BusPlan

ECE 445 Design Document

SCOTT LIU
sliu125@illinois.edu

CONNOR LAKE
crlake2@illinois.edu

AASHISH KAPUR
askapur2@illinois.edu

TA: EVAN WIDLOSKI
October 5, 2018
Contents

I Introduction 3
   i Objective ................................................. 3
   ii Background ......................................... 3
   iii High-Level Requirements ......................... 4

II Design 4
   i Power Supply ...................................... 6
      i.1 Solar Panel ........................................ 6
      i.2 Li-Ion Charger .................................. 6
      i.3 Li-Ion Battery .................................. 6
      i.4 Voltage Regulator ............................... 7
   ii Control Unit ........................................ 7
   iii Transmission Unit .................................. 8
   iv Visual Feedback .................................... 8
   v Data Storage ......................................... 9
   vi Central Server ....................................... 9
      vi.1 API / Database / Web interface ............... 9
      vi.2 Security Protocol ............................... 9
   vii Data Analysis ...................................... 10
      vii.1 Prediction Models .............................. 10
      vii.2 Scheduling Algorithms ........................ 10
   viii Schematics ......................................... 11
   ix Board Layout ......................................... 11
  x Tolerance Analysis ................................... 12

III Cost and Schedule 12
   i Cost .................................................... 12
      i.1 Labor .............................................. 12
      i.2 Parts ............................................. 12
   ii Schedule .............................................. 14

IV Safety and Ethics 15
   i Potential Hazards .................................... 15
   ii Data Privacy .......................................... 15
   iii Digital Security ...................................... 15
   iv General ............................................... 15
   v Current Competition .................................. 16
I. INTRODUCTION

i. Objective

Buses are a form of standard mass transit service and are usually dispatched at fixed frequencies [9]. For example, MTD deploys their buses roughly once every 10 minutes during school hours and once every 30 minutes otherwise [6]. However, during rush hours we have observed buses to be completely packed; while on other occasions we have also found buses to carry only very few passengers. This poor load-balancing results in inefficiencies at multiple stages: losing existing customers due to overwhelmingly full buses, deterring potential customers due to crowded first impressions, operating at extremely light loads, etc.

Herbert argued in his article within the context of urban bus transportation: “Fewer riders lead to less frequent service leads to fewer riders” [13]. Guihaire and Hao also speculate the car drivers might consider switching to public transport if we have a less crowded system [9]. It is therefore imperative that we develop a solution to address this commonplace issue.

Our team proposes the BusPlan project: a network of smart detectors that actively survey the amount of people waiting for a bus, gather information, and send this information to a central unit that performs aggregation and analysis. Ultimately, our BusPlan service would be able to find patterns, predict trends, and provide relevant visualizations back to the bus company, who will use this to make strategic business decisions.

ii. Background

Four stages of public transport operations have been identified by Ceder in his book Public Transport Timetabling and Vehicle Scheduling [3]:

1. Network route design
2. Setting timetables
3. Scheduling vehicles to trips
4. Assignment of bus crew

These steps together form a well established mode of public transport operation planning, and has already been adopted successfully worldwide. However, our team believes that the heart of the aforementioned problem lies within steps 2 and 3. The innovative network of devices that we propose would disrupt the traditional workflow, and greatly enhance the system’s productivity and efficiency. By automatically detecting the amount of passengers waiting at a bus stop frequently and accurately, we would be able to obtain a sequence of numbers taken at successive equally spaced points in time. Furthermore, by analyzing and manipulating this time series data using modern tools, we would be able to achieve powerful forecasts that would replace timetables and scheduling completely. We believe that this would provide significant value to the bus company.

Many theories have already been proposed by experts to find an optimal way to form timetables or schedules [1] [10] [4] [8]. Some propose laying out optimal networks [5] while others focus on forming timetables after the fact [11] with real world case studies. They all offer incredible insight into formulating ideas to tackle this formidable challenge. However, most of them fall short when considering a truly dynamic environment that is subject to frequent change. Even theories that incorporate computer programs [17] [12] have been developed without this particular consideration. This is purely because they can only ever study past, outdated information as opposed to a stream of present, ever-changing information.
The buses that MTD use to provide service are already digitally augmented with STOPwatch technology [15]. It would be completely reasonable to add additional features to further enhance its efficiency. We also recognize the importance of a minimum bus service frequency [2] and will cater to delivering a model which takes this factor into account.

iii. High-Level Requirements

- Nodes must be able to collect an accurate measurement of number of people at bus stops by listening to WiFi probe signals
- Nodes must be able to relay data to a centralized server through other nodes
- We must be able to make predictions with reasonable accuracy and issue bus deployment recommendations that effectively alleviate unbalanced bus loads

II. Design

Each detector has three main parts: a power supply, a control unit, and a WiFi module. The power supply ensures that the system is powered continuously. The WiFi module listens for probe beacons from nearby WiFi stations (phones, laptops, etc). The radio chip connects and relays data from one node to surrounding nodes. The controller unit reads the data from the WiFi chip and intelligently caches and sends it using the radio chip.

The physical design should be a flat and thin box with a large surface area on top for the solar panel, preferably around the size of a MacBook trackpad (4” x 5.5”), with a height approx. equal to a laptop as well. Smaller is better for our device, so we will try and fit the components into as small of a case as possible.

Figure 1 illustrates the basic hardware design.
Figure 2 illustrates the high level design overview.
Figure 3 illustrates a more detailed, node-specific hardware architecture.
Figure 2: High Level Overview

Figure 3: Node Level Overview
i. Power Supply

i.1 Solar Panel

Each detector will be powered using a solar panel, which will charge the battery and power the device. Bus service runs 7am to 2am the following day, however the load at the edge hours will be much less, so we may have the scanning frequency lower than normal, saving power. Worst case, the solar panel must provide enough power for 19 hours of use.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outputs 500mA-1.5A between 3.40V-6.50V in full sunlight</td>
<td>• Place a solar panel in sunlight outside and, using a voltmeter, measure the voltage of the solar panel, verifying that it is in the provided range. Then, create a circuit with a 4.4V voltage drop, and verify the current is within the provided range.</td>
</tr>
</tbody>
</table>

i.2 Li-Ion Charger

The charger takes power from the solar panel and converts it to the correct voltage/amperage for the battery. It makes sure to use constant current or constant voltages stages to charge the battery without charging it too fast.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Li-ion battery charges to 3.65-3.85V when an input voltage between 3.4-5.5V is applied</td>
<td>• Charge battery with a voltage of 3.4-5.5V, and verify that it fully charges after 8 hours by checking that the STAT pin of the BQ2416 chip is high. • Verify that the battery does not reach temperatures greater than 125 degrees C (using an IR thermometer) while charging, indicating a fault with a battery</td>
</tr>
</tbody>
</table>

i.3 Li-Ion Battery

The lithium-ion battery will act as a buffer to aid in power delivery from the solar panel. We do not expect to collect and store enough power to last the device through the night. Instead we will optimize the power usage and size of the battery through empirical power draw testing.
<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Must store &gt; 4200mAh of charge</td>
<td>Connect a fully-charged Li-ion battery with a test circuit discharging the battery at 300mA per hour, and make sure that the battery voltage is above 3.7V for the entire time</td>
</tr>
</tbody>
</table>

### i.4 Voltage Regulator

This integrated circuit supplies the required 3.3V to the node system. This chip must be able to handle the peak input from the battery (4.2V) at the peak current draw (~300mA).

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>• The voltage regulator must provide 3.3V ± 5% from a 3.7V to 4.2V source.</td>
<td>• Plug in a power supply to the input of the Voltage regulator, and a multimeter to the output, and verify that, while changing the voltage between 3.7-4.2V, the output is 3.3V ± 5%</td>
</tr>
<tr>
<td>• Must maintain thermal stability below 125°C at a peak current draw of 200mA.</td>
<td>• Create a test circuit which draws 200mA, and connect it to the output of the voltage regulator. Connect the input to a power supply, and use a IR thermometer to verify the temperature of the regulator</td>
</tr>
</tbody>
</table>

### ii. Control Unit

We would like to use esp8266 WiFi chip to power the logic of the device as well. However since none of us have experience using this chip, we may have trouble connecting to and programming it to successfully perform our desired functionality. We may especially have trouble getting it to talk successfully with the NRF24L01. If this is the case we plan on using a much more familiar chip to us, the ATMega microcontroller as the core controller of our device. This is because it is generic enough to talk to both the esp8266 and the NRF14L01 in real time to manage and control the data.
### Requirement Verification

- The WiFi chip must listen for and count the amount of probe requests, and correlate that to the number of people nearby.
- It must also successfully communicate with and use the NRF radio chip to send the information it gathers. See the transmission unit for the verification of this.

- We will go to different places and compare the number of people that the esp8266 thinks are nearby to the actual number of nearby people. We will also use a separate WiFi module to count the WiFi stations that are nearby. This will ensure the chip is accurately counting the WiFi stations (devices) as well as then guessing the amount of people off that information.
- To make sure it communicates with the NRF chip we will use a known working receiver to listen if the NRF is successfully sending information dictated by the ESP8266.

### iii. Transmission Unit

The NRF24L01 is the primary means of communication for our node, as it’s used for communicating between the node and the data collection node on the bus.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Each node must be able to successfully find and accurately send its stored data to the drive-by node. Each node must be able to send the information in a short enough time and be within range of the drive-by node as it goes by the bus stop.</td>
<td>We will ensure that the hash of the sent data matches the hash of the received data. We will also do empirical testing including testing from inside a bus to ensure that the devices can talk to each other and send the data in time as the bus is going past each bus stop.</td>
</tr>
</tbody>
</table>

### iv. Visual Feedback

Our device would have several status LEDs to indicate the overall status of the device (e.g. if it has successfully been powered on). It would also have two 7 segment displays that shows the current number of nearby WiFi devices.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>The LEDs should display the current power state of the board.</td>
<td>Once we plug in the board, the power LED should light up.</td>
</tr>
</tbody>
</table>
v. Data Storage

Our device needs to store temporary data before it is transmitted to our central server. The esp8266 has built-in flash memory that we will use to store the detected MAC addresses and their timestamps before the NRF24L01 can connect to a nearby device and pass the data on.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>The device should cache the data of seen mac addresses locally.</td>
<td>After a certain amount of time we will request data from the device and confirm that it successfully kept track of all the data it was logging, and sends it all.</td>
</tr>
</tbody>
</table>

vi. Central Server

We will be building an API to be hosted on a central server which will be in charge of getting the gathered and transmitted data from all the devices, performing the bus frequency optimization, and giving data visualizations. The encrypted data will be submitted through an endpoint, decrypted, and placed in a database for processing.

vi.1 API / Database / Web interface

The central server will run on a docker stack on a Linux server. It will be exposed via an API that will listen for new data from the drive-by-drive-by node. It will store this information in a database and make it available in a web interface and also through the ELK stack.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server should accept MAC data via an API endpoint and store in a mongo database</td>
<td>Test the API using Postman and mongo client to verify data is properly accepted and stored in the database</td>
</tr>
</tbody>
</table>

vi.2 Security Protocol

In order to securely transmit the data from the nodes to the server, we plan on using asymmetric cryptography. Each node will have the public key of the central server and encrypt all the data it sends to the drive-by-node with this public key. Only the server then can decrypt the data once it arrives.
### Requirement | Verification
--- | ---
Nodes can successfully encrypt data that only the central server can decrypt | Check to make sure all data leaving each node is fully encrypted by monitoring the air and network traffic. Check that the decrypted data once received by the server is indeed the same exact contents as what was originally encrypted by hashing both the pre-encrypted value and the post-encrypted value.

vii. Data Analysis

vii.1 Prediction Models

The data will be collected in a way such that each entry corresponds to a timestamp. This way we will have a time series where we can train a LSTM model. We will use this model to make predictions.

| Requirement | Verification |
--- | --- |
Predictions must be made and be within a reasonable accuracy range | Verification scripts are to be developed and run to demonstrate that our daily predictions are on average within a 20% tolerance range.

vii.2 Scheduling Algorithms

To schedule the buses, we will first predict the amount of people required at each bus station, and then followed by finding out how many bus should be deployed in advanced to ensure that every bus has the optimal load.

| Requirement | Verification |
--- | --- |
We will need to provide effective scheduling to reduce the unbalanced bus load problem | • We will use software to simulate the effects of our scheduling, and verify that our scheduling does in fact alleviate the problem by comparing the non-scheduled results and the scheduled results and see that, on average, the standard deviation of the load on the bus has decreased by at least 20%  
• Additionally, we will also need to guarantee at least one bus per 20 minute time window |
viii. Schematics

Figure 4 illustrates the circuit schematics.

![Figure 4: Basic Hardware Design](image)

ix. Board Layout

Figure 5 illustrates the PCB design and layout.

![Figure 5: PCB Design](image)
x. Tolerance Analysis

We are using a solar panel that can vary a lot in the amount of power it provides depending on the sun and light it receives. We want to maintain an average power to sustain the device, plus a little extra to charge the battery. On the esp8266 datasheet, which will be the device that draws the most power, the power draw is between 0.35 W to 1 W max during continuous transmission. However since we are in monitor mode, the chip could have higher or lower power requirements. We will assume the ESP6288 chip will need approximately 0.5W of power continuously. The NRF chip has a max power draw of 0.3W. We will assume we need approximately 1 W to power the chip and charge the batter considering about 0.2W of loss through heat. This means that our solar panel should average 1W of power over the course of the sunlit day.

The average light intensity (Outdoor average daily light integral) in Illinois can be measured in moles of light per square meter per day, and it ranges from 15 - 50 \( mol \cdot m^{-2} \cdot d^{-1} \) based on a relevant report [18]. The Growth Chamber Handbook [16] estimates approximately 219 \( W \cdot m^2 \approx 1 \text{ mol} \cdot m^{-2} \cdot s^{-1} \). Given a 11\( cm \) by 6\( cm \) solar panel, the surface area is 0.0066\( m^2 \). Which means it will be able to receive between 2.3 W - 7.9 W, assuming that the solar panels are at least 10% efficient.

This estimation (within one order of magnitude) means that our device should be able to work under standard conditions.

### III. Cost and Schedule

#### i. Cost

**i.1 Labor**

Rate: $50/hour

<table>
<thead>
<tr>
<th>Laborer</th>
<th>Total Estimated Hours</th>
<th>Total Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connor</td>
<td>80</td>
<td>$30,000</td>
</tr>
<tr>
<td>Scott</td>
<td>80</td>
<td>$30,000</td>
</tr>
<tr>
<td>Aashish</td>
<td>80</td>
<td>$30,000</td>
</tr>
</tbody>
</table>

**Total: $90,000**

#### i.2 Parts

<table>
<thead>
<tr>
<th>ESP8266 from ebay</th>
<th>Description</th>
<th>Quantity</th>
<th>Cost / item</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>WiFi chip</td>
<td></td>
<td>5</td>
<td>$3.11</td>
<td>$15.55</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ESP8266 from amazon</th>
<th>Description</th>
<th>Quantity</th>
<th>Cost / item</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>WiFi chip</td>
<td></td>
<td>2</td>
<td>$6.5</td>
<td>$13</td>
</tr>
</tbody>
</table>
NRF24L01+PA+LNA
Description: Radio chip for transmitting captured data to drive-by node
Quantity: 5
Cost / item: $2.53
Total: $12.65

ATMEGA328P
Description: Microcontroller for controlled logic between NRF radio chip and ESP8266 WiFi chip
Quantity: 2
Cost / item: $1.88
Total: $3.76

4200mAh Battery, Panasonic 565068
Description: 4200mAh standard Li-ion rechargeable battery
Quantity: 2
Cost / item: $10.88
Total: $21.76

Medium 6V 2W Solar panel - 2.0 Watt (Adafruit)
Description: 11cm x 14cm solar panel
Quantity: 1
Cost / item: $29.00

BQ24168RGET Battery chargers
Description: IC Battery charger for li-ion battery
Quantity: 1
Cost / item: $5.39

JST 2 pin connector
Description: A connector mounted on the PCB that connects the battery
Quantity: 1
Cost / item: $0.75

TPS61091
Description: 3.3V voltage regulator
Quantity: 1
Cost / item: $2.13

Total cost for prototype: $55.67
ii. Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Scott</th>
<th>Connor</th>
<th>Aashish</th>
</tr>
</thead>
<tbody>
<tr>
<td>9/24</td>
<td>Research for the project proposal</td>
<td>Set up gitlab repos, add basic shell and templates for development. Start writing some basic API code for accepting MAC addresses</td>
<td>Research for project proposal</td>
</tr>
<tr>
<td>10/1</td>
<td>Generate dummy data to simulate the passenger count time series</td>
<td>Finish basic API for accepting data from the &quot;drive-by&quot; node</td>
<td>Design and order the PCB</td>
</tr>
<tr>
<td>10/8</td>
<td>Start writing code for data analysis</td>
<td>Finish basic API for devices to use</td>
<td>Test hardware. Test capabilities of WiFi and RF chip</td>
</tr>
<tr>
<td>10/15</td>
<td>Put the node together after the PCB arrives. Start testing and debugging.</td>
<td>Basic frontend to display and manage data. ELK setup</td>
<td>Verify PCB is completed, hardware testing, micro controller code</td>
</tr>
<tr>
<td>10/22</td>
<td>Work on fixes from the previous week</td>
<td>Work on getting multiple devices to talk to each other. Get encrypted communication setup</td>
<td>Get the microcontroller code working. The communications need to be completed at this stage.</td>
</tr>
<tr>
<td>10/29</td>
<td>Arrange several nodes together to simulate potential scenarios and collect data</td>
<td>Start trying to get full devices working and continue working on frontend</td>
<td>Second round of PCB ordering if needed. Testing and debugging.</td>
</tr>
<tr>
<td>11/5</td>
<td>Collect more data. Perform field tests.</td>
<td>Continue working on full system and continue on frontend</td>
<td>Field testing hardware nodes, simulating bus stop and drive-by node</td>
</tr>
<tr>
<td>11/12</td>
<td>Perform analysis on field test results, train machine learning models</td>
<td>Goal to have full working setup with multiple devices gathering data and drive-by node fetching data and uploading to master server</td>
<td>More field testing</td>
</tr>
<tr>
<td>11/19</td>
<td>Prepare and present predictions</td>
<td>Cleanup, final touches, productizing</td>
<td>Final touches</td>
</tr>
<tr>
<td>11/26</td>
<td>Prepare final demo</td>
<td>Prepare final demo</td>
<td>Prepare final demo</td>
</tr>
<tr>
<td>12/3</td>
<td>Prepare final report</td>
<td>Prepare final report</td>
<td>Prepare final report</td>
</tr>
</tbody>
</table>
IV. Safety and Ethics

i. Potential Hazards

Our devices will be outdoors in potentially harsh conditions, we need to make sure the internals of the device are properly insulated from rain, excessive heat, and other harsh conditions that could cause the battery to explode or ignite. We plan to use a rather small lithium ion battery, but even so they can release a large amount of power in a short amount of time if the battery is ruptured. We plan to use a properly insulated container that will be sealed to prevent water or snow to enter.

ii. Data Privacy

Because we are collecting data from phones and associating it with unique identifiers, this gives us the ability to track the movement of these devices, and therefore most likely the owner of the device as well. This information can be used for inappropriate uses if it does not remain anonymous. Because of this, there is an ethical dilemma of how the data will be used or available to companies that will pay for the service. We plan to keep the data anonymous and detached from any PII (personally identifiable information). We plan to look at the data as a whole, not on an individual level. Our goal is to optimize bus routes and give bus companies insights into how their customers are using their service, not identify and track individual people or devices. We also will have each node encrypt all the data it sends using the public key of the server to make sure nobody has access to the data besides the central server.

iii. Digital Security

Security is a big concern with the data we are collecting. We plan to use asynchronous cryptography to have a public key that will encrypt all data that is sent from each individual device. Theoretically anybody could pull the data, but it would be encrypted with a public key. The private key would be stored only on the central master server that will take the collected data, decrypt it, and store it.

iv. General

In addition to the above, our team pledges to follow the IEEE Ethics guidelines [14] as well as the ACM Ethics guidelines [7] as closely as possible.

We address the following terms that are relevant to our project.

1. To hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, and to disclose promptly factors that might endanger the public or the environment

Like previously stated we will properly insulate the physical components of the nodes

2. To avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist

All our data will be viewed on a high level and not on the granularity of the specific consumer. This will prevent an individual from being targeted or tracked from the bus company.

6. To maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations
We will make sure that we keep up to date with the most recent technological updates and security issues. All technicians shall be trained before handling the electronics. All our data will be viewed on a high level and not on the granularity of the specific consumer. This will prevent an individual from being targeted or tracked from the bus company.

9. To avoid injuring others, their property, reputation, or employment by false or malicious action

All of our data will be anonymized and not be looked at on an individual level.

v. Current Competition

Here is a link to devices that track wifi devices for commercial clients. It helps detect rouge devices and keeps track of potential wifi enabled threats. It is however in a different market and industry than the one we are currently targeting. They focus on more commercial grade office building types, not the data aggregation that we do.

https://www.accuware.com/products/locate-WiFi-devices/

REFERENCES


