# *Smart bus system of waiting time estimation and special passengers tracking*

Jingwen Jiang
*Central Michigan University*
*Department of Engineering*
Mount Pleasant,
United States
jiang1j@cmich.edu

Jianing Tan
*Central Michigan University*
*Department of Engineering*
Mount Pleasant,
United States
tan2j@cmich.edu

Emmanuel Sedicol
*Waterfords Institute of Technology*
*School of Science & Computing*
Waterford,
Ireland
sedicolemmanuel@gmail.com

***Abstract: As one of the main public transportations in modern society, buses are distributed everywhere. However, there are also many problems around bus. First, unknowing the specific position of buses, riders always waste lots of time waiting at bus stops. There are also cases that people find the buses are full when they arrive and have to wait for the next one. In addition, as for parents, they always wonder if their young kids get on the bus or miss the bus. This study research a smart bus system which can estimate waiting time based on bus position tracking and passenger load measuring. It can also track children on bus. All buses have a GPS module attached to them to get the real-time location and a raspberry pi which can measure the passenger load on the bus by detecting the bluetooth transmissions from bluetooth devices. There is also a small feature to the system where parents are able to monitor whether their children missed the bus or not. After analysing the data on the cloud, the location of the bus, the number of passengers and empty seats on bus and the estimated waiting time will be feedbacked to riders and transportation company.***

***Keywords: , Internet of Things(IoT), Smart bus system, GPS, Waiting time, Bluetooth,Real-time***

## I. Introduction

A smart city is a hot tendency because it can help government organize the city and make people's life more convenient by analyzing the data collected by different devices based on Internet of Things (IoT) technology. As a part of smart city, transportation system plays an important role in building it. Smart bus is a efficient tool to collect useful data so there is a need to improve the bus. Nowadays, people always waste a lot of time waiting for the bus. There are also cases that people find the bus is full loaded have to wait for the next one. The main reason of this problems is that passengers don't know the exact time when the bus that they want to ride arrives. A better smart bus system can be implemented if the real-time position of bus, estimated waiting time and the number of passengers are known. In addition, different from home and school where young children can be protected by teachers and parents, bus is a public place which is lack of security guarantee. Being worried of kids, parents wonder whether their children get on the bus or not. It is better for smart bus that the bus has a system can track young kids to see whether they get on the bus or not. Furthermore, it cannot be denied that optimizing the transportation is very important in the smart city. All the real-time data collected by bus can be sent to bus control center to help them make the transportation system more flexible.

## II. Previous Work

In the past several years, vehicle tracking became a hot topic which helped a lot to offer the real-time information of buses. To build a vehicle tracking system, the most common methods are using GPS and using Bluetooth technologies. As for the GPS module, some researchers used Arduino Microcontroller, GPS module and GSM module to establish a bus tracking system(Lee, Tewolde, and Kwon's lecture, 2014), which achieved real-time bus tracking. Another group of researchers used bluetooth for the bus tracking (A.Kumar, A. Mathur, A. Kumar, N. Aggarwal , 2017).

The concept of bus tracking was also applied in area of people tracking. Different from bus tracking, people tracking needs the recognition of ID which help distinguish people. The RFID technology represents a breakthrough in embedded communication and can be used to identify virtually any object, including animals, clothes, and

even human beings.**[4]** (Davis, Steve,2004) As a low consumption application bluetooth beacon also play an important role in people tracking.(Lee.Jg, Kim.J, Lee. Sw , Ko. Yw, 2017)

Number of people measurement in public place has been researched many years. There are many related work about it has done using different ways. A group of researchers achieved passenger trip record in public bus by using RFID to track bus cards and Bluetooth to count number of passenger on the bus. (V. Kostakos,T. Camacho,C. Mantero, 2010). It is also accurate to count passengers through a single camera fixed at an overhead position(J.Perng, T.Wang, Y.Hsu, B.Wu, 2016) With the rapid development of BLE which has low power consumption. Using Bluetooth Low Energy and to measure the amounts of people is becoming more popular.(R. Apoorv, P.Mathur, 2016)

## III. Specification

As for estimating the waiting time, the study researched two main factors that decide the waiting time of riders. One is the real-time position of bus and the other is the number of passengers on bus.
A brief discussion on technologies and methods to meet the different functions of the smart bus system is given below.

### *A. Bus Tracking System*

Every smart bus has a GPS module to locate the bus. The position is one of the significant factors that needed to estimate the waiting time for riders. Wireless communication can be defined as transfer of information between GPS module and the cloud without using wires or cables. Firstly, the time have relationship with the distance between the bus and bus stop, what's more, comparing the time between one stop and the next one to the normal time and we can know the traffic condition at that area at that time which can offer more accurate information for system to analyze the time when the bus will arrive. Passengers can control the time when they need to get to the bus stop based on the waiting time provided. It helps passengers who utilize this system save much more time. It also helps bus dispatch center to develop the public transportation system like adjusting the bus schedule, rebuilding the bus routine etc.

### *B. Passengers Load Measuring*

With the rapid development of technology, almost everyone have a smart phone with bluetooth and the number of bluetooth signal on bus can be regarded as the number of passengers. As is known, raspberry pi has a bluetooth receiver which have the function of detecting the bluetooth transmissions from beacon devices. Installed on the bus, the raspberry pi collects bluetooth signals of phones on the bus. The number of address equals to the number of passengers on every bus and the data is sent to the cloud and feedbacked to riders. So that riders can know if there are empty seats on the next coming bus. The data can also be used to estimate the waiting time. The system compares the real-time number to the load factor automatically. If the real-time number of passengers is close to the load factor, The waiting time estimated by system is supposed to be much longer.

### *C. Parental monitor application*

In addition, this smart bus system will contain an application to inform parents about their children's current status. The aim of this application is to automatically transmit a message by text or through an App simply telling the parents if their children missed the bus or not. A bus time charts will also be sent to the parents or it could be just a single message containing the time of the next bus. Beacon can transmit a unique identifier which help distinguishing people, this unique identifier will be used to identify people who are 12 years under or people aged over 12. This application will prevent delays simple because parents are able to react and quickly pick the child and drive them to school in the event of missing the bus.

## IV. Methond

The concept of this project used GPS module attached to wifi node which is ESP8266 12E in this project to locate the position of bus and collect real-time speed of bus. With these information, the group calculated the arriving time of bus on the cloud. BLE embedded in raspberry pi is regarded as a receiver to detect bluetooth signals from smart phones and ibeacon which has specific address. The Raspberry Pi has an ability to verify multiple Bluetooth devices at a time and recognize certain address. Therefore multiple ibeacons approaching

the bus can be uniquely identified by the system **[5]**.

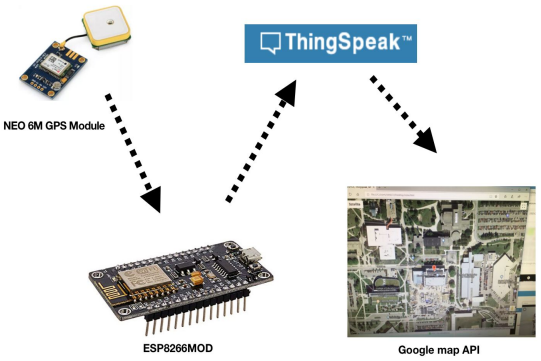



Fig.1 Architecture

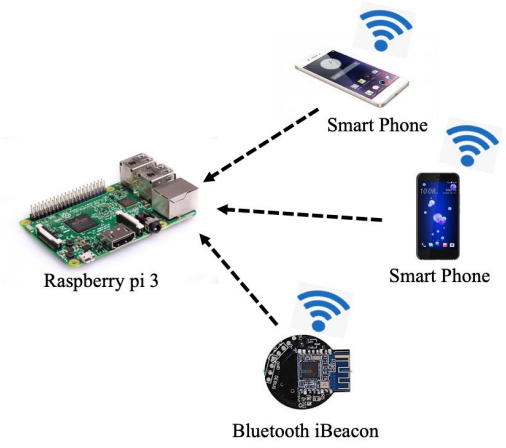



Fig.2 Architecture

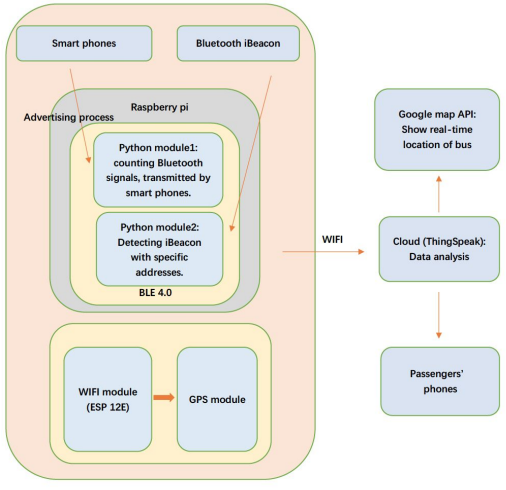



Fig.3 Block diagrams

The various electronic equipment used in this prototype are:

A. Hardware

*(a) Raspberry Pi 3:*

The Raspberry Pi 3 is the third generation Raspberry Pi. It has replaced the Raspberry Pi 2 Model B in February 2016. It consists of in-build Bluetooth 4.0 which keeps on looking for a Bluetooth device nearby**[6]**.

*(b) GPS module:*

The Global Positioning System in vehicle tracking systems is commonly used to provide users with information such as the location coordinates, speed, time, and so on, anywhere on Earth.**[2]** Our project decided to use NEO 6m as the GPS module.

*(c) iBeacon:*

Every iBeacon has a unique address which can be used to separate it with others. The system used iBeacon as a symbol of kid and utilize BLE 4.0 as a receiver to recognize the signal transmitted by iBeacon.

*(d) Wi-Fi module*

Wi-Fi module helps to send data measured by GPS module to the cloud through wifi. The group chose ESP8266 12E as the Wi-Fi module in this project. ESP8266 is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability. It can be easily coded through Arduino software.

B. Software

*(a) MQTT:*

MQTT is a machine-to-machine / "Internet of Things" connectivity protocol, which is a special set of rules that specify interactions between communication.

The basic structure of MQTT is that when a message is published with a topic name to the MQTT Broker, the MQTT Broker broadcasts the message, and only the subscriber with the same topic name can receive the message.

*(b) Cloud:*

The project used thingspeak cloud as a platform to analyze the data which was transferred to cloud. The specific analysis is as follows:

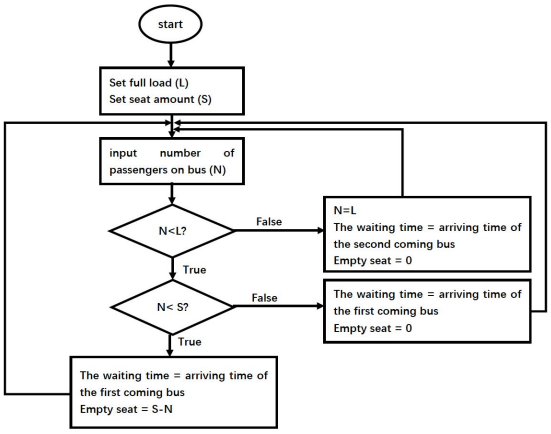



Fig.4 Logic on cloud

*(c) Google Map API:*

The Google Maps API gives developers the opportunity to overlay their own data on top of tiled map layers from Google Maps. The overlaid data is typically supplied through KML files, and is displayed as interactive vector graphics drawn on the client side**[7]**.

## V. Expriments And Results

### A. Experiment 1: Global Position System

The aim of this experiment was to get the location, time and speed of GPS module which will be installed on bus in the future.

The group connected GPS module (NEO 6m) to the Wifi module(ESP8266 12E).Firstly, we wrote the code of ESP8266 12E, which including write API key and read API key from Thingspeak, to send the data (longitude, latitude, speed) to the cloud. After coding for the ESP8266 12E, we created a notepad and put the Google map API's code on it that needed Google map API key, then translated it to html, so that it could display the actual location on Google map.

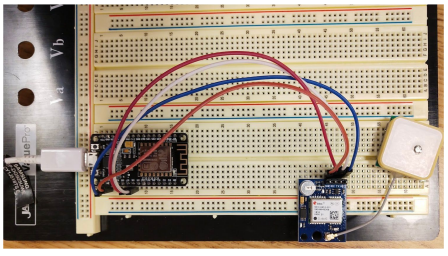


Fig.5 Hookup

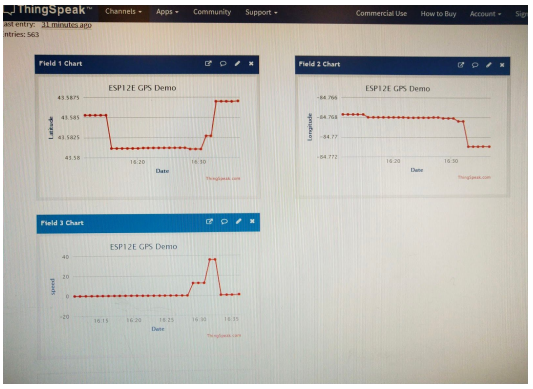



Fig.6 interface of cloud

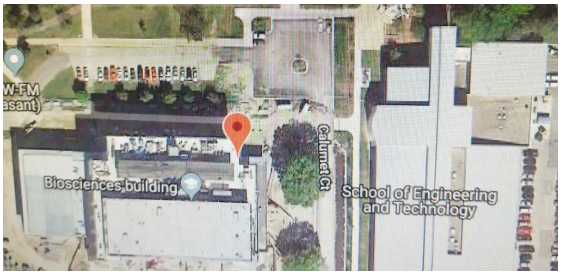
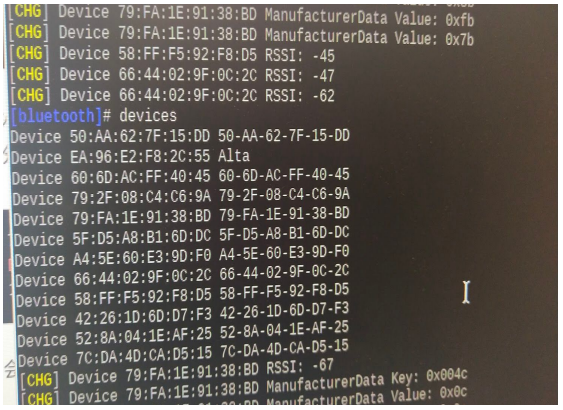



Fig.7 show location on google map

### B. Experiment 2: Bluetooth signals scanner

The aim was to determine how many the bluetooth signal detected by raspberry pi 3 nearby.

Set up BLE 4.0 on raspberry pi using python language. A list of devices and their addresses showed on the screen and it represented the number of smartphone nearby.




Python code to count the number of seats assuming the seats number is 40:

```
def countemptySeat(foundAddr):
    os.system('clear')
    print ('''
          ========================================
              Number of Available Seats
          ========================================
            ''')

    numberOfPeople = len(foundAddr)

    for i in max.all():
        count = i['seats']
        x = count - numberOfPeople
        print("\t\tNumber of Available seats in the Bus : ", x, "]\n\n\n\n\n\n\n")
        max.purge()
        max.insert({'seats' : x})
```

After running the code, we opened 2 bluetooth devices which means there were 2 passengers on the bus and got the number of empty seats is 38.

```
=========================================
        Number of Available Seats
=========================================

Number of Available seats in the Bus : [ 38 ]
```

Fig.8 Results of scanned devices

C. Experiment 3: Parental Monitor Application

The aim of the Parental Monitor Application is to detect the bluetooth signals from an iBeacon or phones using the built in bluetooth sensor in Raspberry Pi. We the used the bluetooth package in python to store all the detected bluetooth devices in an array which will be used in the compare method. Before any calculation theres is two detection stage. First detection is the initial detection, then the second detection is to check for any remaining devices that arrived late. After two detection the bus countdown of ten seconds starts then door closes and the calculation starts.

Python code for detecting devices:

```
nearby_devices = bluetooth.discover_devices(lookup_names=True)
print("[ Number of devices found %d ]" % len(nearby_devices))

# add detected addr in
for addr, name in nearby_devices:
    print("\t\t[ %s - %s ]"% (addr, name))
    foundAddr.append(addr)
```

The group decided to use the Nexmo API to send SMS in the event of any absence from a certain child. First we created a Nexmo account to get our own API credentials, then we installed the Nexmo module using the command:

```
esedicol$ sudo apt-get install Nexmo
```

**Your API credentials**

| Key | Secret |
|---|---|
| 4179aed7 | B6ED8JS3Kap5blsI |

Python code for sending SMS:

```
c = nexmo.Client(key='4179aed7', secret='B6ED8JS3Kap5bIsI')
c.send_message({
        'from': 'Nexmo',
        'to': str(sendTo),
        'text': '%s missed the bus on (%d/%d/%d.)'
        % (name, curr.month, curr.day, curr.year)
    })
```

The group decided to use MongoDB to store and access data. The database will include: children name, their unique BLE address and parents mobile number.

```
import pymongo
from pymongo import MongoClient

client = MongoClient('localhost' ,27017)
db = client.iot
children = db.children
```

Unfortunately we aren't able to use the Mongo Database due to version error so we decided to a different DB called TinyDB.

```
pymongo.errors.ConfigurationError: Server at localhost:27017 reports
n 0, but this version of PyMongo requires at least 2 (MongoDB 2.6).
>>>
```

TinyDB

```
from tinydb import TinyDB

db = TinyDB('iot.json')
mx = TinyDB('max.json')

db.insert({'name' : 'Emmanuel', 'addr' : '8C:85:90:AA:B0', 'no' : 353963767437 })

db.insert({'name' : 'Nathaniel', 'addr' : '8C:85:90:CE:R0', 'no' : 353963767437 })
```

Our presence detection code worked and the raspberry pi was able to detect the beacons by scanning for addresses and comparing the addresses found to the registered addresses.

```
--> Number of devices found 2 ]

--> [ Device Name :  E.Sedicol  ]
--> [ Bluetooth Address :  8C:85:90:AA:AF:B0  ]


--> [ Device Name :  Samsung Galaxy S7  ]
--> [ Bluetooth Address :  84:98:66:29:21:6A  ]
```

Which means the system is able to detect the children, calculate their presence and send alert messages.


## VI. Evaluation

Overall, the goals of the smart bus system is obvious.

Waiting rime evaluation system

Success:

The system feedbacks an estimated waiting time to people who wait for a bus:

Firstly, the system estimates the bus arriving time on the cloud based on the real time speed as well as the distance between bus and bus stop collected by GPS module. Furthermore, the system measures the number of Bluetooth signals successfully and have ability to do analysis on the cloud. and the system can send the estimated waiting time back after

analyzing the data on the cloud. Lastly, the system use Google map API to display real-time bus track.
Failure:
The final resut is far from real situation:
The microcontroller cannot connect with the cloud and do analysis on it. And the Bluetooth receiver cannot detect signals successfully. Beside this, the system does not establish database of traffic information to predict arriving time. And the system can not show the bus track successfully.

Parental monitor application
Success:
System is able to detect and record the presence or absence of a child and also do analysis by running python scripts. Furthermore, system sends notification (text) to school or parents whether child missed the bus or not and be able to get time table and send to parents.
Failure:
System is not able to detect presence of child and not able to send message. Furthermore, system can't access cloud allow device connections fails to allow device connections.

## VII. Discussion

The group used ESP12E to connect GPS module, because NEO 6m GPS module can not get satellite signals indoor. Therefore, the group used ESP 12E instead of connecting with Raspberry pi directly. After coding on ESP 12E, the data collected from GPS module can be display on cloud( ThingSpeak) including: longitude, latitude and speed. Though analysis on the cloud, the waiting time can be calculated. In addition, Thingspeak can share data with Google map API by using Google map API key. Some limitations are still not solved. Firstly, the distance is straight-line distance, which used in waiting time calculation, but in real world, there are variety of road conditions, which may lead to waiting time estimation not accurate. Secondly, the real distance of every degree changes when the latitude changes. This problem also affected accuracy of waiting time estimation. As for the bluetooth signal detection, the group successfully detected several signals. However, the range of signal is going to be considered because the system doesn't expect a very large signal range of bluetooth receiver. The system is only going to scan the smart phones and iBeacons which are on the bus. In addition, the group did bluetooth ibeacon detection, which is an important part in parental monitor system. Overall the parental monitoring system worked as expected. We were able to detect and store the detected bluetooth addresses, also we were able to use the TinyDB for storing purposes in the cloud. Accessing TinyDB was super simple and we were able to query the data needed, using the queried data we were able to do our compare algorithm which worked with minimal errors. We made sure the output was user friendly and easy to read, which we achieve using python scripts. The last point to work on is with twilio, we are starting to try out a few more options other than twilio as a way of sending messages and we are comparing each one. In the end of our test with different API's, we finalised on using Nexmo, which worked the best compared. As of now we aren't able to send anything. A timetable system which we added in the end using the Date Module in python which gave us the current time. Based on the current time we decided that our bus would arrive every 30 mins, we basically did basic operations on the current time to output the estimated time of the next bus. Our main goal was to have a predefined timetable stored in our database, we would then use that data to calculate next bus. Overall all we are happy with this application and even though it has a lot of room for improvements we are proud of what we have accomplished in a short amount of time.

## REFERENCES


[1].C. Sungur, I. Babaoglu and A. Sungur, "Smart Bus Station-Passenger Information System," 2015 2nd International Conference on Information Science and Control Engineering (ICISCE), Shanghai, China, 2015, pp. 921-925.

[2].S.Lee, G.Tewolde, J.Kwon, "Design and implementation of vehicle tracking system using GPS/GSM/GPRS technology and smartphone application", IEEE, Seoul, South Korea, 2014

[3]. S. N. Divekar, S. R. Patil, S. A. Shelke "Smart Bus System", IJSRSET, vol. 4, pp.585-P588, 2018

[4] I.Hashem, V. Chang, N. Badrul,K.Adewolea, I.Yaqooba, A. Gani , E. Ahmeda, H. Chiroma, "The role of big data in smart city" ,International Journal of Information Management, Vol 36, October 2016, Pages 748-758

[5] Newman, Nic. "Apple iBeacon technology briefing." Journal of Direct, Data and Digital Marketing Practice 15.3, 2014.

[6] Monk, Simon. Programming the Raspberry Pi: getting started with Python. McGraw Hill Professional, 2015.

[7]What is a web mapping API www.e-education.psu.edu