**ACKNOWLEDGEMENT**

My initial thanks goes to the Industrial Tanning Division, University of Moratuwa, it’s director Mr. Ananda Gamage and the support stuff for arranging the industrial tanning programme which is a very valuable opportunity to engage and interact with the industry.

Also my heartiest gratitude does to Dr. (Mrs.) Dileeka Dias for being the initial coordinator for internship opportunity at the Singapore Management University. I also would like to thank the Electronic and Telecommunication Engineering department’s training coordinator Dr. Ruwan Udayanga Weerasuriya for guiding me at the most required moments during the training period. My sincere gratitude goes the department stuff, senior students and colleagues who though, guided and help me prior and during the training process to extract the best from the industrial training.

I would also like to extend my sincere gratitude to Associate Professor TAN Hwee Pink, Academic Director, SMU-TCS iCity Lab and Miss. Elina YU Jia, Assistant Director, SMU-TCS iCity Lab for selecting me as an intern at SMU-TCS iCity Lab. Also I am graceful to my supervisors Dr. Alvin VALERA, Research Fellow at SMU-TCS iCity Lab, Dr. TAN Hwee Xian, Research Scientist at SMU-TCS iCity Lab and Mr. Pius LEE Senior Research Engineer at SMU-TCS iCity Lab for guiding, mentoring and advising me though out the entire internship period. Also I would like to thank my fellow researches Miss. Xiao Ping TOH, Miss. TAN Lee Buay and Mrs. Cheryl KOH for the help and guidance given at the induction process. Finally I would like to thank Mr. Liming BAI and Mr. LA THANH Tam, Research Engineers at SMU-TCS iCity Lab for helping me with the technical aspects during the training.

I would like to thank the Singapore Management University staff for facilitating me though out the process. My sincere gratitude goes to the Singapore National Design Centre and iDA Lab staff for facilitating me with tools, equipment and fabrication facilities during the internship period.

Also my special thanks goes to Mr. Isuru Seneviratne and Mr. Udaya Roshan for coordinating the accommodation facilities in Singapore.

**Chapter 1**

**Introduction to the Training Establishment**

**1.1 Singapore Management University**

Singapore Management University (SMU) is a world class university established in 2000 with the mission of generating leading-edge research with a global impact. Singapore Management University is well reputed for its state of the art small size class rooms and interactive, collaborative and project-based teaching approach. The university is a home for about 9300 undergraduate, postgraduate, executive and professional, full and part-time students. University is comprised of six schools in the areas of Accounting, Business, Economics, Information Systems, Law and Social Sciences.



Figure 1

**1.2 TATA Consultancy Service**

TATA Consultancy Service (TCS) is established as a division TATA Sons Limited in 1968. The company incorporated as a separate entity in the year of 1995 while being a part of the India’s largest conglomerate, the TATA group.



Figure 2

TATA Consultancy Services is an IT services and consultation company which provides business solutions to organizations and business entities. The Global Network Delivery Model™ is one of the recognized industry bench mark developed by TCS in delivering IT services to its customers.

TATA Consultancy Services has more than 238,500 of the world’s best IT consultants over 45 countries. They handle more than 4500 IT contracts and have over 1500 more polled. The company operates over 13 European countries making it the Europeans #1 IT service provider in customer satisfaction. Currently the company has a combined value of 40 Trillion Euros making it one of the biggest IT service providers in the Europe Union.

**1.3 SMU-TCS iCity Lab**

**1.3.1 Brief History**

SMU-TCS iCity Labs was initiated as a joint venture of TATA Consultancy Services and Singapore Management University. TATA Consultancy Services had invested S$ 6 million at iCity Lab for research in cloud based IT solutions.



Figure 3

The idea of a new research lab to build cloud based service platforms came in the play in the mid of the year 2011. Due to the heavy investment by TCS the inauguration of the lab came in the August same year. After doing a feasibility study and a survey the first project idea was started in the month of November and deployed in March 2012. One of the key ideologies came at the days of inauguration by the initiators was to take a people interaction approach for the research and development work rather than in-house development. This idea was one of the key reason for the success of the lab and the one of the key features which distinguish the iCity lab from the rest of the research labs in the field.

**1.3.2 Research forces and functional goals**

SMU-TCS iCity lab was initiating as a research laboratory which will help to flourish the Singapore Government’s vision of ‘Smart Nation’. The labs main forces was to develop intelligent systems and platforms for Smart City ideas. Labs research forces could be categorized into four main areas.

1. Aging

2. Healthcare and Chronic Disease

3. Education and life-long learning

4. Employability

**1.3.3 Projects and present performance**

Neighbourhood for Active Living (NodAL) was a pilot project done with the collaboration of Eastern Health Alliance (EHA) to setup a community based care management system for the Singapore East. The main forces of the system was to take care of the patients (mainly elderly) after discharging from the hospitals. The care intervention was done though regular phone calls and visits to the houses by the volunteer groups. One of the key goals of the project was to reduce the frequent hospitalization due to lack of after hospitalization care. This actually transformed the manual paper driven healthcare to a collaborative iCare system.

iCity Visual Analytics Toolset is another project developed by iCity labs that could be used to business and social data analytics. These tools could be used to analyse data forecast future predictions and trends in the field of businesses social behavioural patterns.

SHINESeniors is the current ongoing project of iCity lab which is a Remote Elder Care Platform. The operational perspective of the project is to collect sensor data from the PIR sensors and door sensors for the analysis of human behaviour.

In the currently deployment the lab had implemented four PIR sensors and detect the motion of the elderly with in the house. There is a door contact sensor to detect the outing of the elderly from the house. Also a medication box with a sensor is installed to collect medication adherence data of the elderly. The bed sensor detects the sleeping patters which could be used to detect anomalies in the normal sleeping patters. Besides the sensors a Panic Button was provided to elderly to alter any panic attacks.



Figure 4

In the current modality a ‘No activity alert’ is triggered by analysing the PIR sensor data which is a suspicious call for the cases like elderly falling inside the house or fainting. Currently the distress alters are monitored by the iCity Lab staff and the Good Life, a volunteering association for elder care. In the future the operational power will be fully on the hand of the Good Life association.

Medication box is another interesting sub-project which is a part of the SHINESeniors project. The medication adherence data is quite useful for doctors in analysing the complications. Currently the elderly are alerted via a phone call if they miss any dosages. The lab is in the process of developing an automated process which could be monitored remotely by a care giver and make alters via a smart phone.

By the date the lab had complied 12 publications including two white papers. Due to the immense social impact the lab was able to secure a government grant of $ ……. For research and development.

**1.3.4 Organizational Structure**

Labs organizational structure is mainly into three main streams. They are Operations, Research and Development. Also the lab is monitored and by a two separate Steering Committee and an Advisory Committee.

**Steering Committee**

**Advisory Committee**

**Operations**

Senior Director

**Research**

Senior Research Fellow

**Development**

Chief Architect

Admin Officials

Student Interns

Research Scientists

Research Fellows

Research Interns

System Analyst

Developers

Interns

Figure 5

**1.3.5 Strengths and Opportunities**

Singapore government and health care associations are very much interested in the concept of ‘Ageing in Place’ which is a more sustainable solution for the growing population of elderly. Like most of the countries Singapore also faces the problem of imbalance population pyramid which will eventually lead to a huge elderly population in another 20-25 years. So the Singapore government had noticed this problem and currently promoting the ‘Ageing in Place’ concept. This is one of the main reasons that the iCity Lab started working on the SHINESeniors project as well. The government support is also one of the main strengths that the iCity Labs has. Also it is also a great opportunity for the lab to deploy a project like SHINESeniors.

TATA Consultancy Services is one of the biggest IT solution provider in the world. So in the initial dates the TATA consultation officers were helping the lab to start the projects and backing up the initialization process.

Besides TCS, iCity Lab is also collaborating with A-Star Research Company which the most reputed research institute in Singapore. With that collaboration, a considerable amount of knowledge, consultancy and help is flowing towards the lab. This is also a good strength that is available to the iCity Lab.

‘Lion Befrienders’ is another welfare organization which take care of lonely elderly. They also have a similar sensor system implemented at 500 elderly homes. Even though that they have the sensors, the organizations doesn’t do any ‘Behaviour Analytics’ with the data rather just alerting any when the elderly face any accident inside the house. So recently the organization made a partnership with the iCity Lab and start sharing the data. With that huge amount of data the lab was able to do more statistical analytics from the data obtained. This is a big opportunity for the lab in-terms of data.

**1.3.6 Weaknesses and threats**

Currently the sensor network was developed from off the shelf components. Also the hardware maintenance is done by a separate company. Because of this reason is researches find very hard to find some of the sensitive information. The problem is that the hardware company doesn’t disclose some of their sensitive information which is vital in analysing the data. Also too much noisy data is also another problem. Since the system is maintained by a third party the compensations are very small and the product vender doesn’t agree with some of the faults that the lab find from deep data analytics. This is one of the weaknesses that the lab currently has.

Since most of the people who are in the lab are from the research field the industrial exposure is very much less. So the industrial practices are not well practiced inside the lab which is another weakness of the lab.

Due to the aging population there are many organizations which are looking in the same type of solution which will be a threat in the future at the point of commercialization. Also a community project like SHINESeniors always should be backed by the government for its sustainable deployment. This also another threat that the lab is facing currently.

Besides that the iCity Lab can’t be the care givers or the responsible personalities to look after the elderly in the long run. Even in the current system the ‘Good Life’ association also play the care givers role. So this dependency on a third party is also another threat that the lab faces in the phase of implementing the systems.

**1.3.7. Profitability and usefulness**

SHINESeniors Project is mainly a community project which doesn’t expect much income from it. Even though that after the system is developed to a commercializable level it could be sold for a very high price due to the amount of analytics which was carried out in the developing process. The fact is that in Singapore iCity is the only research lab which does data analytics from the real life data for a considerable number of uses for a longer period. So these research data could be much more valuable than anything.

Ageing in Population is one of the upcoming problems in the entire world. Due to this fact in the near future the requirement of an elder care system will arise from the entire world. So there will be a very high demand for the eldercare system from countries outside Singapore as well. Selling the system to countries outside Singapore also will gain much profit for the lab.

The eldercare system developed by SHINESeniors project is very much useful for the elderly who live alone. In Singapore the birth rate is very due to the high expenses with in the country. So new families are much reluctant to have children. Due to this trend the country will end up with a considerable number of couple who doesn’t have children to take care of them. Also due to the busyness in the life the children will not have time to take care of their parents on site. So the developed system is a very good useful solution to both of the cases mentioned above.

**1.3.8. Suggestions and constrains to improve**

One of the key suggestions is to be independent in terms of development. Specifically the lab should maintain the hardware sensors so that the hardware too could be improved with time. Developing the hardware in-house is one of the solutions and the other is to use off the shelf sensors and assemble them in-house. Either way the lab will be having the required knowledge about the sensor specifications and will have the power to make adjustments when needed. This will solve most of the noise problems and data ambiguity that the lab is facing currently.

To deploy that idea the lab will need more experts on hardware as well. So with that expertise most of the currently facing problems could be solved. Also the lab should be equipped with industrial experts since they are dealing with a considerable number of hardware and software which should be more robust and accurate. Since the applications are more critical and mistakes could cost lives. This fact is very much important in the fully automated operational stage.

**Chapter 2**

**Training Experiences**

**2.0 Introduction**

SMU-TCS iCity Lab is a research lab consisting of a small number of highly talented researches doing research in human behaviour analytics. To do the research on the matter the lab had implanted the sensor system (Figure 1.xx) in 50 elderly homes in Marine Parade area. As a research intern I was able to engage in few research projects and numerous product developments. One of the initial day projects was to develop a low power RF transmitter and receiver pair using an Arduino Pro Mini, Raspberry Pi and two RF24 radio transceiver pairs. After that project I was assigned with a new project to retrieve data from SAMSUNG Gear S2. The device is loaded with the latest Tizen Wearable operating system which is a new system in the market. So the community support and other developer support is very minimal in which I had to work hard to get the things working.

All the other work was mainly based on the medication box which was deployed at the elderly’s homes. One of the initial work was on system integration and testing of the next version medication box. For that I was given some pre-written binaries to be integrated into systemd and test for the proper functionality. System testing consumed a lot of my time since some of the problems arouses after a prolonged operation of the system. Recreating the same problem was very much time consuming and tedious. For the given system the developers were not able to come up with a stable and device independent 3G connection driver which led to another task for me. Due to this problem I had write a stable 3G connection driver for the 3G dongles which run in Raspberry Pi. After that implementation I carried out some rigorous testing and over few weeks and finally an on-site testing which came out with very good results.

In the mean time I also wrote a Bash Script which could be used to make multiple copies of the SD card by installing the applications and making all the custom configurations automatically. The implemented system took a newer approach which is about 80-90% more efficient that the simple ISO image copying techniques. Few design iterations too was carried out with more added features like copying some of the files directly from the server as well. Also I made a simple power circuit using N-MOSFETs which is to be used with the Raspberry Pi to solve a voltage level and current limiting problem to light a set of LEDs. Also wrote a very stable and fool proof algorithm to be work with the ‘reed switch’ in the same design iteration.

On my last days I worked on a Bluetooth Beacon project which is to be used with the Raspberry Pi. It was another implementation for the older MedBox with more added features and some more functionality and data that could be used for Human Behaviour Analytics.

In the days of my internship I was able to participate to a seminar, a conference, a field visit and two exhibitions. My initial participation was to a Singapore-Sweden excellence seminar in ‘Ageing Societies and Innovative Ideas’. Then I participated to an IoT get-together which was hosted by SMU-TCS iCity Lab. After that I visited the Fabrication Laboratory (FabLab) of Singapore University of Technology and Design (SUTD). On the same day I also visited the Computer Science Department of Nanyang Technological University (NTU). The next outbound event was the Singapore’s ‘Future of Us’ exhibition hosted by the Singapore government showcasing the future government plans to improve the quality of living through technological innovation. I participated to the event as an exhibitor. After that I was lucky enough to participate to the IoT Asia 2016 conference as a student log-taker and as an exhibitor at the SMU-TCS iCity Lab’s stall. The experience gained at these events was immense and the exposure is much valuable.

**2.1 RF24 Transmitter Project**

**2.1.1 Project description**

The project was to develop a low power data transfer system for the Medication box (MedBox) as an improvement for the currently prevailing XBEE module. One of the main problem of current XBEE is the high power consumption during data transmission. This make the 2000mAh batteries drain within 1-2 months of operation. So as a solution I was requested to design an ultra-low power system to transmit data from the medication box to the Raspberry Pi gateway. Since the transmission unit also need to detect the opening and closing of the box the sensor system too was needed to implement in the same system.

**2.1.2 Development methodology**

RF24 transceiver module was used for the purpose. It uses the ‘NRF24L01’ transceiver IC which is an ultra-low power IC for data transmission.

Specifications of the transceiver:

* 900nA in power down
* 22μA in standby
* On chip voltage regulator
* 1.9 to 3.6V supply range
* Up to 2Mbps on air data rate
* Ultra-low power operation
* 11.3mA TX at 0dBm output power
* 12.3mA RX at 2Mbps air data rate

Since the requirement was to make the entire mobile unit low power a low power controller also needed to be used. To achieve this, a low power version of an Arduino board was used. The controller uses the well-known ATmega328 IC which is more improved for the low power consumption purposes. The power specification of the ATmega328P IC are as follows.



Figure 6: NRF24L01 transceiver module

* Active Mode: 0.2mA
* Power-down Mode: 0.1μA
* Power-save Mode: 0.75μA

Since the device is in its sleep modes most of the time of its operation, the Arduino Pro mini is a very suitable option for the purpose and the size of the board which is just which make it more compact to fix inside the MedBox.

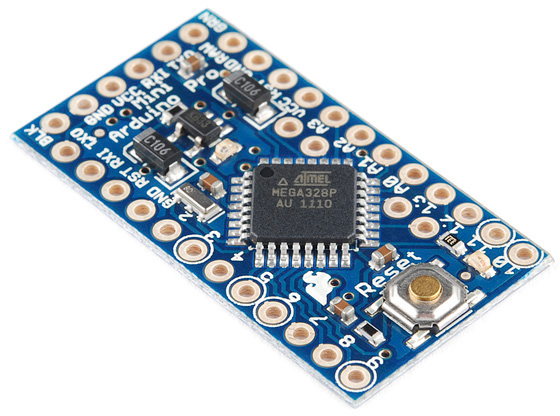


Figure 7: Aruino Pro Mini

After selecting the suitable hardware programming was started. The documentation for the NRF24L01 could be found at ‘tmrh20.github.io’ which is a very useful source to reference the operational information of the transceiver. Initially the sample codes given in the documentation web site is used to test the operation of the module. After initial testing the customization started. A two way handshake was implemented as the communication protocol. The previous iteration only had a one way handshake which is a burst of five packets of data which is not much robust in operation. The test performance of the developed system will be discussed by in the last paragraph of the subsection.

Arduino

Time Start

.

.

.

Time Stop

Raspberry Pi

Go to Sleep

Send Packet

Get ACK

Re-send Packet

Figure 8: Two way handshake implemented

The reed switched used in the system is a simple magnetic switch. It was mainly used due to its simplicity and long lasting nature when compared with a simple metal plate switch sensor. But the reed switch used to make a small bounce at its changes of state which should be removed either by using a hardware de-bouncer or a software de-bouncer.

After making the basic functions operational fine tuning and low power modes were added to the system. The system specification was to transmit a small ping every one hour and sensor data when an input is arrived. Besides all the time the system should be in the sleep mode or low power mode. To achieve this in the Arduino a Watch Dog Timer (WDT) was used with the sleep modes. But the problem with the timer was that the maximum time that could be set in the WDT timer is 8s. So achieve the low power performance it had to keep the WDT resetting until the required time is achieved. There is also a header developed by a third party called ‘Narcoleptic.h’ which does the same operation. But after doing so many tests it was found that there were some runtime bugs in the functions in the header, so I used my own sleeping function to get the work done.

Unfortunately the since iCity Lab is a data analytics lab the school is an Information System School the facility didn’t have any soldering facilities for me to solder some pins to the interrupt pins on the Arduino Pro Mini board. Due to this fact I was not able to proceed with the next step of the project which is de-bouncing of the input signal. The work was halted until I do the ‘Safety Briefing’ at iDA Lab which is a fabrication facility at Singapore National Design Centre.

**2.1.3 System testing the problems encountered**

As the first testing procedure I tested the receivers for its operational range. Since the boxes are located inside the house a very fair test was not able to be conducted due to the variation in the location specifications. So I tried for the worst possible indoor conditions so that we could make sure the proper functionality under any conditions. Indoor and outdoor tested operational ranges are as follows.

* Outdoor operational range: 50 m (Average)
* Indoor operational range: 15 m (Average)

These values are under normal power conditions of the transceiver which could be even maximized by using the max power mode. Since the tested values are good enough for the required specifications and running the system in low power is another concern the max power modes were ignored for the time being.

As a project management practice all the programmes were committed to Github repository for easy access, safety and versioning of the programmes. Developed programmes could be found at ‘github.com/bitandbytes/RF24-tranceriver-for-Arduino-and-Raspberry-Pi’

**2.2 Gear S2 data retrieval**

**2.2.1 Project description**

The final objective of the Gear S2 project is to develop a software platform to aid stroke patients to improve their motor skills. To do so, the project idea is to use the accelerometer in the SAMSUNG Gear S2 to detect the motion of the hand of the patient. Gear S2 is used due to its new advancements and comparatively longer battery life of the device.

So my work task is to retrieve accelerometer data and the heart rate data from the Gear S2 for the development platform.



Figure 9: SAMSUNG Gear S2

**2.2.2 Development methodology**

The new SAMSUNG Gear S2 is running the new Tizen wearable operating system. Tizen wearable is a new alternative to android which is newly rising. Even though the Tizen is new to the market it has some very interesting security features. One of the noticeable feature is the safety in the developer mode. In the Android OS any person who has an .apk can side-load applications to the device. But for Tizen OS anybody cannot side-load applications directly. Only a developer can install custom applications to a Tizen OS since a certificate is required to install applications.

For this project even installing the IDE and the SDK is complex and misleading. Downloading and installing the infrastructure software itself took about 7-8 hours. Unlike Android, Tizen uses Java Script as the programming language and uses HTML5 to develop interfaces. Unfortunately, I am not familiar with Java Script as well as HTML5. So it took some time to get familiarize with the syntaxes and interfaces. After following a basic tutorial I started working on the project. With the guidelines given in the SAMSUNG developer site I was able to retrieve accelerometer data, heart rare monitor data and barometer data. Satisfied the initial request made by the supervisor and also in the meantime another intern too moved into the lab who is more familiar with Java Script and HTML. So the project was handed over to the new intern.

**2.2.3 Problems encountered**

One of the biggest problem that I encountered in the process is the lack of proper documentation and community support for the Tizen SDK due to its novelty in the market. That actually consumed lot of time going through different forums, blogs and support pages. Even though I am a bit familiar with Android and XML a bit moving towards Java Script and HTML5 too was time consuming since I was not familiar with the syntaxes. Also there was not body in the lab who was familiar in the subject so I was on my own during the entire process.

**2.3 MedBox Project**

**2.3.1 Project description**

MedBox is a simple medication box which was used to monitor the medication adherence of the elderly. The currently deployment of the box has a simple reed switch and an Arduino board as the driver. Then the singles from the Arduino is send to a ‘BeagleBone’ board (a Linux based small size computer) which act as the gateway to the system via XBEE interface. Then the data is send to an MQTT broker which is operating in a cloud server via the 3G connection. Even though the collected sensor data is very small though proper data analytics it is possible to get very valuable results about the human behaviour.

One of the recent finding of the lab shows that any abnormalities in the medical adherence would show that the elderly is feeling sicker. Figure 10 is a graph extracted from a poster published by the lab.

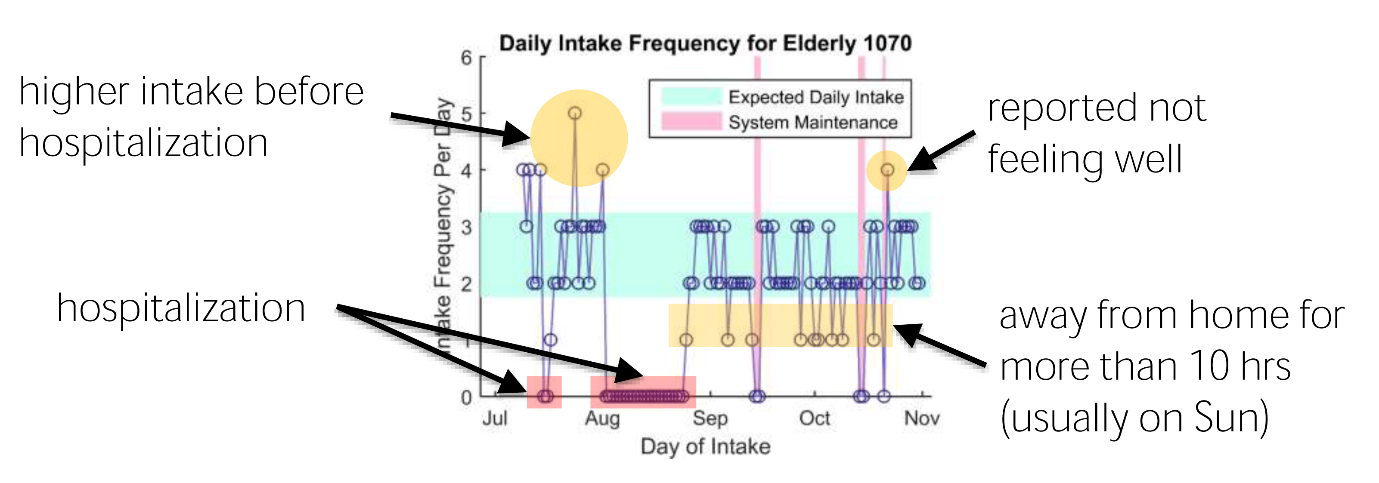
From the graph (Figure 10) it is evident that there are some excessive medical intake in the month of July especially in the end of the month. In August the graph is flat where the elderly was hospitalized nearly ¾ of the month. After that the adherence levels are normal as it should be. After getting to know about the ground truth, the elderly was taking more medicine since she/he was not feeling well during the month of July. So technically it is quite possible to determine the abnormalities of the graph (Figure 10) which would tell that the elderly is not feeling well in the month of July. In the later iterations of the box more sensitive data was planned to collect and more behavioural patterns are to be determined which will be discussed later in the section.

Figure 10: Medical adherence graph

MedBox project is the main project which I was involved, and about 80% of my internship was focusing in improving the MedBox gateway and the sensing systems. I was mainly involved in the next iteration of the MedBox gateway and the sensor system using Raspberry Pi, reed switch and Bluetooth Beacons

**2.3.2 Development methodology**

2.3.2.1 System Integration

Since the pervious iteration of the MedBox system had a BeagleBone gateway the next version was to change the gateway to a Raspberry Pi due to its stability, performance and useful features. To get familiar with the Linux shell, systemd and other system features my first task in the project was to integrate the BeagleBone service programme system into the Raspberry Pi. The system had six service applications which are to be executed as services in the Raspberry Pi. The set of service programmes are as follows.

* XBee listener programme and an MQTT client
* MQTT broker inside the Raspberry Pi
* Twitter service
* Remote MQTT publishing client
* Reverse SSH programme
* Service for 3G connectivity

The system was designed in a modular manner so remover the dependencies between the services as much as possible and making it able to develop parallely. Use of the MQTT protocol is also a very important feature in the system due to its quality features which will be discussed later in this sub subsection.

Raspberry Pi

MQTT

Brocker

XBee listener programme and an MQTT client

Reed Sensor Input

3G Connectivity

Remote MQTT publishing

Reverse SSH

Twitter service

SSH Server

Remote MQTT Brocker

twitter.com

ISP

Figure 11: Flow diagram of the gateway

The basic operation of the system is to get the data from the reed sensor and send it to a remote MQTT broker. In the MedBox gateway (Figure 11) the ‘XBee listener programme and an MQTT client’ service captures the input from the reed switch and publish the data into a local MQTT broker running inside the Raspberry Pi. This implementation was used with the idea of having more sensors to the system so that all the sensors could publish locally and to the same broker and that the ‘Remote MQTT publishing client’ service could publish the data to the remote broker. This simplifies the system than having multiple remote publishing clients inside the Raspberry Pi.

Beside the basic service there are some auxiliary services developed in to the system to provide feedback and maintain stability in the system. One of the implemented running services is the Twitter service which tweets to a pre-made twitter account every one hour. By this it is possible to know whether the Pi is up or down even when the SSH and or the MQTT publishing fails.

The Reverse SSH is another useful service which was implemented to the system. Generally due to the Network Address Translator (NAT) nodes behind the NAT is not visible. This same scenario happens due to a Firewall as well. Due to this problem it is not possible to access the gateways directly from a remote network.

As a solution for this problem a Reverse SSH connection is used. In the Reverse SSH connection the gateway (node) connects to a port of a remote SSH server which has a public IP. When the connection is made a remote user can log in to the remote SSH server and then into the specific port of that server using SSH. If the username and the password for the already connected device is known by the user she/he could access the gateway (node) through the remote SSH server. Being able to access the shell of the gateway gives a whole new level operational functionality including remote debugging, system testing, system maintenance and remote system updates.

One of the must to mention security feature of the SSH connection is the ssh-key. This ssh-key is generated by the ssh-keygen as pair of public and private keys. The private key should be kept private with the user and the public key can be shared with the SSH servers. So whenever a user try to access an SSH server the server goes through the set of public keys that it has and then provide access if it finds the public key of the private key that we are having with us. This way no password data is transferred in the line so that anybody who eavesdrop can’t get any information. Also users can share the same public key with multiple SSH servers so that the same private could be used to log into those servers. During the implementation few problems were aroused in the reverse-ssh scrip which will be discussed in the last paragraph of this sub subsection.

In the time when I am getting the System Integration work the developers were trying to implement a stable 3G connectivity system for the gateway. The developed system was based on the ‘wvdial’ application for Linux. One of the main problem in the given system was that it is not device independent and ISP independent. Every time a new 3G dongle was introduced it was required to make configurations specifically for the device. Also there were some bugs in the application where only some of the dongles works properly with the ‘wvdial’ system. So one of my initial goals were to make a stable 3G connectivity system for the gateway. As a result I developed a wrapper script for the old ‘sakis3g’ script which could monitors the 3G connections and ensures the full time connectivity to the 3G network. More details about the script and the work carried out will be discussed in the next sub-subsection.

Use of the MQTT protocol is another important feature in the system which make it an advanced IoT system. The MQTT protocol is a light-weight messaging protocol which is even used by the industry for IoT systems. The operation of the protocol is very much simple that a client can set a topic and publish to messages to that topic. Multiple clients too can published to the topic generated. Anybody who is subscribed to the topic will get the messages in real-time. One of the nicest features of the protocol is that if the Quality of Service (QoS) is set to ‘2’ even when the subscriber is offline the messages will be stored in the broker itself and all the messages which the client haven’t received will be set to the cline when subscribed again. MQTT protocol comes with a bag of some other parameters that could be used depending on the scenario. Parameters like client alive times and request for a new connection for publishing could be set as required.

One another request that came from the supervisor is to make the XBee device a Hot-pluggable device. In the give system it was required to reboot the OS in order to reload the XBee driver and the supporting services. For the device to be hot-pluggable it was required to detect the device and reload the drivers and supporting services. Fortunately the key to detect the device was already included in the Debian system. The only thing required to do is to add a file in rules.d folder and then execute a script which reload all the driver services. After writing the reload script and adding the driver detecting statement into the rules.d folder the XBee module was hot-pluggable and was working like a charm.

During the work process and at the testing stages loads of technical problems aroused. One of the initial problems was in the reverse-ssh service where the service cannot connect to the SSH server. The observation was that the SSH server keep dropping the handshake from the gateway when it’s rebooted or switched off immediately. When the connection request comes to a different port or the next handshake request is late over 12-15 hours from the previous one, the handshake is accepted. After going through all the parameters of the SSH service of the server I was able to figure out that the ClientAliveInterval, ClientAliveCountMax were not set in the SSH server. So the server doesn’t drop the client connection for a very long time which caused the error. Since the change should be done in the server side and I was not granted the root privileges to do so, I made a request to set the parameters. After these parameters were set the problem was solved and the reverse-ssh connection was working well.

After some time I made few copies of the Debian System and started testing. In that case another problem aroused in the reverse-ssh connection. For all the systems the same ‘private ssh-key’ was used for all the Raspberry Pis. Due to this scenario the SSH server started detecting the new systems as a Man in the Middle attack. In this case the server actually let the Pis to make the reverse-ssh handshake. But the problem arises when the client tries to connect to the gateway which gives a warning of MIM attack. So to bypass this warning it was required to set the SSH connection option StrictHostKeyChecking as ‘no’ and UserKnownHostsfile as ‘/dev/null’. Even though this bypasses the warning the techniques was unacceptable since there is a security risk in doing so.

When testing the system over couple of days another problem aroused. The problem was that the USB interface of the Raspberry Pi shutdown automatically due to no proper reason which eventually hangs the 3G connection. Even finding the point of error was so difficult in which I had to go through the system log very thoroughly to figure out the cause of this problem. After narrowing down the cause I had a long discussion Dr. Alvin VALERA on the matter. Finally a simple solution was came out which is to detect the 3G connection status using a simple ping command the then reboot the system whenever the 3G is down. The probability of occurring this scenario is very minimal; so the probability of losing any data due to this is reboot is insignificant. After adding the reboot scripts the system was working fine and 3G connection was up the entire time it was testing.

Final test was done on site where a gateway was launched in an elderly home in the area of Marine Parade. The deployed home is a special place since the previous gateways had problems in maintain proper 3G connectivity due to very low signal levels. Due to the new improvements in the 3G system, the gateway was functioning properly throughout the entire testing time which is over one month, which remarked a great achievement in my work. From the experiences that the research team had at the specific elderly’s home, they were much pleased about the stability of the 3G connection and the operation of the system.

The knowledge of using ‘systemd’ is a one very important thing which I learned during the system integration process. Systemd is a very useful tool in implementing system services specially when there are so many dependencies between each other. The older ‘init’ method needed specific order of execution which is tedious to implement in a system which has so many cross decencies. The new systemd solves that problem and the only configurations that were needed to make is to specify the dependencies for each of the services. Then the systemd will generate a specific order of starting the services.

In the systemd services file we can specify a considerable number of options which will be useful depending on different scenarios. As an example the service could be set to restart whenever it is aborted or excited due to any of the following reasons.

1. Clean exit code or signal
2. Unclean exit code
3. Unclean signal
4. Timeout
5. Watchdog

Also the delay until the next restart could be set to wait until the system settles down using ‘RestartSec’ parameter. Likewise there are over 30 parameters which could be set and manipulate the system services.

Even though the systemd was simple to use, adding too much unwanted dependencies make the system boot slower. After going through the current system software I removed some of the unwanted dependences which improved the boot-up times. The dependency manipulation was mainly done in the 3G services which will be discussed in the next sub-subsection.

2.3.2.2 Sakis3g wrapper script

Sakis3g is a bash script which could be used to initiate the internet connection via 3G modems. The script consist of bash function and a binary which could be decompiled, change and recompile if needed. The script contains a usb\_modeswitch which could be used to switch modes from a storage device to a network interface. More details about the usb\_modeswitch will be discussed later in this sub-subsection.

One of the key requests that came from Dr. TAN Hwee Xian is to make the 3G connection as stable as possible since their current gateway and the newly developing gateway had issues in the 3G interface. Previous deployment was using the default ‘Network Manager’ in Debian and the newly developing gateway was try to use ‘wvdial’. The ‘Network Manager’ was not very stable in maintaining the 3G connection and ‘wvdial’ need very specific configurations which is much dependant on the 3G modem driver and firmware.

Fortunately sakis3g script was much more independent in terms of the driver versions and firmware. The only dependency was the Access Point Name (APN) which depends on the ISP. Other than that the script could be set to auto configure most of the other configurations. By taking the advantage of these features I was able to make a very stable 3G service for the gateway.

Sakis3g script was mainly designed to be used with the interactive interface. But the requirement was to use the script in the bash shell and set the configuration directly from the options. Also the script did not have any auto re-connecting options in which I needed to be taken care of. So I wrote a wrapper bash script which take care of all the requirements. Features of the written bash script are as follows

* Initiate a 3G connection when the system boots.
* Monitor the 3G connectivity
* Re-connect when the connection hangs
* Switch modes from USB storage to 3G interface when required

In the initial version of the script the usb\_modeswitch feature was not included. One of the key functions of the older script is to setup of the options for the modem. The main problem with the course is that settings required to configure different modems is modem dependant. As an example some modems requires to select the USB interface which the 3G connection is operating, but some modems only had one interface which is directly detected by the script it-self. So to solve this problems I tried the required setting for the give set of modems and aggregated a set of options which works with all the modems.

Also included a 3G connectivity monitoring system using a simple ping and reconnect whenever the connection is lost. In the extreme case where a reconnection does not work out properly, I also added a reboot code which would initialize everything. This was mainly included as a solution to a problems that aroused in the hardware, at the long term testing stage.

The next problem came in the development process was due to a problem in the usb\_modeswitch which is currently embedded in the original sakis3g script. The current version did not support the new 3G models so the automatic mode switch was not functional. In order to made it functional I had to force mode switch the storage interface to a 3G interface. The most difficult part in doing a force mode switch is that I had to identify whether the device it is a 3G modem or not. The objective was achieved by going through the verbose data of the USB device and finding out the places which differentiate the old USB modem and the new USB modem. Finally I was able to figure out that the non-switching USB modems shows only one interface where the others USB modems shows more than one USB interface before switching in the verbose data mode. Taking this data into account I added a script on top of the old wrapper for the sakis3g which will detect thing devices which does not mode switch and then switch the modes forcefully by using the latest version of the usb\_modeswitch. This solved the problem giving only one constrain.

Constrain is that this methodology will only works on Huawei 3g modems. Since the lab is only purchasing the modems by the Huawei there is no problem with that. After finalizing the script another solution too came up to remove that contain by using the usb\_modeswitch database. The solution is to go through the files in the usb\_modeswitch data base and if the file is found for the respective modem, the mode switching could be carried out. Since there was no requirement at the moment the new methodology was not implemented. All contains and the new improvement ideas were included in the software documentation and all the files were uploaded to the Github repository.

Over the time of development, I was able to achieve a considerable improvements in terms of cumulative boot up times until the 3G modem makes the connection. Given below are more of the test results which was done on a Raspberry Pi B+.

|  |  |
| --- | --- |
| Case | Boot-up time |
| Original service with no auto reconnection | 1 min 30 sec (avg) |
| Service with auto re-connection | 2 min 35 sec (avg) |
| After cleaning services dependencies | 1 min 42 sec (avg) |

Figure 12: Table 1: Boot-up time improvement

Over the time with the improvement of hardware the system boot-up times improved drastically. In the new Raspberry Pi 3 the cumulative boot up time was only 32 seconds on average.

After finalizing the latest script with the usb\_modeswitch included the system was put to test for 5 days of continuous operation at the lab. As expected the 3G was stable and all the other services were working perfectly fine. A week before I leave the lab another set of gateways with the latest 3G script was deployed at two elderly houses and the system was working fine until I left the place.

2.3.2.3 LED Power Circuit

The designed LED power circuit is a voltage level shifter circuit which could be used to power three sets of LEDs. The requirement for the use of LEDs came as a request from Singapore’s Eastern Health Alliance (EHA) where they needed to have an alarming system to the MedBox implementation to notify elderly whenever they forget their medication. This changed the direction in which the design iterations were moving on since the previous requirement was to make the system as much as low power as possible. That’s mainly due to the use of LEDs as a notification system where the battery is more prone to drain. So the current iteration was revised and a new design came though where we unified the transmitter and gateway into the medication box itself. So the LEDs will be directly driven for the Raspberry Pi itself. This simplifies the system considerably and reduce the maintenance cost of replacing the batteries. The only trade-off was that the MedBox should be directly powered by the home circuit and the box is immobile.

Besides that there was another problem in using the LEDs with the Raspberry Pi due to the difference in the voltage levels. Raspberry Pi GPIO pin logic is 3.3v and 0v which is not acceptable to be used with LEDs. Also the specified maximum draw-able/sink-able current is 16mA which also not sufficient to drive the LED array that is used in the designed. So it was required to make a voltage level changing circuit which could handle the specified current levels. In order to achieve that a very simple MOSFET circuit was designed using BS170 N- Channel MOSFET. Since the design had three arrays of LEDs, three sets of the MOSFET circuits were required to use in parallel.

Since the main requirement if the circuit design is to limit the current draw/sink from the GPIO pin the resistor values were needed to be carefully selected. So I designed one set of the MOSFET circuit in LTSpice which is a very popular platform used to simulate circuit performance.

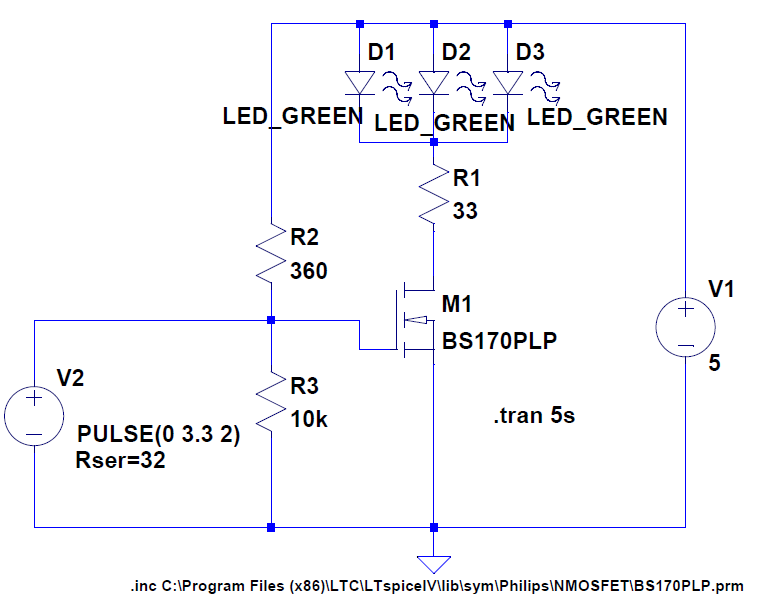


Figure 13: LTSpice schematic for the MOSFET circuit

In designing the circuit one simple problem came across was the internal resistance of the GPIO pin which is not known. So I did a simple experiment using different resistor and measured the current though the path when the pin output is high. Then plotted a graph and get the internal resistance of the GPIO pin. With that value in had all the LTSpice models for the generic LEDs and BS170 MOSFET, I started simulating the circuit for the specified current and voltage values. After going through number of trials an acceptable set of values were found for the circuit.

After finding the correct resistor values I started designing the circuit. One of the biggest challenge in the design process was that I had to make the circuit design which could be printed on a single layer PCB. After coupe of iterations a circuit design was made and the PCB was fabricated at the iDA labs which is located in Singapore National Design Centre. In the initial design the circuit was mainly fabricated for the testing purposes. The wire connection were needed to be made using a ribbon cable for the Raspberry Pi.

After the initial design another idea came though by Dr. Alvin VALERA to make the circuit fix on top of the Raspberry Pi as a shield. If I was able to make it small enough that it could fix it inside the current enclosure of the Raspberry Pi the implementation of the new circuit will be very simple since there was no requirement to make any physical changes to the excising enclosure and hardware. With that idea in mind I started designing a new circuit which could be fixed on top of the Raspberry Pi and also fix inside the currently available Raspberry Pi enclosure. After few attempts I came up with two designs which meets the requirements. After discussing with the supervisors one design was selected to be fabricated. That was selected by considering the orientation of the heard pins and the convenience of using the pins.