JP Chickadee Project

Author: Ryan Frazier

Overview - What is it?

Dr. Alec Lindsay (Biology Dept.) asked Computer Science students if we would be interested in upgrading their current system with the idea of autonomous data collection.

- Joined the group in Fall 2017/18
  - Project started in winter 2016/17
- Five active group members
  - Nick Potyok, Jon Lefler, Dan Gaechter, Jerry Conner, Ryan Frazier
- 2018 Student Technology Innovation Award

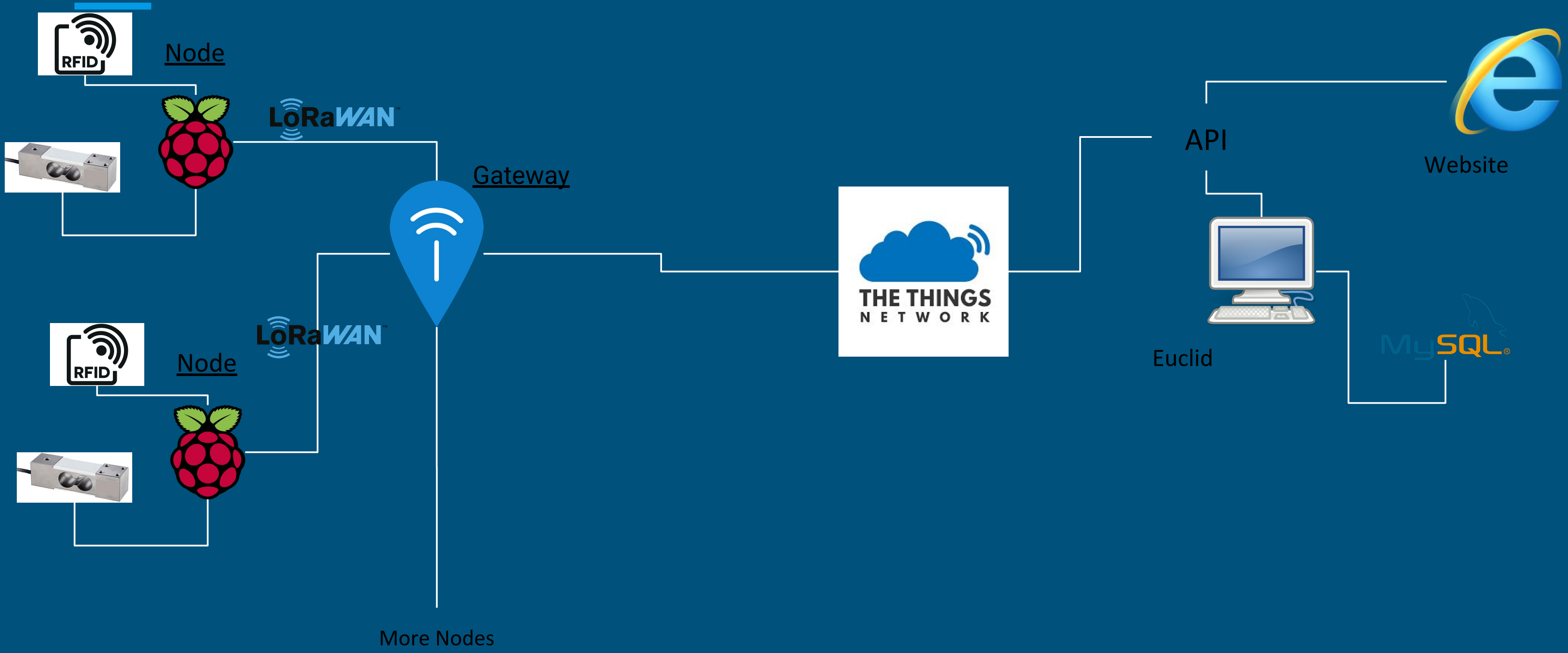
Task Objective

Analyze, design, develop, test, and implement a ‘smart’ bird feeder.

- RFID tag and weight of individual chickadees
- Battery life of at least one week
- Capability of withstanding U.P. climate
- Save collected data into a MySQL database
- Publicly accessible website displaying the data

Data will be used by the Biology Dept. to determine eating, movement and dominance behaviors of chickadees in the local area.

Data Flow


Node
RFID
LoRaWAN
Gateway
Node
RFID
LoRaWAN
More Nodes
THE THINGS NETWORK
API
Website
Euclid
MySQL

Data Flow

‘ Refer to white board ‘

1. Once a chickadee lands; the RFID tag and timestamp are saved into a log file until a predetermined quota has been reached

2. Transmission begins
   a. Concurrently with data collection
   b. A JOIN request is sent to the LoRa Gateway from our Node
   c. An acknowledgement is sent back to the Node and a connection is formed
   d. The log file is sent line by line until empty
   e. Packets are encrypted using LoRaWAN default encryption

Data Flow cont.

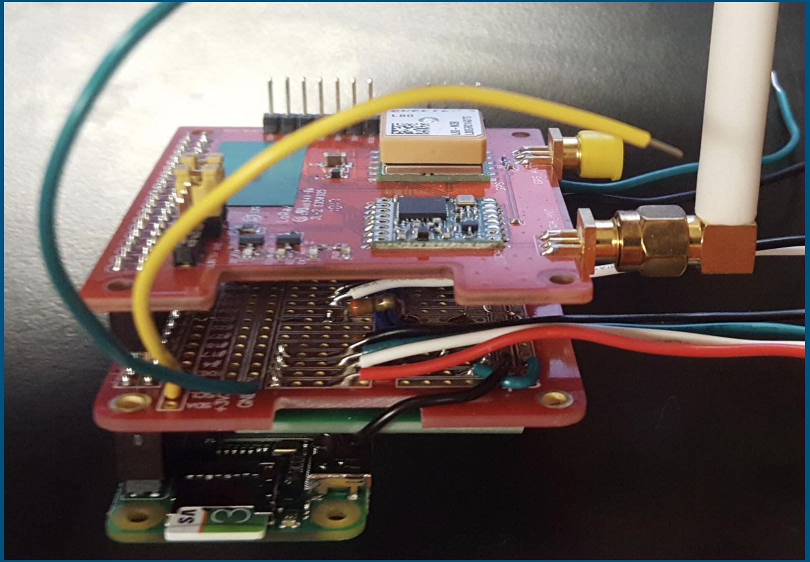
3. LoRa Gateway receives the incoming data. Is then forwarded to *The Things Network*.
    a. TTN decrypts the data and sends it to my API via POST request.

4. API receives the data, eliminates duplicates, and inserts it into the Database.

All of this happens concurrently, meaning that I am able to continue with capturing chickadee landings and building new log files.

Hardware - Modular design practices

- Electrical: Individual Node Materials
  - Raspberry Pi Zero W
  - Raspberry Pi prototype board
  - Dragino LoRa/GPS hat
  - LoRa (915Mhz) Glass-fiber antenna
  - RFID/Weight sensors



- Non-Electrical: Bird Feeder Materials
  - 5 gallon bucket
  - 12 foot hollow metal rod
  - 3 foot PVC tube
  - Many custom designed and 3D printed components

Hardware - Electrical

- A ‘complete’ Raspberry Pi needed additional micro-processors in order to be fully functional.
  - DC/DC converter
  - HX711 Load-Cell
  - RFID UART chip
- Battery size
  - 12 volt 12 amp hour

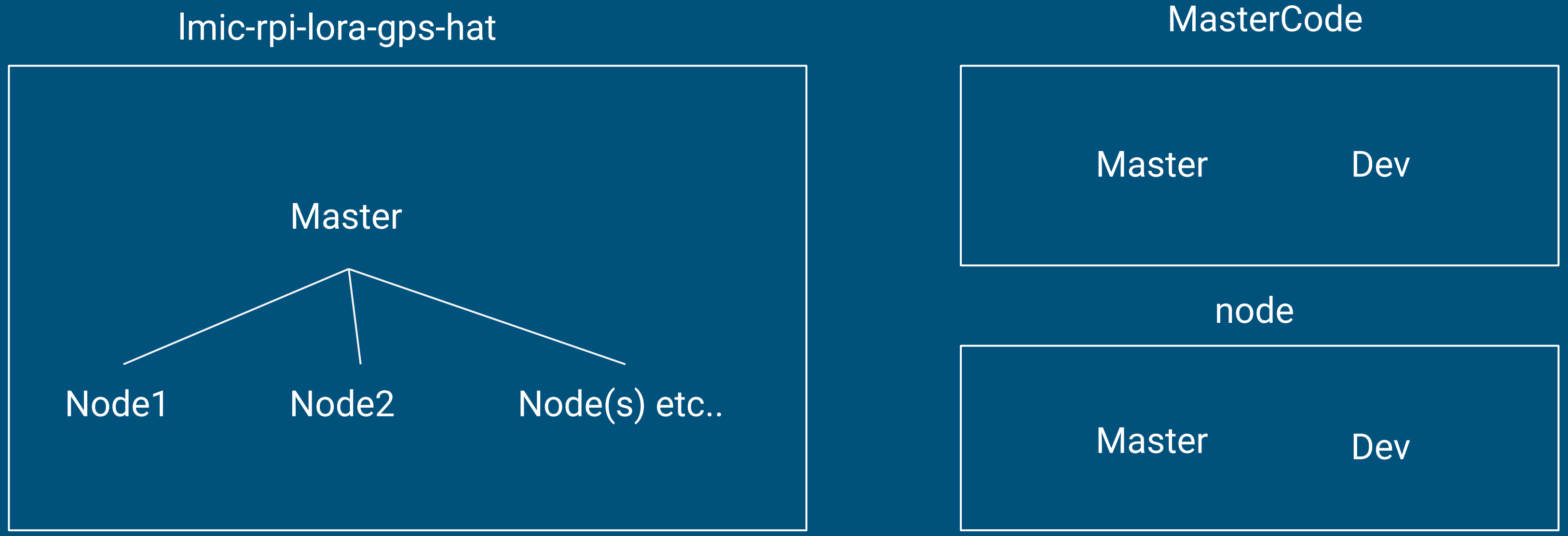
Software

- LoRaWAN - Data Transmission
  - Spec 1.0 released in 2015.
  - Network layer protocol for communication between gateways and end-node devices.
  - Chirp-Spread Spectrum
- Programming Languages
  - Python 3
    - Data collection and processing.
    - API integration.
  - C
    - Node - LMIC library
    - Gateway - RAK831

Software cont.

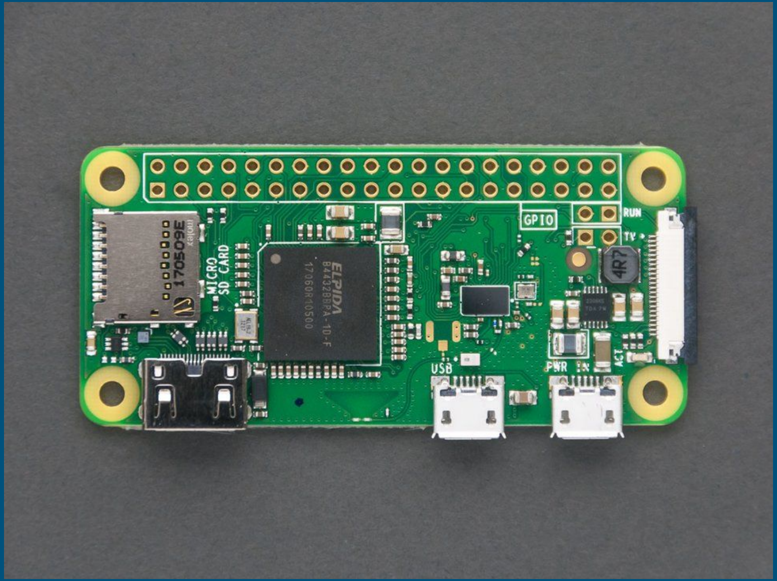
- Collaboration tools
  - Trello - Task management system
    - Create tasks
    - Time management
    - Workflow reasoning
  - GitHub - Source Code management
    - All source code centrally located
    - Allowed us to implement an easy Node Master image scheme
  - Telegram - Day-to-day communication
    - Instant messenger

GitHub Scheme for Nodes



Testing - What worked and what didn't

- Battery conservation
  - LTE vs. LoRa
  - Renewable energies
  - Intermittent sending
- Hardware Changes
  - LoRa -> LTE -> LoRa
  - Raspberry Pi 3b vs. Raspberry Pi Zero W
- Software Changes
  - Load-Cell Activation vs. RFID Activation



Testing cont.

Some elements were not able to be included in the first iteration of our product

- Accurate Weight sensor
  - Ideally the weight sensor should activate the RFID sensor
  - Exceeding Encoding rate
  - CPU bug

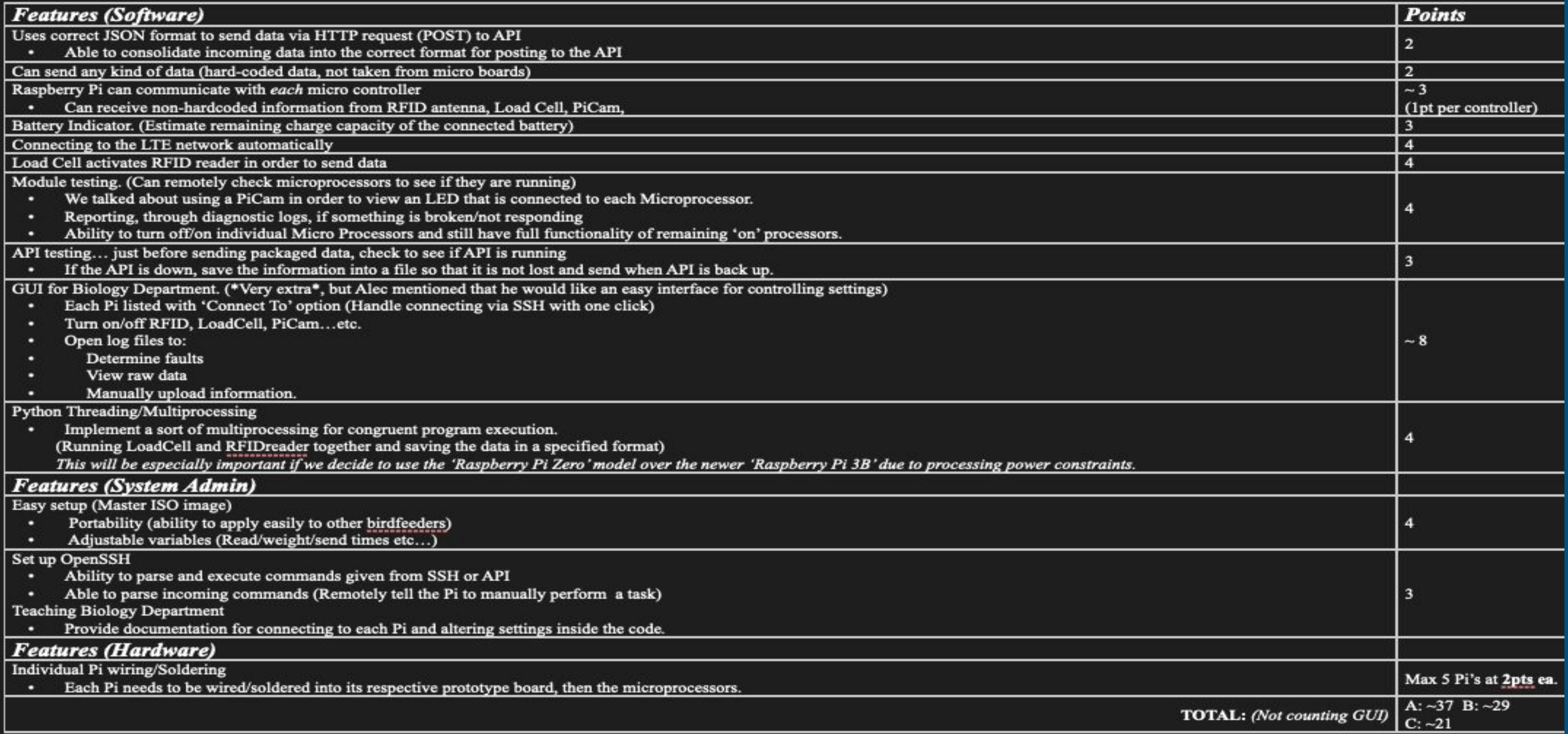
- Immediate data transmission
  - Too much battery consumption

Conclusion

- We are able to push out our first iteration product by the deadline
- Entailed much more learning than what I had originally planned
- We have upgraded hardware and software for future releases
  - Battery Monitor
  - Load-Cell Activation
  - LTE Raspberry Pi Hat
  - Battery Day/Night cycle

Product Demonstration

Questions?



Features (Software)	***Points***
Uses correct JSON format to send data via HTTP request (POST) to API • Able to consolidate incoming data into the correct format for posting to the API	2
Can send any kind of data (hard-coded data, not taken from micro boards)	2
Raspberry Pi can communicate with *each* micro controller • Can receive non-hardcoded information from RFID antenna, Load Cell, PiCam,	~ 3 (1pt per controller)
Battery Indicator. (Estimate remaining charge capacity of the connected battery)	3
Connecting to the LTE network automatically	4
Load Cell activates RFID reader in order to send data	4
Module testing. (Can remotely check microprocessors to see if they are running) • We talked about using a PiCam in order to view an LED that is connected to each Microprocessor. • Reporting, through diagnostic logs, if something is broken/not responding • Ability to turn off/on individual Micro Processors and still have full functionality of remaining 'on' processors.	4
API testing… just before sending packaged data, check to see if API is running • If the API is down, save the information into a file so that it is not lost and send when API is back up.	3
GUI for Biology Department. (*Very extra*, but Alec mentioned that he would like an easy interface for controlling settings) • Each Pi listed with 'Connect To' option (Handle connecting via SSH with one click) • Turn on/off RFID, LoadCell, PiCam…etc. • Open log files to: • Determine faults • View raw data • Manually upload information.	~ 8
Python Threading/Multiprocessing • Implement a sort of multiprocessing for congruent program execution. (Running LoadCell and RFIDreader together and saving the data in a specified format) *This will be especially important if we decide to use the 'Raspberry Pi Zero' model over the newer 'Raspberry Pi 3B' due to processing power constraints.*	4
Features (System Admin)	
Easy setup (Master ISO image) • Portability (ability to apply easily to other birdfeeders) • Adjustable variables (Read/weight/send times etc…)	4
Set up OpenSSH • Ability to parse and execute commands given from SSH or API • Able to parse incoming commands (Remotely tell the Pi to manually perform a task) Teaching Biology Department • Provide documentation for connecting to each Pi and altering settings inside the code.	3
Features (Hardware)	
Individual Pi wiring/Soldering • Each Pi needs to be wired/soldered into its respective prototype board, then the microprocessors.	Max 5 Pi's at **2pts ea**.
TOTAL: *(Not counting GUI)*	A: ~37 B: ~29 C: ~21