A Parking Guidance and Information System for TinyOS

CSE 521S Final Presentation

By Matthew Lindsay, Patrick McBryde, and Michael Schultz

Project (Re-)Introduction

"However, [the World Wide Web] could start a revolution in information access." (Tim Berners-Lee, 1991)

- When driving to a new location, what do you want most?
  - How do I get there/what does it look like? Google Maps.
  - Where should I park when I get there? Hope.

- Parking can be highly inefficient and extremely frustrating
  - A driver should be able to find a spot easily
  - No aimless circling of a parking lot

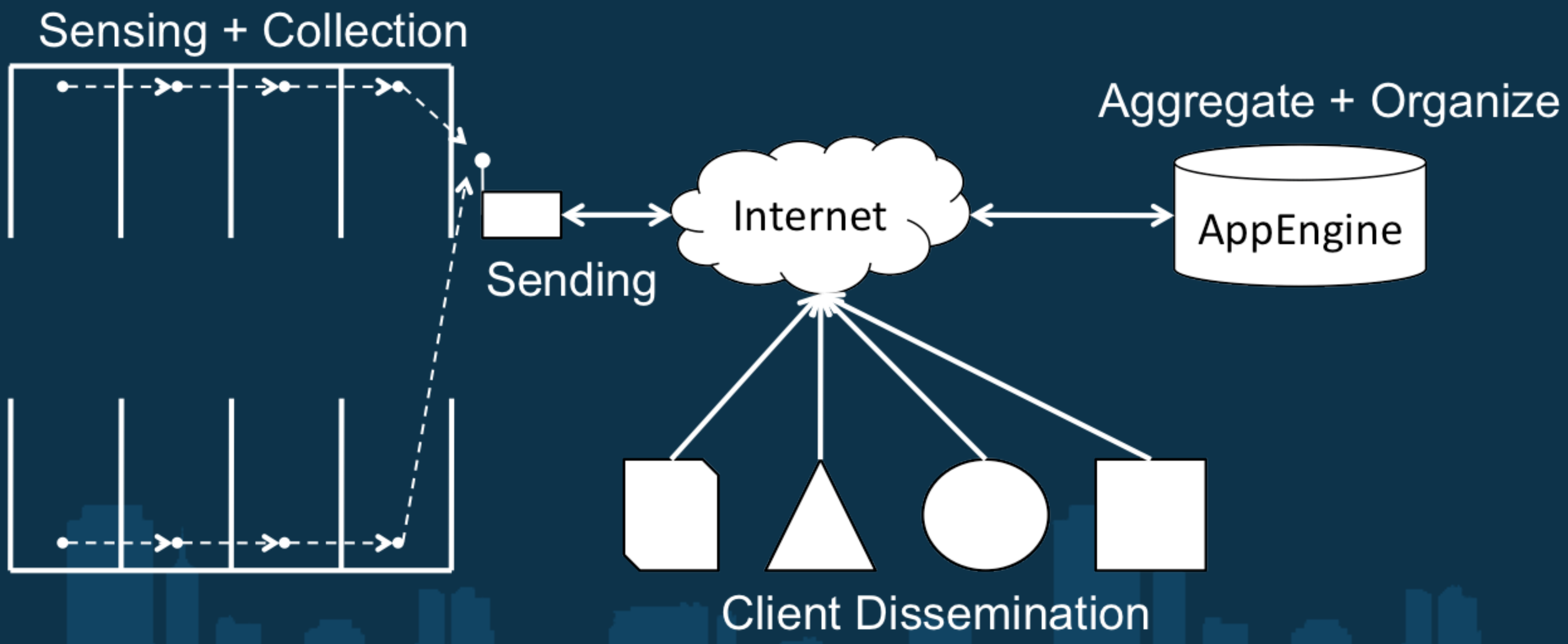
- We can use today's technology to improve this situation!

Project Goals

- Develop a WSN to monitor each parking spaces in an area
  - Take advantage of low(-ish) cost TelosB motes
  - Use external connector to connect sensor(s)
- Reliable data delivery over wireless sensor network
  - Using Collection Tree Protocol
- Aggregate data for visitors and management to use
  - Web interface + published JSON format for queries

Design

- High-level view of system design



Hardware

- TelosB
  - Ultra low power
  - IEEE 802.15.4 compliant
  - Integrated sensors

- LV-MaxSonar-EZ1 - Ultrasonic Rangefinder
  - Supply voltage 2.5V to 5.5V
  - Detects objects from 6 inches out to 254 inches with 1 inch resolution (0-6 inches range as 6 inches)
  - Output formats include pulse width, analog voltage, and serial digital

- Web Connected *nix computer
  - Macbook Pro

Software

- Motes
  - Sensor
    - TinyOS
    - Collects data from multiple sensors.
    - Updates base-station every 15 Seconds.
  - Base-Station
    - BaseStation app included with TinyOS install
- Base-Station
  - The base-station collects sensor data from the wireless sensor network and transfers what is needed to the AppEngine. This includes occupancy, light levels, and temperature.
  - Uses JSON to wrap sensor data and CURL to perform HTTP put to server.

Backend Aggregator

- AppEngine uses Google's infrastructure for web apps
- Allows our application to scale well (within constraints)
- When data comes in ...
  - extract it and update the space and lot stats
  - log changes in the parking lot and parking space
  - tell client success or failure
- When data goes out ...
  - get data from all spaces in a lot
  - find lots near to target lot
  - chart historical data (by request)

Information is money!

Frontend Display(s)

- Status of selected lot + map of nearby (~2 miles) lots
  - Markers let you see status of those lots too
- Chart of average use over time interval for date range
  - i.e., from 5am to 7am in the last week
- Exported in JSON format for integration with other apps
  - Lot information + Per space information

(You'll see these in the demo!)

Experimentation

- To verify the system works we performed a test using a real vehicle and parking space.
  - Link to video of system test.

- What could be improved.
  - Vehicle detection: increase the ultrasonic rangefinder supply voltage and add a second rangefinder sensor.
  - Rangefinder sampling decreased to save power.
  - Power consumption by the TelosB could be greatly improved.
- Day long simulation to test backend and chart interface
  - Random activity every 60--120 seconds
  - Increased activity from 6:00-7:00, 12:00-13:00, and 18:00-19:00

Demo

Related Work

- Streetline
  - San Francisco startup
  - Began deploy with SF*park* in Summer 2010
  - Results pending
  - Great for us: validates the work and creates the market

- Signal-Park
  - Older, more traditional approach
  - Sonar detector per space in garage
  - Wired to central computer for collection
  - Not really publicly available information

Conclusions

- We created a functioning parking guidance and information system for TinyOS
    - Hardware
    - Scalable Infrastructure
        - RESTful API for apps

- With more resources and time this could be a viably solution for existing parking areas

Questions

Resources

LV-MaxSonar-EZ1 Data Sheet