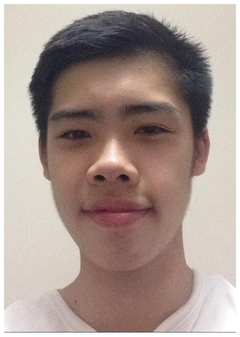
2.0 Project wiki
https://github.com/Raymond-Y/ares-aqua/wiki
4.0 Team Member List and Roles
Team Member 1: Raymond Yu - 45896192
In this project, Raymond will be responsible for handling the python processing of data received from the base station. Raymond will read RSSI values and determine the location of the mobile nodes via the least square equation. Raymond must also use a Kalman filter to develop a localisation system which combines the RSSI data and the acceleration sensor info. The predicted locations for both mobile nodes must be shown on a live updating GUI. Sensor data received must also be uploaded to a web dashboard interface such as InfluxDB. This sensor data will be used as training data used by the KNN machine learning algorithm to improve the accuracy of localisation. If the estimated locations of the nodes are too close, Raymond must send an alert to the base.

Team Member 2: Zak Burke - 45895140

Zak is responsible for programming the static, mobile and base nodes using BLE mesh. This mesh will allow the mobile node sensor readings to be transferred to the base no matter where the base is located within the building. Zak will also be implementing the IMU sensor unit which will measure speed and heading information and transmit this back to the base node for greater accuracy. Finally, Zak will be analysing the data sent back after processing to alert users when they are too close to each other for social distancing purposes.

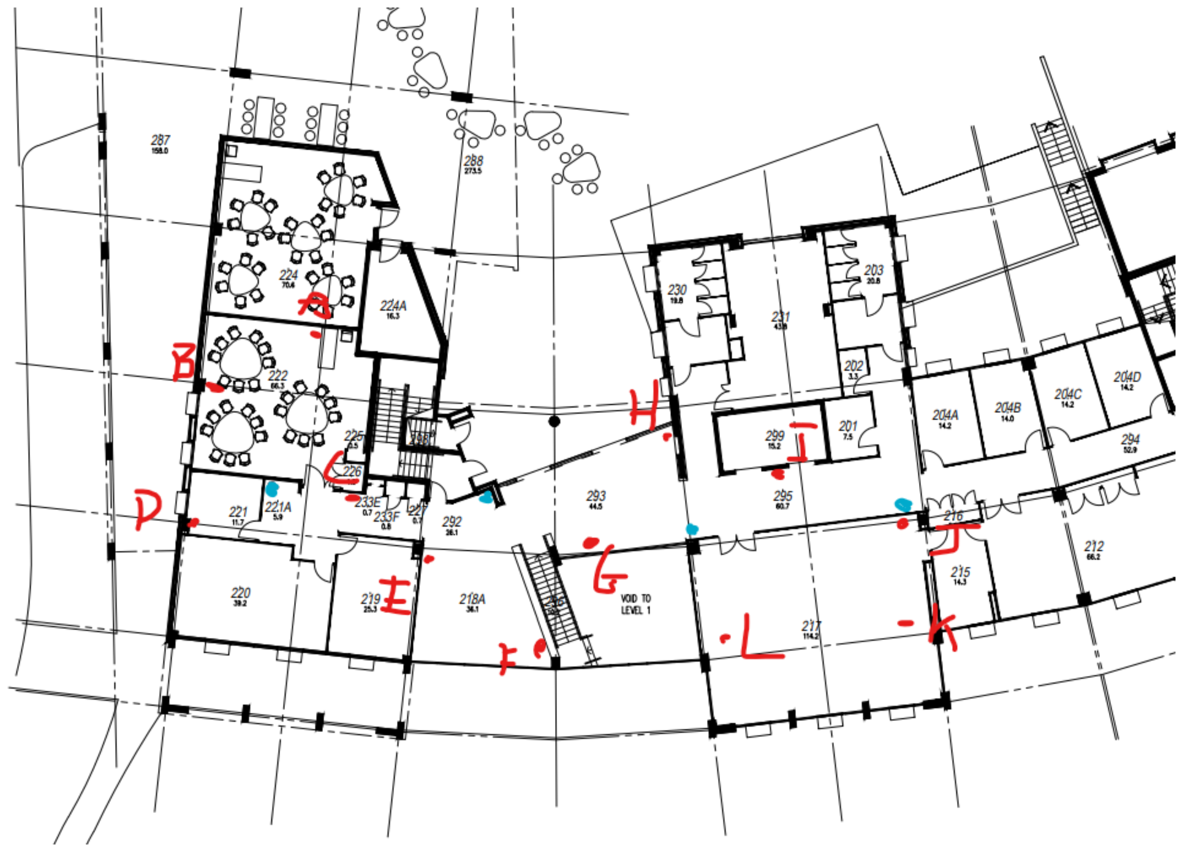
Team photo:



5.0 Project Overview/ Scenario

5.1 Project and Scenario Description

In this project, up to two users will be holding a mobile node (thingy52) and walking around a section of the GP south building (UQ building 78). Beacons that advertise in intervals of 100ms will be placed around level 2 of the building with static nodes being placed in the hallway so as to always have a connection with a base node while being inside in a room.



This project requires practical 2 to be extended to allow tracking of the 2 users within the 2nd level of the GP south building (building 78). In accordance with recommended Queensland health guidelines, when users are too close together, users must be warned. Exceptions for these warnings must be implemented, in cases such as when the two users are from the same household.

5.2 KPIs- Key Performance Indicators

1. <u>Implementation of bluetooth mesh network to allow data transferred from mobile nodes to the base</u>
   Due to the large area in which users can be, there are situations where the base is located too far away from the mobile node for a direct connection to be formed. This kpi involves the implementation of a bluetooth mesh network containing the static nodes, the

base and the mobile nodes which ensures that data can be transferred from the mobile nodes to the base. Due to strategic placement of static nodes spread out along the tested area, it is ensured that the base will always be able to connect with the nearest static node.
Initially, the mobile node will read surrounding ibeacon and static node advertised rssi values. Then these readings along with the mobile node's motion info will be transmitted from the mobile through the mesh network until it reaches the base node. If the base node has already received data with the same packet ID then it will discard it.
In this respect, this project can be considered successful if the base is able to receive data at all locations within the specified area in building 78. Both RSSI info and motion info will be transferred successfully.

2. <u>Localisation of the two mobile nodes</u>
   This KPI requires a python program to be developed which attempts to track the location of the two mobile nodes. One localisation system will be developed which requires the rssi values of static nodes and ibeacons surrounding the mobile node to be transferred to the base. Using these values along with the known static coordinates of these ibeacons and static nodes, the least square equation must be solved to estimate the location of the mobile nodes.
   Then, this estimation will be fused together with the received acceleration data using a Kalman filter to produce more accurate predicted mobile node locations.
   The positions of both mobile nodes will be displayed on a python GUI in real time. Due to the limited number of sensors used, this KPI can be considered successful if the predicted positions are accurate within 3-4 metres.

3. <u>Social Distancing Feature Implementation</u>
   For this KPI, it requires the implementation of a warning system to alert users when they are too close and not social distancing properly. Although the recommended social distance length is 1.5m, due to the limitations of the amount of sensors which limit the localisation estimations to around 3-4m, this range will be used instead.

   One of the exceptions to these warnings are when the two users are of the same household. One way of implementing household type is using the mobile node's led. The led on the mobile node can be red, green and blue along with different combinations of these colours. However, this is still only limited to around 7 colours. The led colour will indicate the specific family that the user belongs to and can be changed easily by pressing the button on the mobile node. If the colours on the mobile node are the same for two users, then even when the users are too close, the warning will not be sent.
   The led will be constant when the users are not within the 3-4m radius. When the users are too close, the leds will flash rapidly as a warning.

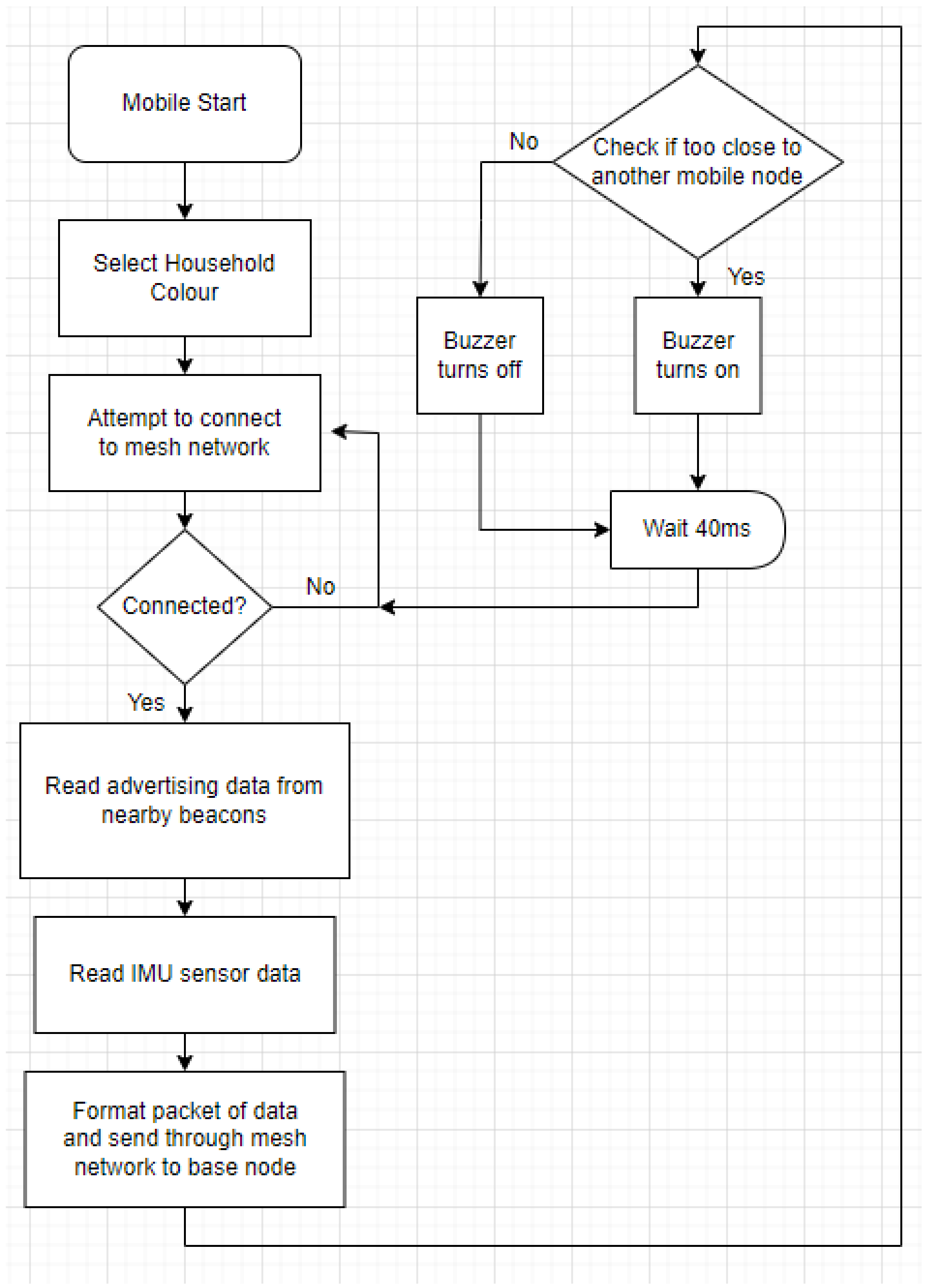
4. <u>Web Dashboard</u>

For this KPI, the base unit must be able to upload received RSSI and motion data in real time to the web dashboard. This KPI also requires a sufficient amount of data to be uploaded, which can be used for training.

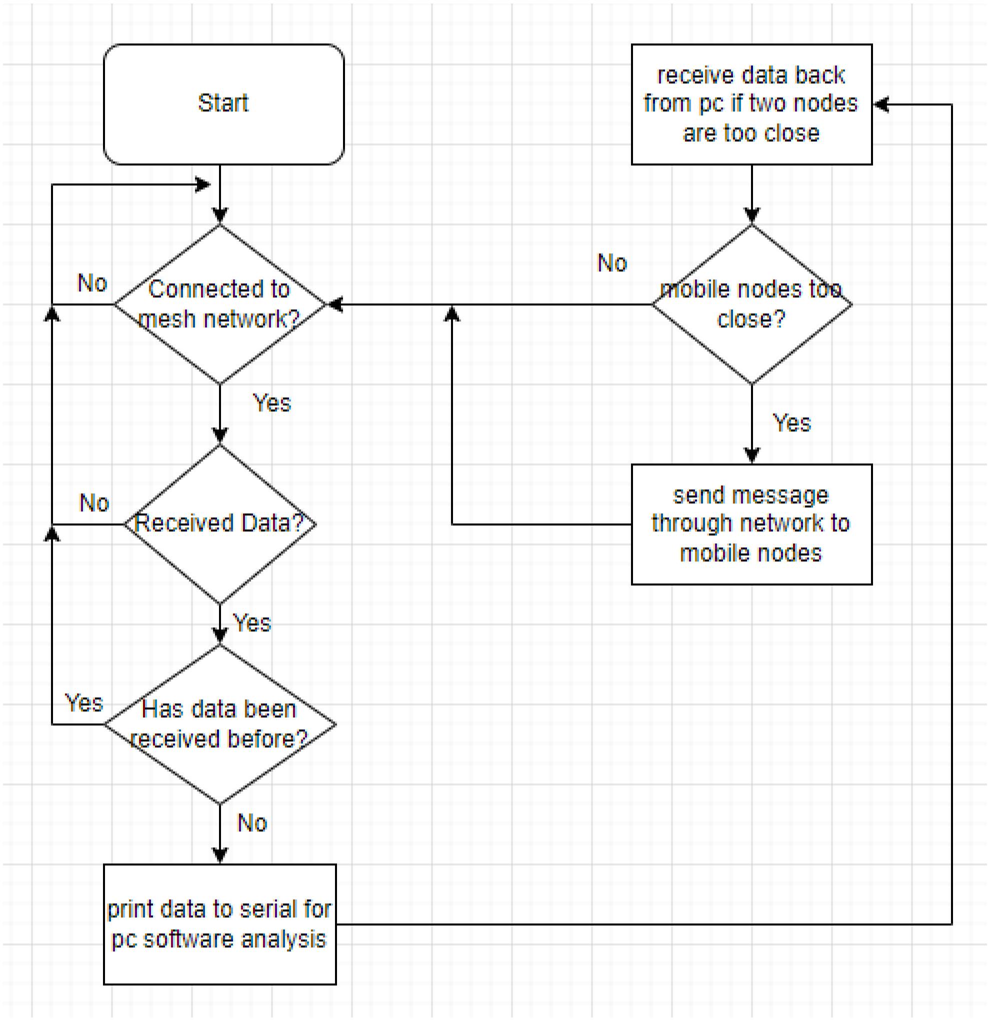
5. Machine Learning - KNN
   For this KPI, the machine learning KNN algorithm must be applied such that the localisation system which utilises RSSI values will have improved accuracy.
   - KNN should improve localisation accuracy to ~2m

5.3 System Overview

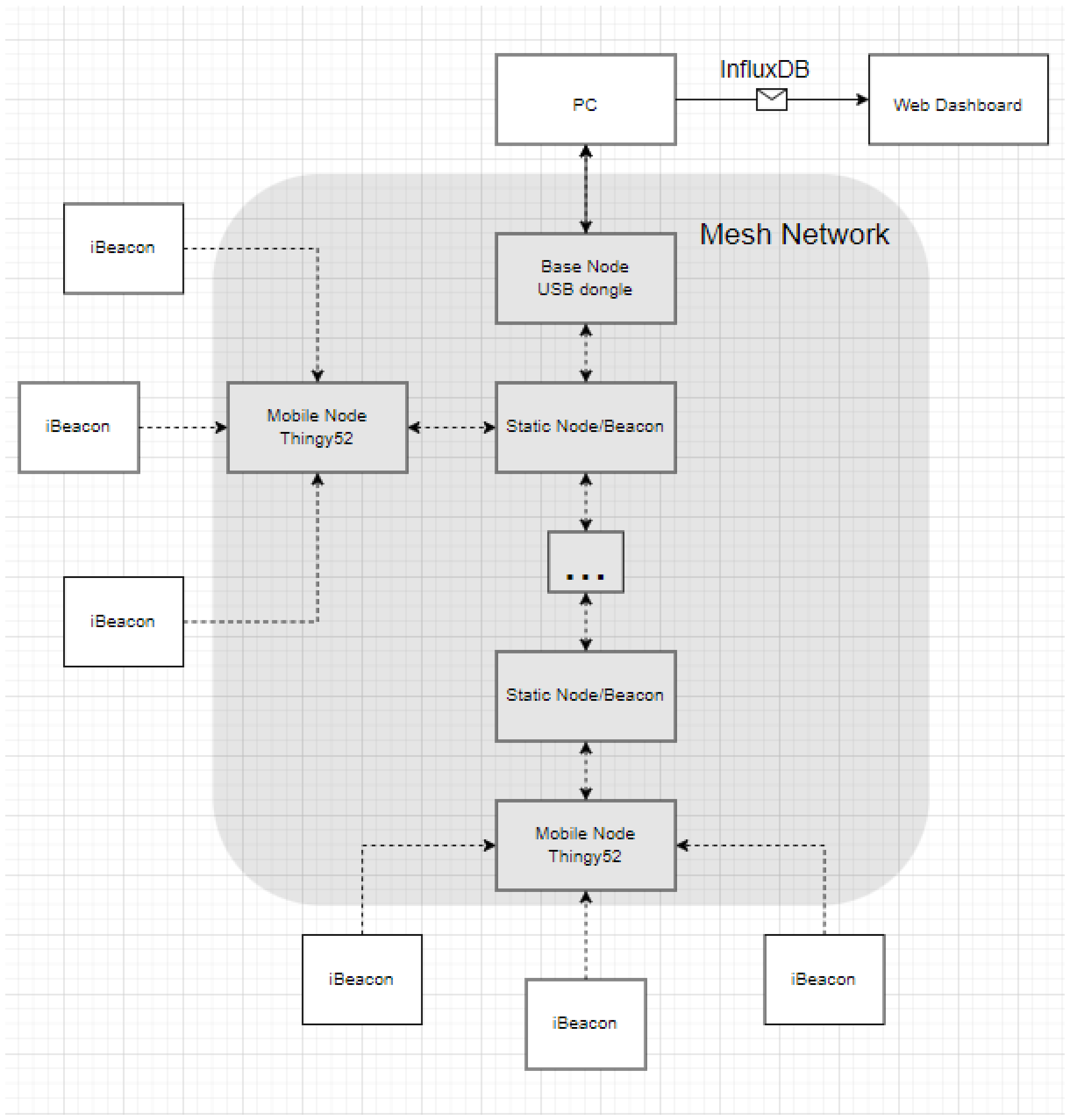
MOTE

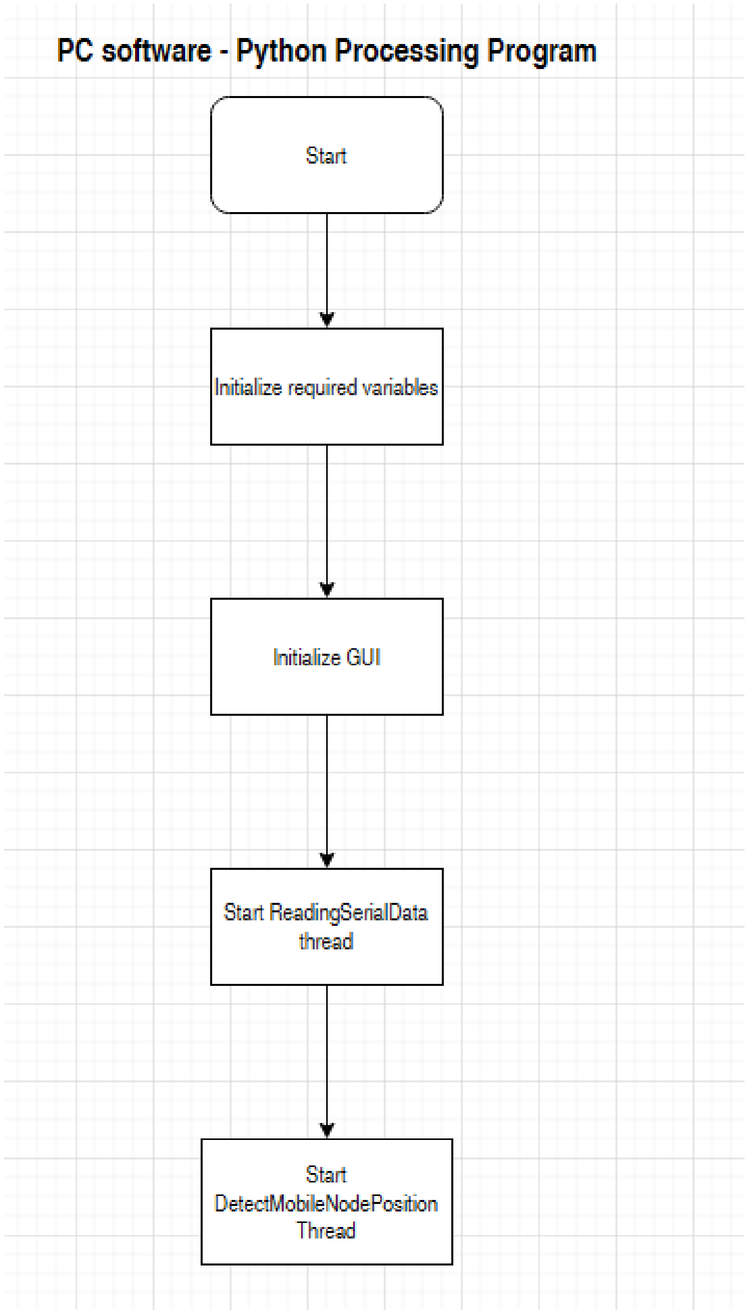

Mobile Start
Select Household Colour
Attempt to connect to mesh network
Connected?
No
Yes
Read advertising data from nearby beacons
Read IMU sensor data
Format packet of data and send through mesh network to base node
Check if too close to another mobile node
No
Yes
Buzzer turns off
Buzzer turns on
Wait 40ms

BASE NODE

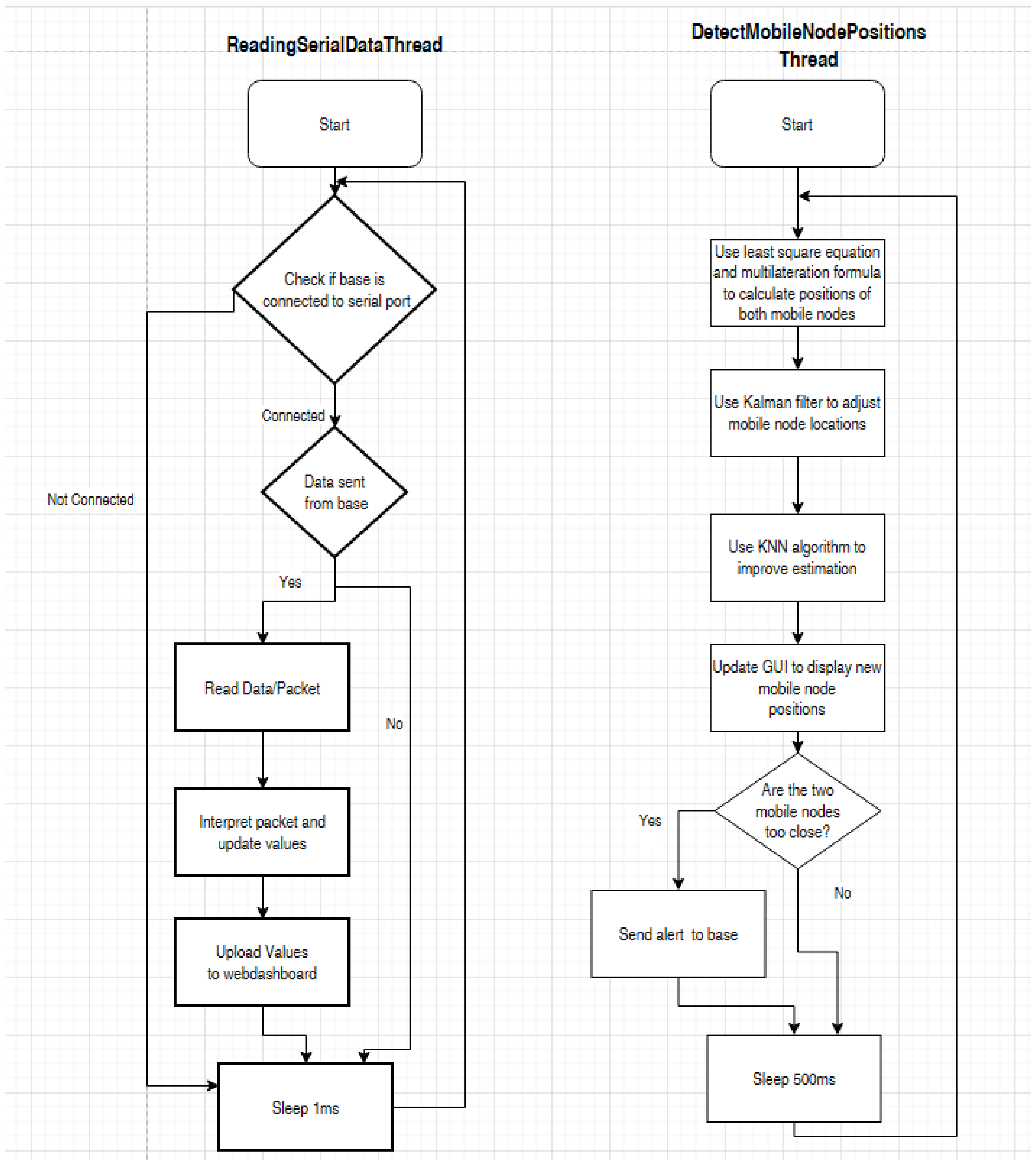

Start
receive data back from pc if two nodes are too close
No
Connected to mesh network?
mobile nodes too close?
Yes
Yes
No
Received Data?
send message through network to mobile nodes
Yes
Yes
Has data been received before?
No
print data to serial for pc software analysis

Block diagram of system




PC software - Python Processing Program
Start
Initialize required variables
Initialize GUI
Start ReadingSerialData thread
Start DetectMobileNodePosition Thread

The python processing program will have a program flow similar to the flowchart above. The GUI will be initialised and then two threads will be started.More specific details above how the threads operate is shown below.

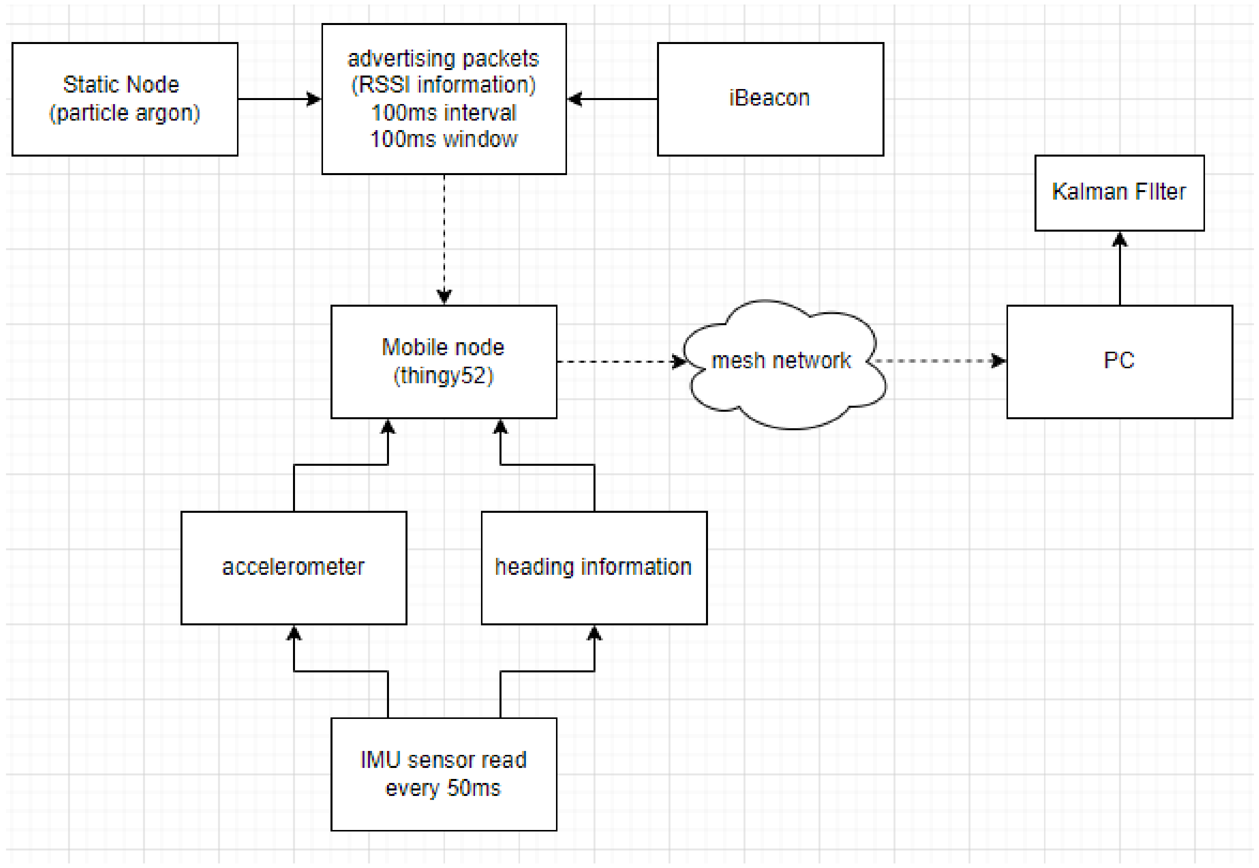

ReadingSerialDataThread
Start
Check if base is connected to serial port
Connected
Not Connected
Data sent from base
Yes
No
Read Data/Packet
Interpret packet and update values
Upload Values to webdashboard
Sleep 1ms
DetectMobileNodePositions Thread
Start
Use least square equation and multilateration formula to calculate positions of both mobile nodes
Use Kalman filter to adjust mobile node locations
Use KNN algorithm to improve estimation
Update GUI to display new mobile node positions
Are the two mobile nodes too close?
Yes
No
Send alert to base
Sleep 500ms

5.4 Sensor Integration

Inertial Measurement Unit (IMU) sensor will be used on the thingy52 to measure acceleration and heading information. This in combination with the RSSI data received from beacons and static nodes will be transmitted through the bluetooth mesh to the base node to be analysed by the PC software using a kalman filter.

The thingy52 will read the acceleration and heading information data every 50ms through a non-blocking thread.

The thingy52 will also scan for advertisements with a 100ms interval and a 100ms window, this is because the iBeacons and Static Nodes will both be transmitting every 100ms and this minimises the chance of missing an advertisement packet.


Static Node (particle argon)
advertising packets (RSSI information) 100ms interval 100ms window
iBeacon
Kalman FIlter
Mobile node (thingy52)
mesh network
PC
accelerometer
heading information
IMU sensor read every 50ms

5.5 Wireless Network Communication

The network topology used is mainly a bluetooth mesh network which contains the base, the static nodes and the mobile nodes. This mesh network is used such that no matter the location of the mobile nodes and the base, the base will always be able to receive data from the mobile nodes. The static nodes are placed in positions such that the mobile will always be able to transfer to at least one static node. The static nodes are placed such that they will always be able to reach at least one other static node. The Base should be positioned so as to always be able to reach at least one static node.
When the static nodes receive sensor readings from the mobile nodes, if this particular static node is directly connected to the base, it will pass the readings to the base. Otherwise, it will

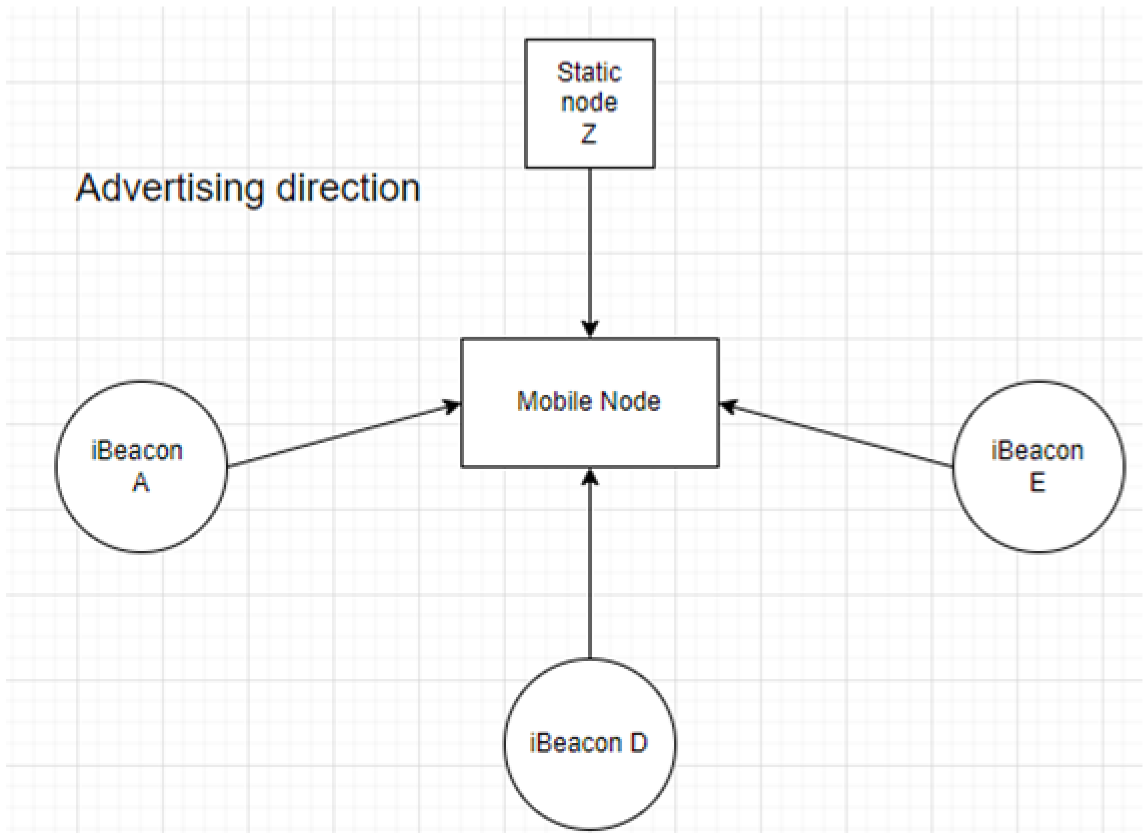
continue to transfer the readings to other static nodes via bluetooth mesh until it reaches the static node with the connection to the base.

Mobile to Mesh Messaging Protocol

Data size will depend on the number of beacon advertisements successfully delivered to the mobile node. In the example below, there will be 4 sets of beacon ID and beacon RSSI values and a data size of 8 bytes.

preamble	sender ID	receiver ID	data size (bytes)	data	packet id (counter)
1 byte (0xAA)	1 byte	1 byte	1 byte	variable	1 byte

Data structure Example



beacon ID	beacon RSSI	…	accel x	accel y	accel z	heading information
1 byte	1 byte	depends on number of beacons	2 bytes	2 bytes	2 bytes	1 byte (1.4 degree increments)

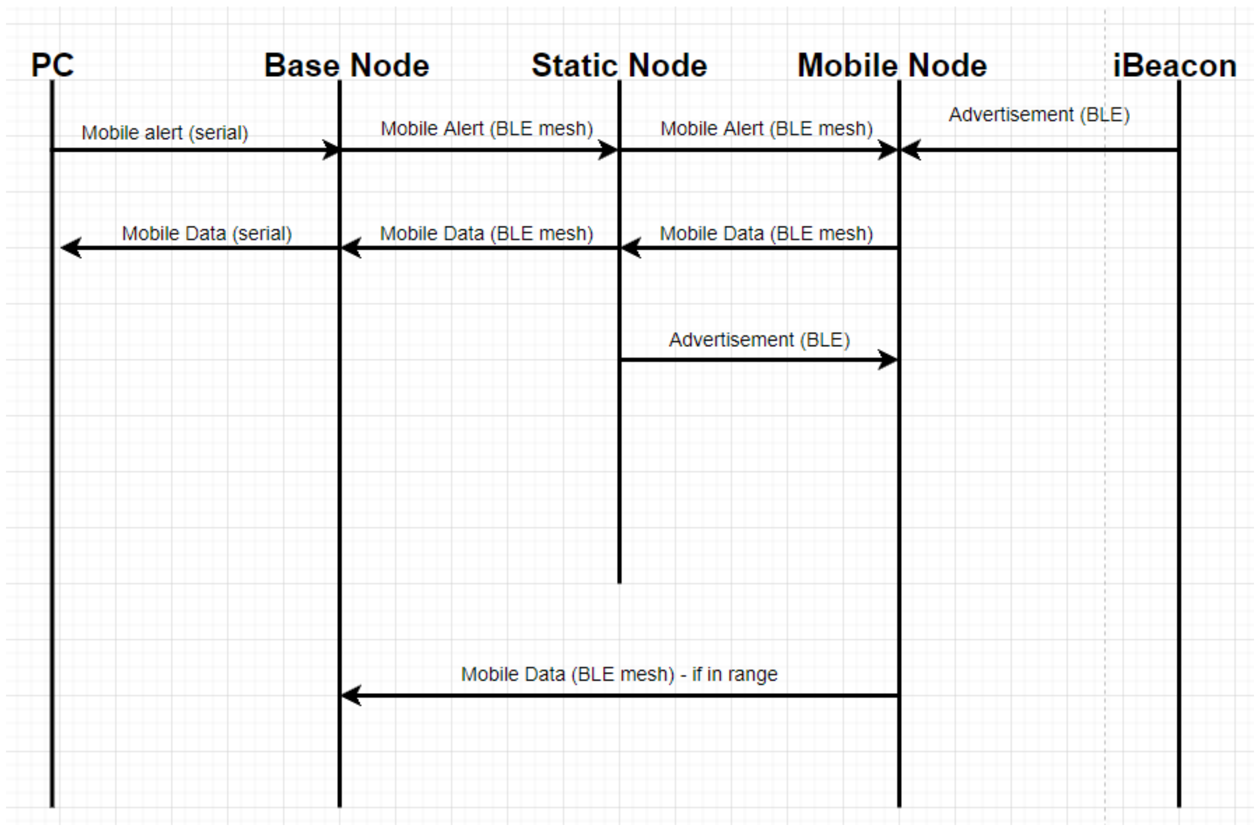
Base to Mesh Messaging Protocol

preamble	sender ID	receiver ID	proximity warning (ON - 0x01 or OFF - 0x00)
1 byte (0xBB)	1 byte	1 byte	1 byte

Information passed from the base to the python processing program will be in JSON format as outlined in the message protocol diagram below.

Top-view Messaging Protocol

The below diagram shows the overview of how messages will and data will be transmitted across the network



<u>Json format output from Base to PC</u>

Output format : '{"Device_ID": "Device_ID_val", [Data] }'

- Device_Id_val -> indicates value of device
  - Ibeacons A-L will have a letter value of "A" to "L"
  - Static Nodes will have a value of "Z" to "W"

- Mobile Nodes will have a value of ‘1’ or ‘2’

- [Data] – Varies depending if it is a iBeacon/Static Node or a mobile node
  - For ibeacons/Static nodes, data format : ‘ “rssi”: “rssi_val” ’
  - For mobile nodes, data format: ‘ “xaccel”: “xaccel_val”, “yaccel”: “yaccel_val”, “zaccel”: “zaccel_val”, “heading”: “heading_val” ’

5.6 Algorithm Schemes
Multilateration and least squares equation used to determine position of mobile node based on rssi data

- Originally -> set error to 0

$$f_i = r_i - \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2} = 0$$

- Rearrange equation to linear format

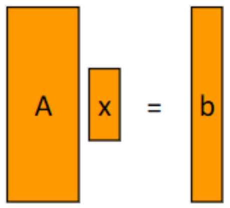
$$2x_0(x_k - x_i) + 2y_0(y_k - y_i) = r_i^2 - r_k^2 - x_i^2 - y_i^2 + x_k^2 + y_k^2$$

- Form system of equations

$$Ax = b$$

$$b = \begin{bmatrix} r_1^2 - r_k^2 - x_1^2 - y_1^2 + x_k^2 + y_k^2 \\ r_2^2 - r_k^2 - x_2^2 - y_2^2 + x_k^2 + y_k^2 \\ \cdots \\ r_{k-1}^2 - r_k^2 - x_{k-1}^2 - y_{k-1}^2 + x_k^2 + y_k^2 \end{bmatrix} \quad A = \begin{bmatrix} 2(x_k - x_1) & 2(y_k - y_1) \\ 2(x_k - x_2) & 2(y_k - y_2) \\ \cdots & \cdots \\ 2(x_k - x_{k-1}) & 2(y_k - y_{k-1}) \end{bmatrix}$$

$$x = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$



- In these equations, k will be 16 as there are 12 ibeacons and 4 static nodes
- 1-12 are for ibeacons and 13-16 for static nodes
- The x/y values are the x/y coordinate values of the locations of the ibeacons/static nodes
- x0/y0 is the estimated x/y coordinate location of the mobile node
- ‘r’ - are the RSSI values received from the ibeacons/static nodes

        - Then use least square equations to solve it

$$x = (A^T A)^{-1} A^T b$$

- Info referenced from CSSE4011_Lec6

KNN Machine Learning Algorithm

- KNN will be used to improve the accuracy of localisation using RSSI values

- Inputs required:
    - RSSI training data
    - Distance metric for distances between data points
    - K value - > number of nearest neighbours
- Steps to execute KNN
    - Obtain new data point
    - Find the distances between new data point and all the previous training data points
    - Find and rank the top/closest k nearest neighbours
    - Majority vote
- Info referenced from CSSE4011_lec7

6.0 Equipment

- 2 x nRF52832 Nordic Thingy:52 (mobile node)
- 4 x nRF52840 particle argon (static node)
- 12 x iBeacon
- nRF52840 USB dongle (base node)
- PC with python installed
- 2x Micro usb cables (for power to j link and static node/mobile node)
- J-link segger debugger (for programming)
- Lithium 3V cr2477n button cell batteries (iBeacon)
- GP South

7.0 Progress