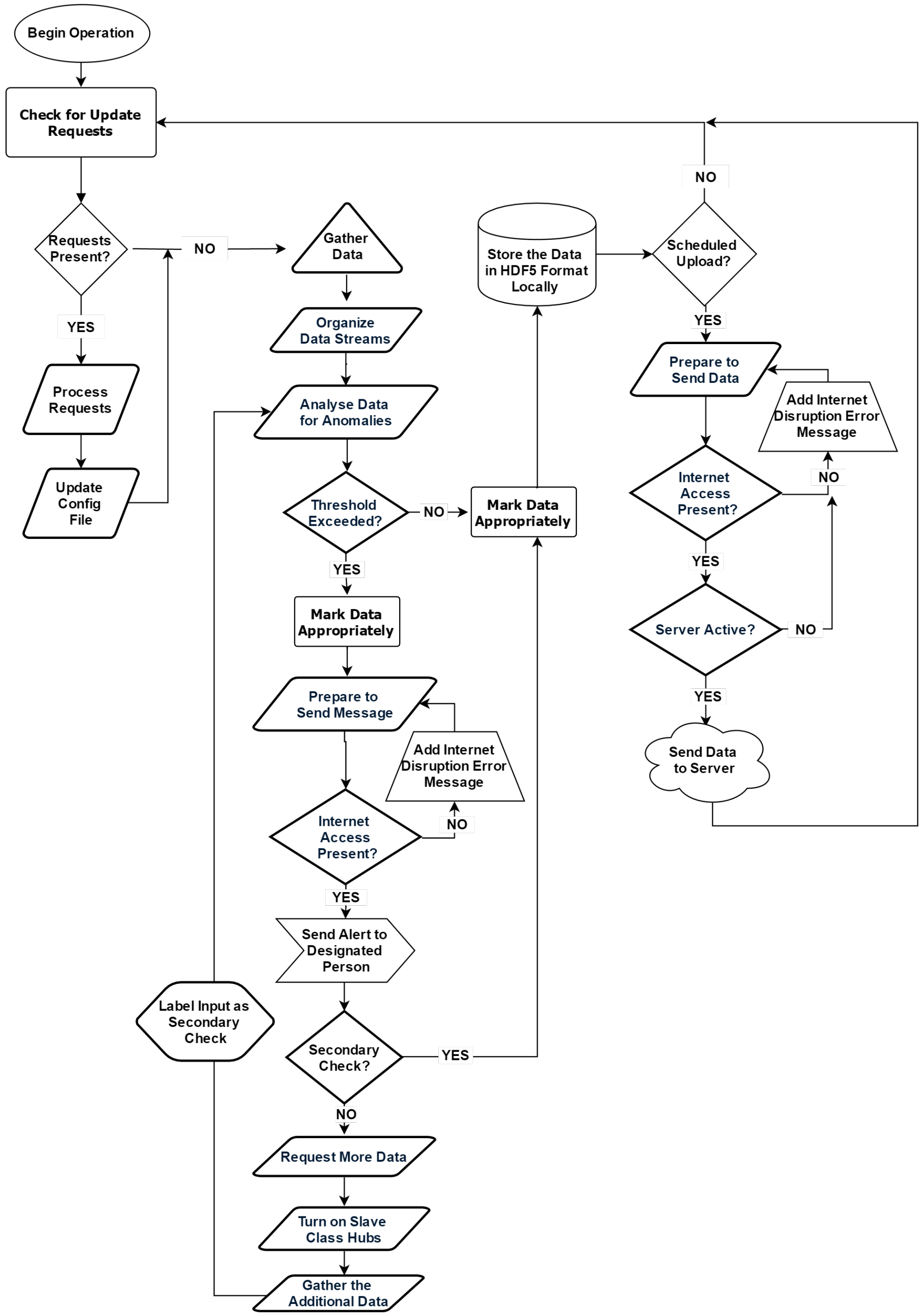
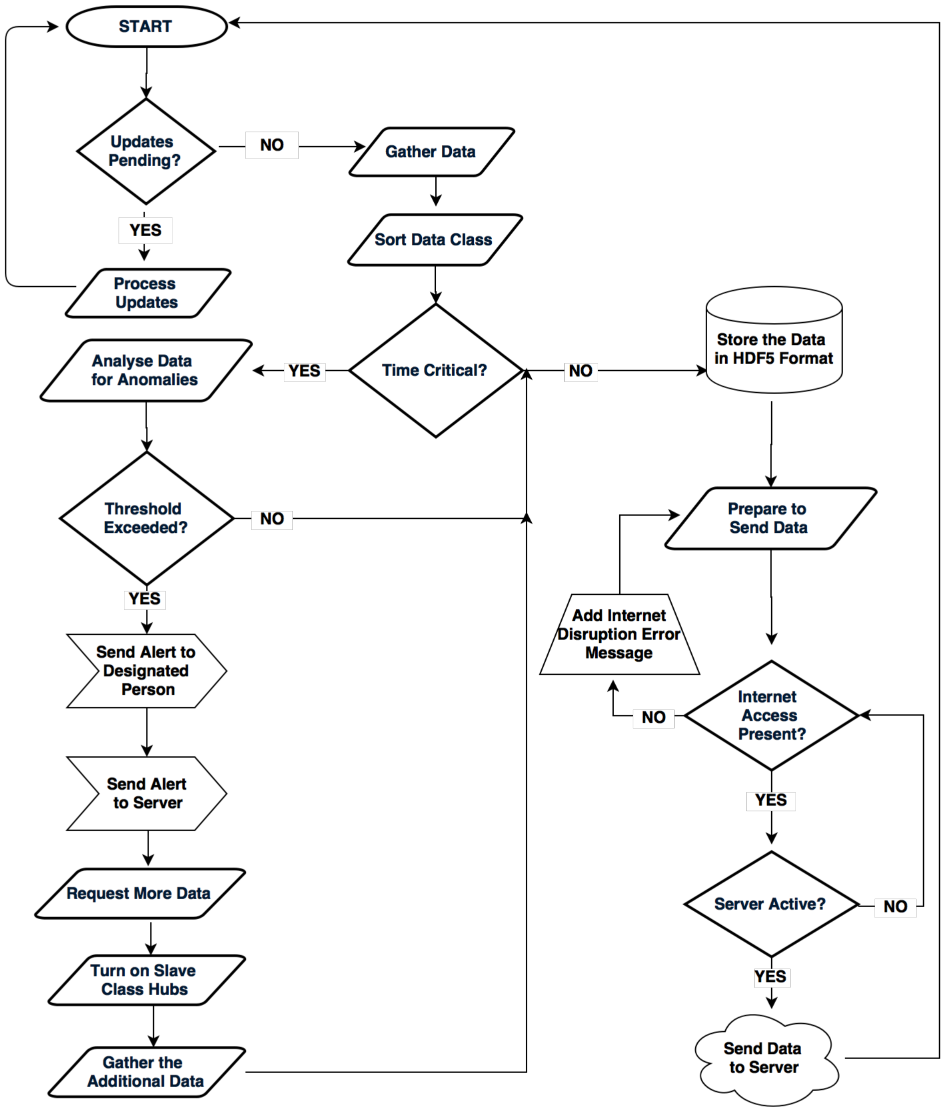


Figure 3.6: A diagram displaying the high level algorithm of semi-autonomous mode of operation



need another layer of abstraction or a system to connect them into one single system. Another issue that was present is the limitation imposed by the Category 5 Ethernet Cable (CAT5) and CAT5E cables, which are used by the PoE configuration. The aforementioned limitation is the distance that the cable can cover, which is 90m according to the TIA-EIA-568-A standard [33], before signal degradation requires for the connection to be terminated or re-amplified. This issue becomes more apparent when using DC current, which is the method of power transmission in PoE systems. The reason it becomes apparent is due to power losses as described in their white paper by CommScope[4], Petrosky et.al [25] as well as by numerous setup manuals for PoE installations and patents such as the patent described in [26] as well as the Cisco Manuals [28] and [29] to name a few.

By unifying the main control under one server, and allowing it to communicate with any of the switches or hubs from a single location, the scalability becomes a trivial matter. This is linked to the fact that an element can be introduced to the system without excessive reworking of the code or a large scale rearrangement of the system. As a result, ease of introduction, as well as easily established hierarchical relationship between the elements, allows the system to grow and expand on any level, assuming the central processing entity possesses the resources needed to handle the expanded data inflow. This matter is in turn resolved by the utilization of cloud computing, which allows for prompt and simple resource allocation. This greatly simplifies the task of expansion, as long as the operator planned ahead, in order to eliminate potential down time of the system. As the new elements can be introduced without interrupting the operation of the remainder of the system.

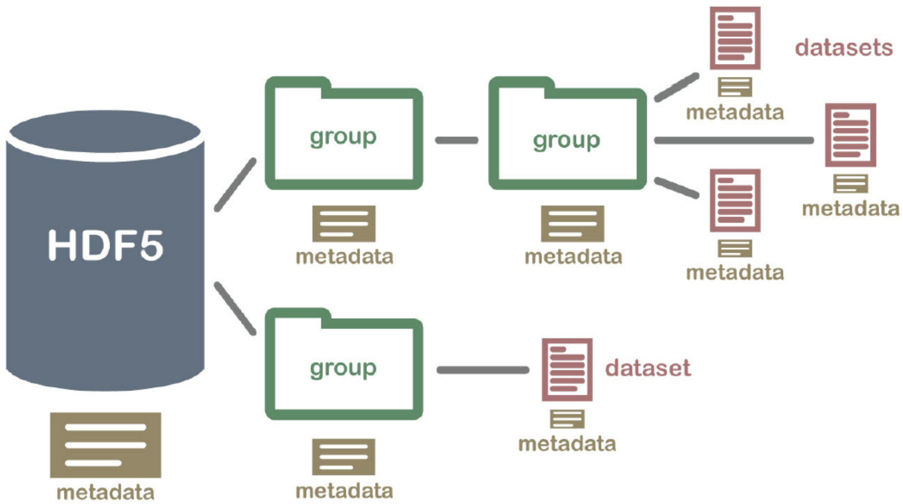
To facilitate direct access to the switch and its settings, it must first be configured to allow SSH [28] [29]. Once that is done the switch can be accessed remotely by any machine that has access to the internet and the terminal, through the server virtual machine, assuming it has the access privileges for that level. To further automate the process, a macro can be saved for each switch if necessary by writing a simple script which will pull the necessary credentials from a configuration file containing the information on the switches, hubs and sensors [**?**][34]. Due to the need for security, as the switches may potentially have a lot of control within a building, measures need to be taken to secure the information and access to the hardware as well as control instances. However due to the focus of this work being the improvement of scalability, adaptability and responsiveness this aspect will be further discussed in Chapter 5 as future work and recommendations. One of the primary uses for automatic SSH scripts in this work is to issue commands to the switch remotely and autonomously, such as in the case where additional slave hubs need to be activated. For this purpose Tool Command Language (TCL), is used on the switch side. It is invoked by the server once an SSH connection is established and can be configured to perform a variety of tasks in addition to controlling the power supply. A CRON job[14] is used to perform a series of regular, periodic checks on the non-time-critical data, the server also constantly listens to any alerts from the hubs. The script can be activated in two cases, either an alert about an anomaly is received directly from the master hub, which processed the critical data locally, or derived from processing the information on the server. Each hub, if requesting additional data, will transmit its ID along with the request, which will be used to look up the required credentials. Once found, a script will be run to issue the command to the corresponding switch and slave hubs. When the data is processed on the server the IP address of the hub that detected the anomaly will be used to find the required credentials.

3.7 Data Processing and Presentation

As mentioned previously, data is formatted and stored in HDF5 format. The use of this format allows to assign meta data to the gathered measurements, which simplifies and speeds up the process of parsing the data for further processing[35]. As well as cross platform import, in cases where different parties want to use different software to analyze the data locally[36]. In addition once the data has been arranged correctly, the new transition times of file transfer become shorter, as the data files are a lot more compact and light weight. In order to achieve this, the raw data is first gathered from the individual sensors. A preassigned name is used to label the source of information, the name is obtained from the JSON configu-

[14]**CRON** is a software utility that acts as a time-based job scheduler in Unix-like computer operating systems

Figure 3.7: A diagram displaying the structure of the HDF5 data format [3]



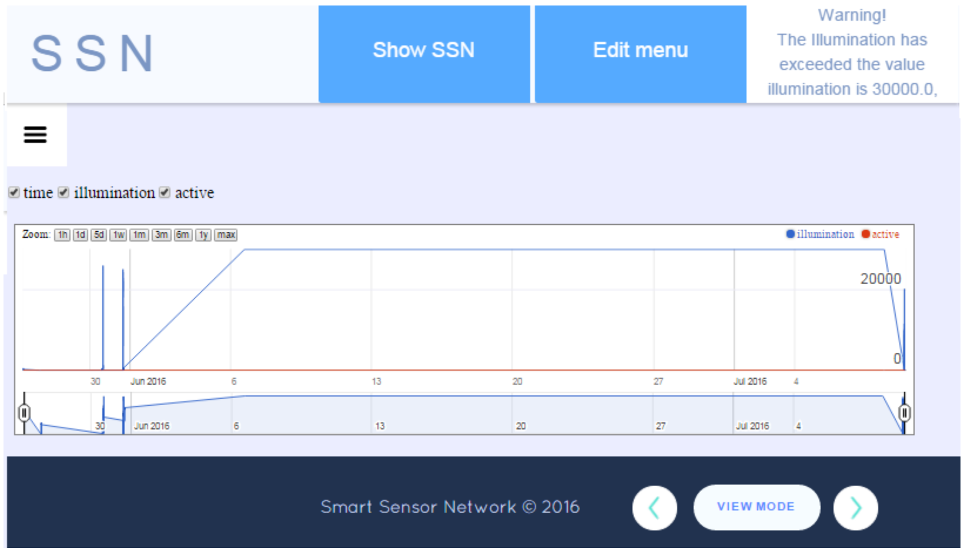
ration file. In addition the type of data that is being gathered, as well as its safety constraints are also used as the meta data for the measurement. Once formatted correctly, the newly arranged data is stored in its own sub directory of the local database, as seen in Figure 3.7, along with the other measurements. On the server side the data is the extracted, parsed into arrays and displayed to the user.

3.8 The Hierarchy

Hierarchy is a very important aspect in a project that emphasizes scalability such as the one discussed in this work. As the two are directly linked and it relates to:

- Access levels - who should have access to which devices and services?
- Control level - how many units do you want to issue a command to or which area do you wish to cover?
- Error location - What is the location of the device that is causing errors (in case of malfunction)? What is connected to it? What in that chain of com-

Figure 3.8: A diagram displaying the graph view representation of the data



mand could be causing it?

As such a few approaches are used to introduce the concept of hierarchy into the system. The database format that is used to send, store and process the data is Hierarchical Data Format - version 5 (HDF5), which is great for fast transfer of data due to its ability to keep the file sizes low and well organized. The latter point also makes it great for processing, as accessing any piece of information from the database or parsing it becomes simple when using HDF5.

Hierarchy is also employed in other aspects of the design, as can be seen in Figure 3.10, the hierarchy determines the amount of control a unit has. For example a server has full authority, while a single salve hub has none. This ”chain of command” makes propagation of commands rather simple and clean.

For the comfort of user experience the web interface also allows the operator to see the hierarchy of the system visually as can be seen in Figure 3.11. As shown by the figure the separation can be done on any level that is required by the application. Due to the fact that in this particular case the testing was done internationally by deploying the system in Canada and in Europe there are two main branches of information. Additional levels can be used for commercial applications, such as

Figure 3.9: A diagram displaying the table view representation of the data

time	illumination	active
2016-07-09 12:18:11.	0.0	1
2016-07-09 12:18:09.	0.0	1
2016-07-09 12:18:07.	0.0	1
2016-07-09 12:18:04.	0.0	1
2016-07-09 12:18:02.	0.0	1
2016-07-09 12:17:49.	0.0	1
2016-07-09 12:17:47.	0.0	1
2016-07-09 12:17:45.	0.0	1
2016-07-09 12:17:37.	0.0	1
2016-07-09 12:17:30.	0.0	1
2016-07-09 12:17:22.	0.0	1
2016-07-09 12:17:18.	0.0	1
2016-07-09 12:17:15.	0.0	1
2016-07-09 12:17:13.	0.0	1
2016-07-09 12:17:06.	0.0	1
2016-07-09 12:16:56.	0.0	1

floor numbers, buildings, city blocks etc.

Figure 3.10: A high-level diagram outlining the hierarchy of the system

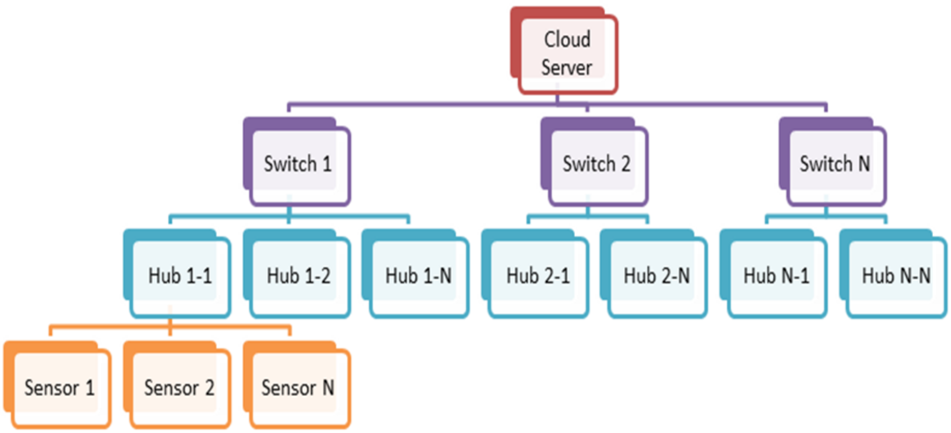
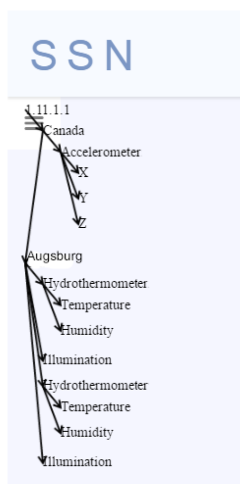


Figure 3.11: A screenshot of the web view of the hierarchy of the system



Chapter 4

Testing and Applications

A number of components in this system were tested and the testing was separated into two general stages: a local testing stage and a global testing stage. The local testing examined the system performance and response to such factors as:

- Hardware variation
- Module integration and alteration
- Power transmission and control using PoE
- Localized data processing
- Local control sequence testing
- Slave hub activation and anomaly response

Whereas the globalized part of the testing focused more on the aspects of the scalability of the systems. Such as:

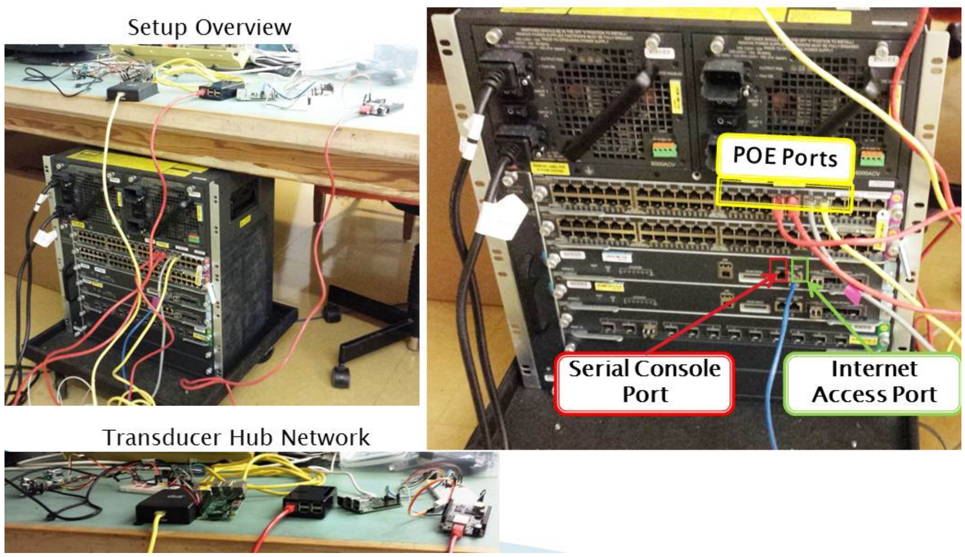
- Deploying the system in a cross-continental setting IN Canada and Germany
- Testing the response speed of the system
- Data transfer speed testing
- Configuration modification and propagation response testing
- Multi-modular deployment test

The experimental setup used for local testing can be seen in Figure 4.1. The performed tests examined the closed loop response of the system in a zero-access setting, without access to the internet and the main server. In this setting the data was to be processed locally and the commands were issued by the master class hubs. This test looked at a number of aspects of the system. It examined how localized data processing performed, whether commands issued directly by the master class hubs would be accepted and followed by the network switch, and whether the power supply acts in accordance to the issued commands. These are all important aspects of a system that uses PoE, and ongoing testing was conducted during the development of the system to ensure correct operation. As a test of adaptability and modularity of the system, a different microcontroller board was used, in particular the BeagleBone development board[1]. After a simple configuration of the output pins, no other changes were necessary. Furthermore, the reconfiguration can be done remotely, using either the online editor interface, or a direct back end connection using SSH. The sending, receiving and processing schemes as well as the control modes all performed as expected without any errors, and did not require any excessive remodeling or re-coding.

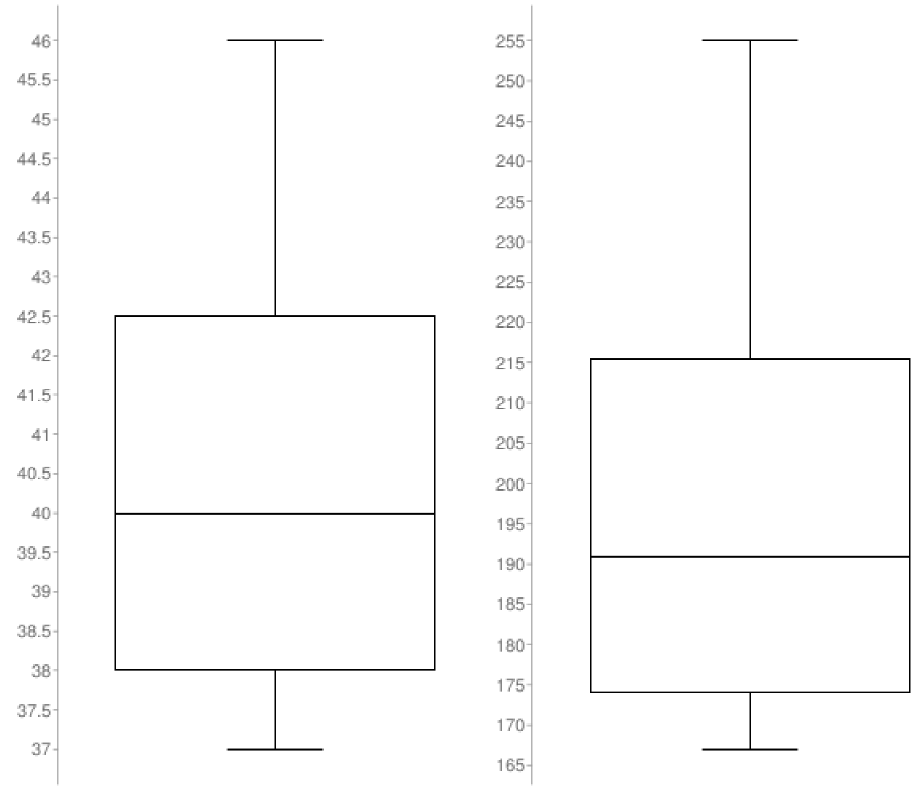
Once the system proved to be operational, and successfully passed all of the local tests a number of aspects necessary for scalability were tested. These tests focus more on the software portion of the prototype, and were meant to access the feasibility of use of such a system rather than claiming certain benchmarks. The first aspect to undergo examination was the aspect of connection speeds between the machine and the server. Due to the fact that the primary limitations of the connection speed are the Internet Service Provider imposed limitations and the number of hops the transmission has to go through, which usually correlates with the geographic location of the communicating parties and is decided by the Internet Service Provider (ISP). The average time for the European setup was 42ms while the average Canadian time was 190ms. Due to the server being located in Germany these results are to be expected. For more optimal transfer times a number of modifications can be implemented, among which are the use of a cloud computing service whose servers are geographically close to the site of operations or personal servers which would be housed directly at the project cite, or in relatively close vicinity.

[1] http://beagleboard.org/bone

Figure 4.1: Experimental setup for local testing



The general setup for testing the scalability was done by connecting two setups from two different continents. The first setup was done in Vancouver, Canada, North America while the second one was done in Augsburg, Germany, Europe. Each system consisted of at least one master class and at least one slave class module. Each of the modules had a number of sensors connected to them. While some of the sensors that were connected varied between the two setups, an illumination sensor, a vibration sensor, temperature and humidity sensor as well as power sensor were used in different configurations on all of the MCBs. After the initial setup was complete, to test operational stability and performance, a latency test was performed. The test was carried out over the course of 15 days, at approximately the same local time, the peak demand period of afternoon was used as studies have shown that around 14:00 is when internet usage is the heaviest according to [37]. During the measurement, three samples were taken and recorded. The resulting latency statistics can be found in Figure 4.2 where sub-figure 4.2a displays the



(a) Augsburg **(b)** Vancouver

Figure 4.2: Latency test results for the two cities where the tests were conducted.

latency measurement results obtained in Augsburg and sub-figure 4.2b shows the measurement results in Vancouver. Over the course of testing there has been minimal package loss (11%), even in cases of long transition routes, such as in case of the Vancouver measurement. During the testing there were no errors detected on the server side with neither the data nor the request handling. The system showed no issues handling the requests from both of the setups and operating with both sets of MCBs, even in cases where their requests arrived nearly simultaneously.

A full scale test of the system will be conducted as part of the Green Campus

Project in Hoschule Augsburg, Germany. There the system will be used to monitor the energy consumption and generation around a section of the new building that has been built,as well as its structural integrity and interior climate. For this purpose vibration sensors, LDT0-028K was chosen as the sensor of choice as it has shown to perform well for the purposes of earthquake detection and building vibration [38] as well as parasitic power generation as discussed by P. Glynne-Jones in their work outlined in [39]. Large area of effect strain sensors, such as a long-gage fiber optic sensor, which have been shown to perform well as civil structure monitoring device as explained in their overview paper by H.N. Li et. al. [40] as well as in the paper by S. Li et. al. who discussed the feasibility of a distributed system of such sensors in [41]. As well as temperature and humidity sensors, such as the DHT22 and power measuring sensors will be used. The alpha prototype has tested the correctness of operation of the vibration sensor, temperature and humidity sensor and the power measurement sensor by assembling a circuit for each and calibrating their outputs. Once the obtained readings matched the expected values, which were obtained by performing a control measurement by industry manufactured devices, the operation level was deemed feasible. During the testing the feasibility of use of DHT11 temperature and humidity sensor was tested, however high levels of inaccuracy in readings (over 15% deviation) and other hardware related issues made us consider the sensor as unreliable for the purposes of testing.

Chapter 5

Conclusions and Future Work

In conclusion, the developed architecture has improved a number of aspects, as originally outlined by the goals. The scalability was improved by simplifying the modification of the element base, by incorporating JSON configuration files and their respective control schemes. It was also improved through the incorporation of IoT communication principles, as well the use of PoE and cloud computing, which allows for a more accessible and efficient expansion and larger areas of deployment. The design also showed improvements compared to similar systems in adjustability, as all of the elements can be altered, added or removed without interrupting the operation of the system as a whole. Furthermore any software changes automatically propagate through the affected areas, which also allows for a relatively high degree of plug-and-play ability. Lastly the design introduces improvement to aspects of reliability and adjustability, by incorporating algorithms for self checking, a semi-centralized architecture, and absence of dependence on any single component within the system. Furthermore the use of cloud computing allows for redundancy in terms of processing power and robustness, as well as secure remote access and control capability. Whereas the hierarchy that is entwined into the organizational structure of the design, allows for a high degree of controlability, due to easy localization, isolation, and targeted addressing. Nevertheless, in order to adapt the design for commercial use there are still a number of improvements that could be made that would benefit the performance. The major aspect to be considered is the enhancement of security and, in case of in-house servers, more elaborate data routing and handling. While a certain level of security it is obtained by securing the server the system can benefit from employing Software Defined Networking, SDN, to improve internal routing, which would in turn improve and optimize data handling as well as packet control to ensure no tampering or unauthorized access occurs. In their work Kapil B. [42] outline how the employment of Software Defined Networking (SDN)s can benefit a network, and quantify the performance benefits of such a transition form the traditional hardware defined networks. Furthermore the use of SDNs could also further improve the speed as well as the type of data that can be available, for example a video feed of a room or area of interest, which could in turn be used for an array of purposes, such as motion detection, as well as temperature reading of a room if an Infra Red camera is used as one example.

Another improvement that can be made to the system is further segmentation and automation of access to different levels of the system. This includes adding the capability to select an ”area of effect” which can be defined based on the application, but in case of buildings it would define what portion of the building is affected by a certain action.

While there has been work done on Real Time Ethernet (RTE) systems, they primarily focused on data transfers within small time windows, primarily in an industrial setting for motion control [43] [44]. With this being the case the question of power, and PoE has not been considered in that context. A possible course of development for this concept can be its integration with RTE for more industrial applications and even more time efficient operation, however that was outside of the scope of this work and has therefore not been investigated in detail.

As a result, a system that matches the set goals has been developed. The system is capable of operating by utilizing Power over Ethernet (PoE), which drastically simplifies installation procedures as well as reduces power consumption costs. By placing the central client of the system in the cloud, an online processing service, a high degree of scalability has been achieved, as well as a high degree of controllability, which is are very important aspects of the Internet of Things (IoT), field. The system can be further improved to increase the security of the transmitted data, as well as to reduce the size of the MCBs used. Nevertheless as it currently stands it can greatly benefit smart buildings, large buildings that wish to reduce the amount of power consumed by the Heating, Ventilation and Air Conditioning system (HVAC), as well as buildings that apply new building technology and materials and need to analyze their performance. The power necessary for actuation can be provided by using local energy storage units in form of some battery that can be trickle charged using PoE. As can be expected the system function well with the big data approach and has been tested to be operational in a trans-continental set up, where the gathered data was accessible from any machine that had access to the internet, and the processing and transfer of the data was completed in a timely and efficient manner, due to high processing capabilities of cloud computing machines and small size of the files that used HDF5 database format.

Bibliography

[1] I. Lobachev and E. Cretu, "Smart sensor network for smart buildings," in *Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2016 IEEE 7th Annual*, pp. 1–7, IEEE, 2016. → pages iv

[2] IEB Media GbR, "Cat5e cable wiring schemes and the 568a and 568b wiring standards," 2010. → pages viii, 14

[3] F. G. Osorio, M. Xinran, Y. Liu, P. Lusina, and E. Cretu, "Sensor network using power-over-ethernet," in *Computing and Communication (IEMCON), 2015 International Conference and Workshop on*, pp. 1–7, IEEE, 2015. → pages viii, 32

[4] COMMSCOPE INC, *Laying the groundwork for a new level of Power over Ethernet*, 2015. → pages ix, 2, 30, 50, 51

[5] R. Goldstein and D. Neuman, "Mega-buildings: Benefits and opportunities of renewal and reusethe essential role of existing buildings in a carbon neutral world," in *Proceedings of the American Institute of Architects National Convention and Design Exposition held in Miami, Florida, USA, 10-12 June*, 2010. → pages 1

[6] United Nations, Department of Economic and Social Affairs, Population Division, *World Urbanization Prospects: The 2014 Revision, Highlights (ST/ESA/SER.A/352)*, 2014. → pages 1

[7] J. A. Rice, K. Mechitov, S.-H. Sim, T. Nagayama, S. Jang, R. Kim, B. F. Spencer Jr, G. Agha, and Y. Fujino, "Flexible smart sensor framework for autonomous structural health monitoring," *Smart structures and Systems*, vol. 6, no. 5-6, pp. 423–438, 2010. → pages 5

[8] O. Kasten and M. Langheinrich, "First experiences with bluetooth in the smart-its distributed sensor network," in *Workshop on Ubiquitous Computing and Communications, PACT*, vol. 1, 2001. → pages 6

[9] P. Rawat, K. D. Singh, H. Chaouchi, and J. M. Bonnin, “Wireless sensor networks: a survey on recent developments and potential synergies,” *The Journal of supercomputing*, vol. 68, no. 1, pp. 1–48, 2014. → pages 6

[10] S. Xie and Y. Wang, “Construction of tree network with limited delivery latency in homogeneous wireless sensor networks,” *Wireless personal communications*, vol. 78, no. 1, pp. 231–246, 2014. → pages 6

[11] M. E. Keskin, İ. K. Altınel, N. Aras, and C. Ersoy, “Wireless sensor network lifetime maximization by optimal sensor deployment, activity scheduling, data routing and sink mobility,” *Ad Hoc Networks*, vol. 17, pp. 18–36, 2014. → pages 6

[12] M. Magno, T. Polonelli, L. Benini, and E. Popovici, “A low cost, highly scalable wireless sensor network solution to achieve smart led light control for green buildings,” *IEEE Sensors Journal*, vol. 15, no. 5, pp. 2963–2973, 2015. → pages 6

[13] O. Kreibich, J. Neuzil, and R. Smid, “Quality-based multiple-sensor fusion in an industrial wireless sensor network for mcm,” *IEEE Transactions on Industrial Electronics*, vol. 61, no. 9, pp. 4903–4911, 2014. → pages 6

[14] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, and X. Li, “Does wireless sensor network scale? a measurement study on greenorbs,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 10, pp. 1983–1993, 2013. → pages 6

[15] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, “A wireless sensor network for structural monitoring,” in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 13–24, ACM, 2004. → pages 6

[16] K. Chintalapudi, T. Fu, J. Paek, N. Kothari, S. Rangwala, J. Caffrey, R. Govindan, E. Johnson, and S. Masri, “Monitoring civil structures with a wireless sensor network,” *Internet Computing, IEEE*, vol. 10, no. 2, pp. 26–34, 2006. → pages 6

[17] W. Townsend, “The barretthand grasper-programmably flexible part handling and assembly,” *Industrial Robot: an international journal*, vol. 27, no. 3, pp. 181–188, 2000. → pages 7, 27

[18] H. J. Visser and R. J. Vullers, “Rf energy harvesting and transport for wireless sensor network applications: Principles and requirements,” *Proceedings of the IEEE*, vol. 101, no. 6, pp. 1410–1423, 2013. → pages 7

[19] A. Pantelopoulos and N. G. Bourbakis, “A survey on wearable sensor-based systems for health monitoring and prognosis,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 1–12, 2010. → pages 8

[20] C.-Y. Chong and S. P. Kumar, “Sensor networks: evolution, opportunities, and challenges,” *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247–1256, 2003. → pages 8

[21] Cisco Systems INC, *Cisco Power over Ethernet (PoE) Power Requirements*, 2015. → pages 13

[22] G. Mendelson, “All you need to know about power over ethernet (poe) and the ieee 802.3 af standard,” *Internet Citation,[Online] Jun*, 2004. → pages 13

[23] A. A. Mangiaracina and L. O. Nielson, “Lighting utilizing power over the ethernet,” Apr. 23 2008. US Patent App. 12/108,074. → pages 13

[24] J. Johnston, J. Counsell, G. Banks, and M. J. Stewart, “Beyond power over ethernet: the development of digital energy networks for buildings,” in *CIBSE Technical Symposium 2012-Buildings Systems and Services for the 21st Century*, pp. Session–5, 2012. → pages 13

[25] J. Petroski, “Power over ethernet thermal analysis with an engineering mechanics approach,” in *2016 32nd Thermal Measurement, Modeling & Management Symposium (SEMI-THERM)*, pp. 50–56, IEEE, 2016. → pages 14, 30

[26] B. Buckmeier, E. Edralin, and J. M. Hess, “Power over ethernet for 10gbase-t ethernet,” Aug. 20 2015. US Patent App. 14/831,102. → pages 14, 30

[27] B. Nordman and K. Christensen, “Local power distribution with nanogrids,” in *Green Computing Conference (IGCC), 2013 International*, pp. 1–8, IEEE, 2013. → pages 14

[28] D. Barnes and B. Sakandar, *Cisco LAN switching fundamentals*. Cisco Press, 2004. → pages 16, 30

[29] Cisco INC, *Cisco Catalyst 4500E Supervisor Engine 8-E Configuration Guide (Wireless), Cisco IOS XE Release 3.7E*. Cisco Press, 2 ed., 2014. → pages 17, 30

[30] Cisco Systems INC, *Cisco Catalyst UPOE Power Splitter*, 2015. → pages 17, 19

[31] National Instruments INC, "The NI TDMS File Format," tech. rep., National Instruments INC, 09 2015. → pages 20

[32] P. Marian, "Digital-output relative humidity humidity and temperature sensor/module am2303," 2016. Note: original document from Aosong(Guangzhou) Electronics Co.,Ltd. → pages 26

[33] Panduit Corp., *The Evolution of Copper Cabling Systems from Cat5 to Cat5e to Cat6*. 2016. → pages 30

[34] OpenBSD and u. p. y. IETF Group, url=http://www.openssh.com/manual.html, "Openssh manual." → pages 30

[35] T. Krijnen and J. Beetz, "Efficient binary serialization of ifc models using hdf5," 2016. → pages 31

[36] M. Folk, G. Heber, Q. Koziol, E. Pourmal, and D. Robinson, "An overview of the hdf5 technology suite and its applications," in *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*, pp. 36–47, ACM, 2011. → pages 31

[37] N. Brownlee and K. C. Claffy, "Understanding internet traffic streams: dragonflies and tortoises," *IEEE Communications magazine*, vol. 40, no. 10, pp. 110–117, 2002. → pages 38

[38] F. R. d. C. Alves, *Low-cost vibration sensors: tendencies and applications in condition monitoring of machines and structures*. PhD thesis, INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA, 2015. → pages 40

[39] P. Glynne-Jones, M. Tudor, S. Beeby, and N. White, "An electromagnetic, vibration-powered generator for intelligent sensor systems," *Sensors and Actuators A: Physical*, vol. 110, no. 13, pp. 344 – 349, 2004. Selected Papers from Eurosensors {XVI} Prague, Czech Republic. → pages 40

[40] H.-N. Li, D.-S. Li, and G.-B. Song, "Recent applications of fiber optic sensors to health monitoring in civil engineering," *Engineering structures*, vol. 26, no. 11, pp. 1647–1657, 2004. → pages 40

[41] S. Li and Z. Wu, "Development of distributed long-gage fiber optic sensing system for structural health monitoring," *Structural Health Monitoring*, vol. 6, no. 2, pp. 133–143, 2007. → pages 40

[42] K. Bakshi, “Considerations for software defined networking (sdn): approaches and use cases,” in *Aerospace Conference, 2013 IEEE*, pp. 1–9, IEEE, 2013. → pages 42

[43] M. Felser, “Real-time ethernet - industry prospective,” *Proceedings of the IEEE*, vol. 93, pp. 1118–1129, June 2005. → pages 42

[44] S. Vitturi, L. Peretti, L. Seno, M. Zigliotto, and C. Zunino, “Real-time ethernet networks for motion control,” *Computer Standards and Interfaces*, vol. 33, no. 5, pp. 465 – 476, 2011. → pages 42

Appendix A

Supporting Materials

A.1 Power over Ethernet - Operation Specifics

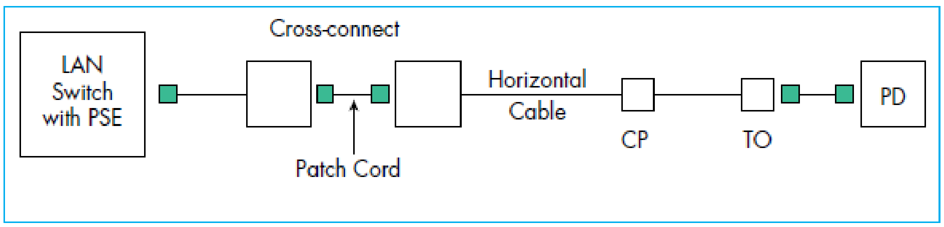
One of the main principles incorporated into the design on PoE is to ensure that the Ethernet data along with the power transmitted do not interfere with one another during operations. This has allowed PoE to be used for a variety of applications where the two streams must function simultaneously side by side. General components of a PoE setup are:

- **PSE** - power sourcing equipment.Usually this function is fulfilled by a LAN switch with power supplying capabilities.
- **Cabling** - The Ethernet cabling that is used to connect the elements in the system.
- **CP** - Consolidation Point. Is a zone within a building up to which the cabling that was installed from the equipment room can be fixed.
- **TO** - Telecommunications Outlet. Can be a port for a telecommunication connection, such as a phone jack, or an Ethernet port in a wall.
- **PD** - Powered Device. This is the recipient of the supplied power.

As can be seen in Figure A.1 as well as Figure A.2 there are two generic configurations that can be utilized when designing a PoE system layout.

The configuration outlined in Figure A.1 shows an approach where the power supply is centralized. As such the patch cord could be used to complete certain connections, however the PoE power distribution would be handled by a single unit. This brings on a challenge of positioning the switch in order to optimize its reach.

Figure A.1: Power supplied form LAN equipment with end-span power sourcing equipment[4]



This challenge, along with the issue of excessive power dissipation is addressed by the configuration concept shown in Figure A.2. A mid-span device that can supply power, without disrupting the data transmission. For this reason they are tiled power injectors, and can also act as a standalone source for design scenarios where that is required.

Figure A.2: Power supplied form mid-span power sourcing equipment[4]

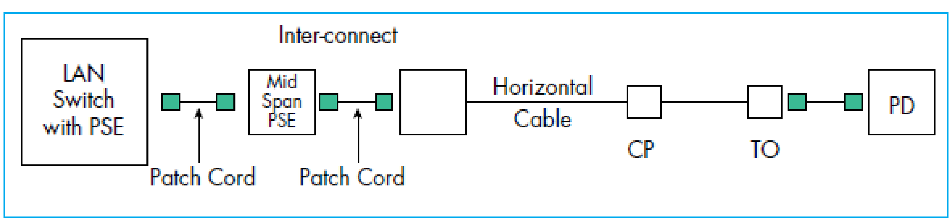


Figure A.3 looks more closely at the two alternatives of two-wire set-up for PoE systems. The IEEE 802.3bt standard, allows the use of both pairs of cables, totaling four cables, to transmit the power, systems that utilize this approach refer to it as PoE Plus. This results in an umber of benefits, including more efficient power transfer due to smaller power dissipation. However the most important benefit, is that by using both of the cables once can use both of the alternatives together, effectively doubling the amount of transmitted power.

Figure A.3: Alternatives A and B for the design for use with end-span power sourcing equipment according to IEEE 802.3at-2009[4]