
INTERNSHIP REPORT

for Summer at
Center for Smart Systems (CS²)
Singapore University of Technology and Design

Don Kurian Dennis
Department of Computer Science and Engineering
Indian Institute of Technology Patna

Submitted in partial fulfillment of requirements of CS399 - Summer Training/Internship

August, 2016

Contents

1	Introduction	3
2	Major Projects	4
2.1	Universal IoT Gateway	4
2.1.1	Background	4
2.1.2	Project Proposal	5
2.1.3	Implementation Details, Challenges Faced and Results	7
2.2	AirQ - Air Quality Measurement	8
2.2.1	Background	8
2.2.2	Project Proposal	9
2.2.3	Implementation Details, Challenges Faced and Results	9
2.3	BATMAN and Mesh Routing	11
2.3.1	Background	11
2.3.2	Project Proposal	11
2.3.3	Implementation Details, Challenges Faced and Results	11
3	Minor Projects	13
3.1	Ontology Based Project Management	13
3.1.1	Background	13
3.1.2	Project Proposal	13
3.1.3	Implementation Details, Challenges Faced and Results	14
3.2	Setup and Configuring Lab Servers	14
3.2.1	Background	14
3.2.2	Implementation Details, Challenges Faced and Results	15
4	Conclusion	16
5	Acknowledgment	17
6	Internship Feedback form	18

1 INTRODUCTION

I underwent my summer internship at the Center for Smart Systems, Singapore University of Technology and Design(SUTD). I had along with me, Aditya Gupta and Rishabh Goel from the Department of CSE, IIT Patna at CS^2 for the summer. We had Dr. Vishram Mishra and Dr. Hock Lim Beng as our mentors at SUTD.

The Center For Smart Systems (CS^2), primarily focuses on developing and integrating IoT devices to form usable, scalable products for masses and our projects were driven towards achieving that goal. The three of us worked on various projects with varying levels of contributions on each. I personally worked exclusively on the projects titled '**Universal IoT Gateway**' and '**AirQ**', and collaborated with the others on '**Ontology Driven Object Management**' and '**Setting-up and Configuring the Lab Servers**'. I was additionally asked to investigate the possibility of extending the 'Universal IoT Gateway' to support **Disaster Resilient Communication Pathways** using mesh routing.

The Center for Smart Systems (CS^2) is a collaboration between MIT, SUTD and ST Engineering. The lab was previously located at NTU Singapore under the name of Intellisys, which shifted to SUTD around March 2016. There are various projects running in the lab simultaneously funded by various institutions. The projects we worked on were primarily in collaboration with ST Engineering.

Individual project details, accomplishments and results are presented in this document with the amount of disclosure allowed by SUTD. The project details are separated into two sections, namely *Major Projects* and *Minor Projects*. The Major Project section contains projects where I was the primary contributor while Minor Projects detail other projects I contributed to. Each project description has my contributions and project results mentioned at the end. My internship was from 1, May 2016 till late July 2016 and it was completed successfully and to the satisfaction of everyone as evident from the internship feedback form.

My internship at CS^2 was in collaboration with ST Engineering and commercial interests are vested in the products we developed. To that end we cannot divulge details of some of the projects. Description and details hence can be vague or seemingly incomplete in some sections.

2 MAJOR PROJECTS

The internship at CS^2 involved a number of major and minor projects. Some spanned the entire period of internship while others were smaller and were completed within shorter spans of time. The major projects I was part of are mentioned in the following section.

2.1 Universal IoT Gateway

2.1.1 Background

Internet of Things (IoT) products are often an integration of various sub-modules, which among themselves are individual products. For example, a cloud controlled home automation module might have a Bluetooth 4.0 based sensor for controlling wireless speakers and remotes, a Zigbee based module for communicating with temperature control modules, a WiFi module for communication with the Internet and so forth. One of the major hurdles developers in IoT face is the integration and interfacing of such submodules. There are numerous standards, technologies and protocols existing in the market and it becomes cumbersome when designing new products composed of components which work on different protocols. Often, it becomes the responsibility of the developing team to write new and fresh translations and gateways between such protocols. The code has to be secure, stable and reliable - especially so since most IoT devices work remotely and are targeted at users with little or no debugging knowledge.

Even among a deployment of different purpose IoT devices, it is desirable that the devices be able to communicate with each other, share data and be aware of each other in the network in a simple uniform language (protocol) so that it becomes easier to integrate more devices, remove more devices and use multiple devices in conjunction to complete certain tasks effectively.

This is where a gateway comes into play. Gateways support multiple communication protocols and are able to talk to various protocols and devices. Further, the gateway exposes a uniform, device independent protocol for the programmer/user making it easier to communicate with the supported devices. Various gateways from different manufacturers are available in the market but they all suffer from some common drawbacks.

- Gateways are often created with a specific task in mind, for example a home automation

gateway will support sensors and actuators that will help automate tasks around the house.

- These gateways do not provide any way to extend the support for a new device or even use multiple devices together - at the same time.
- Gateways cannot communicate between themselves often, even when the gateways are for the same purpose.
- Gateways from different manufacturers, even when for the same use case, cannot communicate among themselves.
- There is no uniform standard for gateway design or working.

In the current scenario, for someone to have a gateway support a new device, usually, he will have to contact the manufacturer, have them write a new/updated firmware for the gateway supporting the new device and supply the firmware update through an OTA download or otherwise. Things could be even worse - the manufacturer could refuse to support the new device or the gateway could be such that a firmware update might not be feasible or possible. Not an ideal scenario.

2.1.2 Project Proposal

This is where the proposed **Universal IoT Gateway (UG)** can ease the work flow. The UG uses ideas of virtualization and an ontology based metadata management to be able to support communications between any device irrespective of the protocol. Further, the ontology based structure of the gateway makes it easier to add and remove devices from the network in a plug and play mode. The UG is also able to communicate across networks and be aware of nearby networks. A common, natural language like command interface is made available to the end user by the UG, independent of the device protocol or the device being utilized for some particular task. BATMAN mesh routing and disaster resilient non-mainframe networks are additional side effects of the nature and structure of UG.

Ontology is meta-information about name, type and relationship between entities. Device ontology can provide information regarding frame type, meaning of corresponding indices in frame, sensors in the device, sensor type, sensor data function etc. The ontology data forms a hierarchical system (tree) of information from the device level to the sensor level. Generic information is supported at the topmost layers which gets specific as we move down the tree. The device manufacturer is responsible for publishing its device ontology onto a publisher site.

The ontology based management uses the PHY Protocol + MAC ID to key into ontology

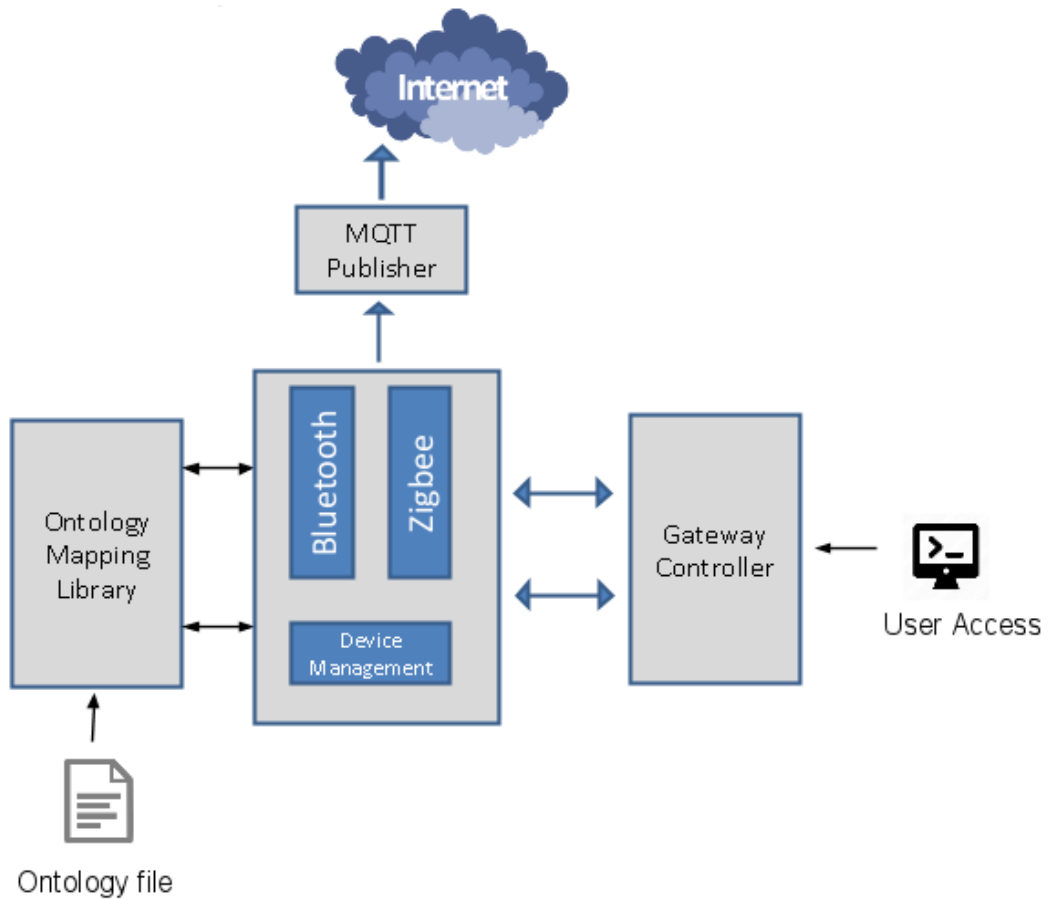


Figure 1: IOT Gateway Modules

mapping. That is, each device ontology is uniquely identified by the device protocol type and the MAC ID of the device. The ontology is publicly available at the ontology website for anyone to download and use.

The UG utilizes multi-radio interface for communicating and scanning/searching for new devices in its network. The Underlying network/node discovery protocol is followed in a similar way by almost all devices in PHY layer for example, any BLE device will have the standard node discovery protocol on it but the application/sensors on it are unknown.

The Universal Gateway performs node discovery and gets the MAC address of the discovered node. It identifies new nodes and tries to see if it contains the ontology information for these in its local database. If it does have the ontology information, it adds the new device to its networks of available devices and can begin to accept user instructions for it immediately.

On the other hand, if the ontology information is not found locally, the UG searches for the ontology in the publisher website using the devices MAC ID and PHY layer interface as key and downloads the ontology information. It proceeds to connect to the device and

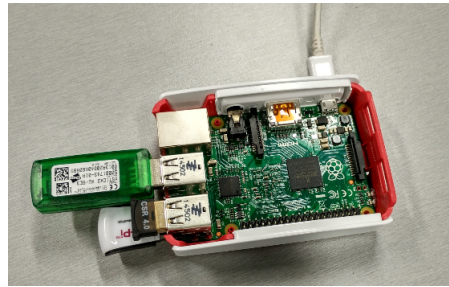


Figure 2: IOT Gateway Based on Raspberry Pi 2

include it in the discovered devices network.

This ontology based structure of the Universal IoT Gateway (UG) removes the drawbacks conventional device specific gateways have. The universal IoT gateway,

- supports dynamic, real-time plug and play of devices
- enables devices to communicate between themselves
- facilitates the sharing of data between multiple devices
- helps devices be aware of each other in the network
- helps the user identify appropriate devices from the network if multiple devices can accomplish some given task
- can communicate with other universal gateways to form a network
- is device and manufacture independent and works across all devices and manufacturers without needing any change at the device/manufacture end. The only requirement is the ontology information.

2.1.3 Implementation Details, Challenges Faced and Results

At the end of the May, 2016, we were able to demonstrate a working prototype of a Universal IoT Gateway based on an Raspberry Pi 2. The prototype UG supported the Zigbee, Bluetooth (with BLE) and WiFi stack and the demo included a network of devices including TI Sensor Tags V2, XBee light/temperature/humidity sensor, MIO Global Fuse fit band, Phillips Hue Lights and other Raspberry Pi Networks. The presentation was a huge success with our audience, which included high ranking officers from ST Engineering (who is funding the IoT Development) extremely pleased with our progress.

Along with other contribution in integration and infrastructure creation, I was able to rewrite and significantly modify the BlueZ driver for Linux/Bluetooth stack to support BLE and our specific use case. The BLE/Linux driver structure was a huge challenge in integra-

tion as it supported an event based approach while our gateway required a server-client architecture. A significant change in the BlueZ code was required.

My modified bluetooth server, named *gatt-server*, after the BLE gatt profile was built on top of the BlueZ library, modified to our needs. The server architecture consisted of a main server thread, which listened for instructions on a POSIX message queue and queued these instructions into a internal queue. Another thread in the server, serviced the instructions from the queue. Some of the instructions, such as node discovery and fetching the ontology data ran irrespective of the availability of the ontology information. For other instruction, a separate process, *gatt-client* was spawned for the execution. The *gatt-server gatt-client* IPC was accomplished via another POSIX message queue interface.

I implemented the *gatt-client* using the BlueZ API in C. Unfortunately, bluez API for C is not that well documented and it runs on a *glib* based interface. I faced many challenges converting and accommodating the event based *glib* interface into our server-client architecture.

After the implementation work was done, I subjected my *gatt-server gatt-client* implementation to a rigorous testing routine with custom test cases, dummy devices, devices with wrong information, devices that fail to respond etc and the server functioned as expected and was found to be extremely robust and stable.

Finally, after integrating the module into the gateway, I also designed the prototype protocol and packet structure for communication among different modules within the UG, with the MQTT servers and the ontology database.

2.2 AirQ - Air Quality Measurement

2.2.1 Background

Singapore, being one of more developed countries in south east Asia and the world invests heavily in environment friendly research. One of the projects in this context is the National Weather Monitoring project under National Environmental Agency (NEA) Singapore.

Under the project, the NEA has deployed 22 odd weather monitoring equipment across Singapore which helps in keeping a continuous track of the weather data across the area. The sensors help monitor different parameters like PM10, PM2.5, NOX, SOX, temperature, humidity, volatile organic compounds etc. Though the project as a whole is a success, there is scope for a lot more improvement:

- The devices currently being used are really expensive, each costing around 5000 S\$ (3680 USD)

- Individual devices are responsible for the monitoring of a large area. Currently there is one device for every 32.7 km^2 . This means that any local noise/pollutant, like someone smoking near a PPM detector, pollutes the data for the entire area.
- Also, data is usually extrapolated in areas further away from these stations- therefore accuracy in question.
- The maintenance of the devices presents another cost deficit.
- Better sensors are now available but the closed, non-modular structure of the original deployed devices make it difficult for easy upgrade and maintenance.

2.2.2 Project Proposal

Center for Smart Systems is trying to device a new air quality monitoring sensor - AirQ, which could be cost effective, easy to update, reliable and stable in remote use. The aim was to bring the cost of the devices so low that it becomes feasible to deploy a larger number of these across Singapore and help increase the resolution of the current system.

- The AirQ is designed to be robust, stable and extremely cost effective.
- It is aimed to be able to work in extremely harsh conditions and still stay online.
- Even in the unlikely situation of AirQ going off grid, the device is aimed to be able to work offline for long durations, log the data collected and send it to the central server once connection is restored. To compete
- The sensors and accessories integrated into AirQ are designed in a modular fashion so that upgrades can be easily done.
- OTA software upgrade is also an aim for AirQ.
- It should be solar powered with low power consumption and with little or no maintenance overhead.

2.2.3 Implementation Details, Challenges Faced and Results

I was responsible for creating the first prototype of AirQ. A LinkitOne board from MeadiaTech was used for AirQ with particulate matter, temperature and humidity sensors interfaced to it. The combined cost of the board, the sensors and the battery pack was still well under S\$150 (110 USD).

After I interfaced the sensors to the AirQ, I performed some local tests and then integrated the GSM and WiFi modules for connectivity to the internet. The GSM stack presented some unique problems in terms of their implementation of the GSM *bind()* and *connect()* functions.

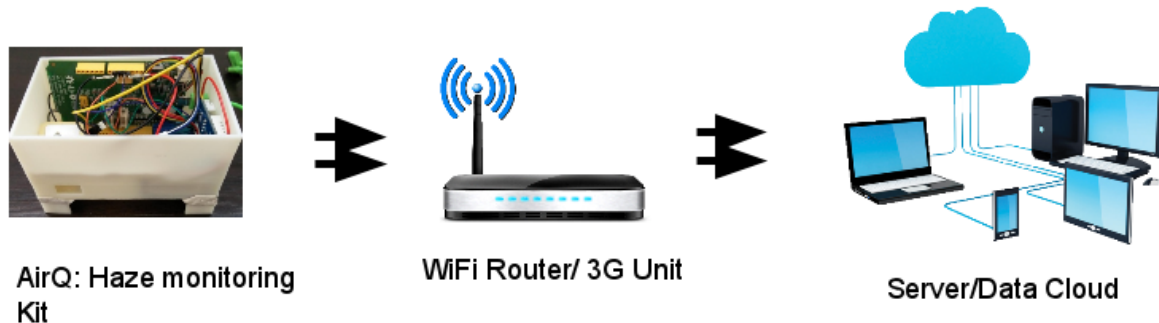


Figure 3: AirQ Architecture

With the help of the support team from the MediaTech forums and I was able to work around these problems and finalize the implementation of AirQ.

The server client communication was facilitated using the MQTT/Mosquito protocol so to keep the data transmission footprint to a minimum. I had to optimize the connections, the data writes and the number of open ports to minimize the data transfer footprint and battery usage. The AirQ prototypes used a battery pack attached to a solo power charger for charging and also supported local logging for situations with a disconnected network coverage. Finally, I boxed AirQ in a structure we 3D printed at SUTD and deployed for field testing.

The field testing of AirQ lasted for a couple of weeks with five prototype devices deployed. For comparison purposes, we also deployed commercial AirBox devices, which are much more expensive than AirQ. The data we received from both the devices showed that AirQ in its current state is very reliable in terms of performance, up time and data accuracy.

The AirQ was presented at CS²'s booth at the Maker Fair '16, Singapore edition. It was also presented at National Environmental Agency (NEA) and further testing is currently underway before it is deployed as a packed product.

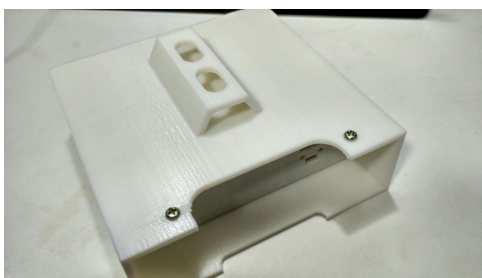


Figure 4: AirQ 3D Printed Enclosure

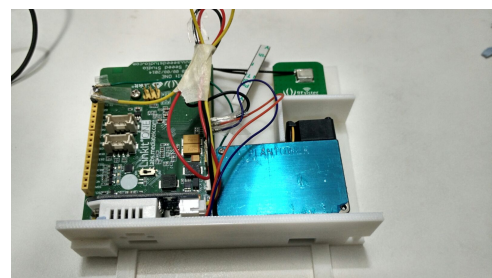


Figure 5: AirQ

2.3 BATMAN and Mesh Routing

2.3.1 Background

One of the advantages of the Universal IoT Gateway is its awareness of adjacent networks. As a side effect, the UG lends itself to be configured as a network of interconnected nodes. Since our current UGs are raspberry Pis, the power of the RPi allows the implementation of some interesting functionalities. One such functionality we tried to explore is that of mesh routing through UG nodes.

2.3.2 Project Proposal

The idea is to use the stability and network connectivity provided by the gateways to form a routing architecture that can get an Internet packet to the Internet when connectivity of individual nodes are down but there exist some nodes which are still connected. This is especially useful post a natural hazard when Internet connectivity at some nodes might be lost. The routing protocol will be able to get Internet destined packets from devices connected to the gateway to other gateways till we find one gateway that has a live connection to the Internet. Further, the local network among the gateways themselves can be used for communication like telephony or messaging.

2.3.3 Implementation Details, Challenges Faced and Results

I was responsible for investigating the possibility and feasibility of mesh routing through the gateway prototypes we had at CS². I studied and used the BATMAN protocol for the same. The BATMAN protocol, designed for mesh routing, once installed on the mesh nodes, creates a virtual interface called *bat0* that can be used for routing. For our specific use case, we also needed to create a bridge device between the *bat0* interface and the interface with Internet so that the routed packets can be transferred to the Internet.

At the end of a brief exploration;

- I was able to setup a small network of three Raspberry Pi nodes connected in a mesh.
- The nodes were able to communicate with each other using the BATMAN protocol.
- Internal routing was successfully tested between nodes not within the radio range of each other.
- A SIP server was setup within the mesh network to provide internal telephony and IM services.

- The mesh network was able to route Internet destined packets from devices not connected to the Internet through the ones that are.
- The mesh network was able to adapt to dynamic Internet connectivity changes and route the packets accordingly.

3 MINOR PROJECTS

Along with the two major projects, I was also part of other project teams. The details of these projects and my contribution to them are mentioned in this section.

3.1 Ontology Based Project Management

3.1.1 Background

Labs and institutions like CS^2 run a large number of IoT projects on their servers at the same time. All these projects use different set of devices and sensors deployed across various locations. For each of these project, there is a project specific server, database and other architecture components. This kind of a work flow is not exactly ideal due to various reasons. Firstly, the sensors within the various projects share some common properties - a project on indoor comfort measurement and weather data analysis both use devices which collect temperature data. Storing the data in different databases is bound to introduce a level of redundancy that can be removed. Further, such sensors that do common work across projects should be able to be shared across projects as well. This will significantly reduce the investment in project setup and maintenance. Also, data from a previously running project should be easily accessible to a new project with same sensors. This is not a feature provided by the existing work flow.

Another aspect of the said workflow is the effort required to setup the infrastructure for each and every project. Setting up the database and servers are mechanical work, which can possibly be automated resulting in a significant cost cutting. Added advantage is the smaller infrastructure footprint which will now exist and the smaller investment required in maintaining this infrastructure.

3.1.2 Project Proposal

Considering all this, the CS^2 started a project titled 'Ontology-based IoT Backend' which aims to use ontology information to automate the process of creating the IoT backed database and allow sharing and reuse of data across projects at the same time removing data redundancy. Additional aims of the project involve:

- High scalability (reuse of knowledge).

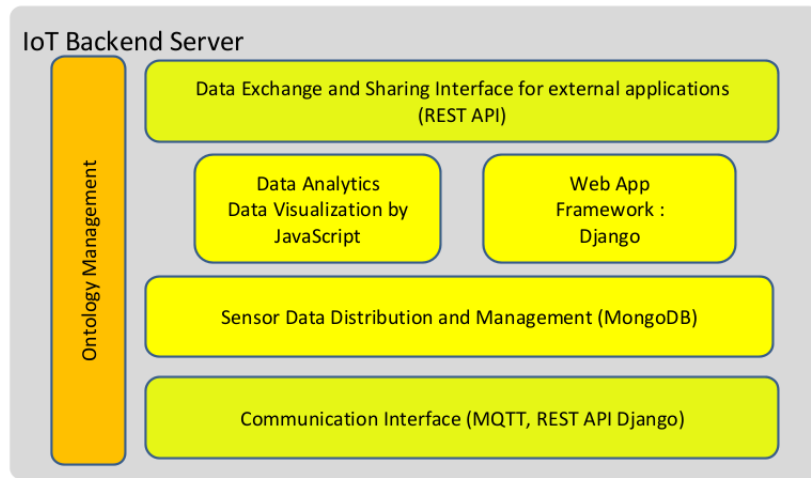


Figure 6: Backend Architecture

- Low deployment cost.
- Quick Development.
- Easy integration (RESTful API).
- Robust big data processing (cloud computing).

3.1.3 Implementation Details, Challenges Faced and Results

My contribution was in helping the team finalize the architecture for the project, the tools to be used and the modular structure to build on. We surveyed various combinations of tools and technologies we could use and finalized that a Django backend powered by a mongoDB database will be the best. We also decided to provide a RESTful API for managing the ontology database.

At the end of the project, we were able to demonstrate a working version of the Ontology Based Backend with support for automatically adding new projects, sharing data between projects, identifying and removing redundancies etc.

3.2 Setup and Configuring Lab Servers

3.2.1 Background

The Center for Smart Systems is a newly established lab. It started operations around March 2016. The infrastructure for CS^2 was not completely set up and configured and one of our tasks at SUTD was to setup and configure the servers and other hardware at CS^2 .

3.2.2 Implementation Details, Challenges Faced and Results

SUTD follows a three layer network structure. The outer most layer being the internet, followed by a Demilitarized Zone(DMZ) for the data center and at the lowest level, the local network for internal machines. Even within the local network, each subnet is actually NATed behind a common IP address. The three layer structure allows only traffic in one direction. No machine in the internet can access machines in the DMZ (SUTD uses a dynamically changing DNS) and no machines in DMZ has access to the LAN. Further STUD uses an Enterprise level security system on all its internet links. All this presented a lot of challenges while configuring and installing our lab servers. We had to learn, read up and understand a lot about how the network architecture is configured, work with the SUTD IT department on the formalities of our requirement and configure our servers accordingly. Due to the stringent security measures imposed on networking at SUTD, specifically on machines at the DMZ, access to the DMZ and machine in the DMZ were sparse. We had to make sure that our machines deployed within the DMZ could restart, update and maintain themselves. Further, we also had to make sure all the machines could survive and repair themselves after an kind of internal or external driven crash.

We were successfully able to configure two CS^2 servers at SUTD data center, for hosting our publicly visible projects. All the projects which were stalled due to non availability of servers are now running on these machines. We were also able to configure three servers within our lab space for local usage and testing. These servers are not publicly accessible and are predominantly used for internal testing of new projects before their deployment.

4 CONCLUSION

As stated in the individual project sections, all the projects we underwent were completed to the satisfaction of our director Dr. Lim Hock Beng and others at SUTD. The projects and the results we obtained in the short span of time were impressive to the audience we presented it to as well.

The projects are currently being continued by others at (CS^2) and we will be helping them remotely as much as we can. Project specific conclusions, contributions and results are mentioned in the project description section itself.

5 ACKNOWLEDGMENT

My internship at SUTD has been really rewarding. The opportunity gave me a platform to work with experts in their field and helped me get hands on experience in solving real life problems. I was also able to meet some great people at SUTD from various fields of experience and expertise, hear them and discuss with them and learn a lot. This internship has been a great experience for my personal and professional life.

I would like to thank Dr. Jimson Mathew, our TPC coordinator and the whole TPC team for helping me in securing this internship. I thank Dr. Lim Hock Beng for accepting us as interns at his lab at SUTD and for the guidance during the three month period. Dr. Vishram Mishra helped us a lot in the entire process of getting to and settling into Singapore and the lab. He was also a great mentor and helped me immensely with various problems I faced during the internship. I would like to thank him for all the help.

Siyuan Wang and Rui Yi were with us as interns themselves during our stay at Singapore and I would like to thank them for their collaboration on various projects. Finally, I would also like to thank SUTD and IIT Patna for facilitating this internship opportunity.

CERTIFICATE OF STUDENT'S PRACTICAL TRAINING
(ISSUED BY MENTOR/SUPERVISING AUTHORITY OF ORGANISATION)

1. Name & Address of Organization : Center for Smart Systems
Singapore University of Technology and Design
8 Somapah Rd, Singapore 487372
2. Place of Training : Singapore
3. Name of the Student & Roll No. : Don Kuan Dennis (1301CS17)
4. Date of Commencement of Training : 4th May - 2016
5. Date of Completion of Training : 29th July - 2016
6. Actual Number of Working Days Attended : 75
7. Days of Leave availed, if any : -

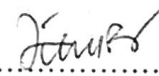
8. Brief Details of Training (Nature of Training / Projects taken up, if any)

Developing a universal IoT Gateway, developing and deploying air quality sensors,
developing an ontology based adaptive (wasp) database to cook autonom projects,
deploying and configuring lab servers in demilitarized zone.

9. Overall Performance of the Student during Training : Very Good ☒
- (Please tick the appropriate box)
- : Good ☐
- : Satisfactory ☐
- : Unsatisfactory ☐

10. Remarks on the Conduct of the Student, Punctuality, Interest etc. :

Don has contributed significantly to the development of a prototype
IoT gateway and related sensor integration tasks. He is hardworking,
a fast learner and works well in a team.

Signature of the Authorized Officer : 

Name & Designation of the Officer (with seal) : Prof Hock Beng Lim

Place : SUTD, Singapore

Date : 27 July 2016

Note: The certificate of training may please be completed in duplicate. One copy may be retained by the organization for their record and one copy may please be sent directly to "Professor In Charge, Training & Placement, Indian Institute of Technology Patna, Bihta, Patna -801103 (Bihar)". Or scanned copy to tpc@iitp.ac.in