CarTel: A Distributed Mobile Sensor Computing System

But before we start...

• Categories of Sensor Networks
  – Static
  – Mobile
Mobile Sensor Networks

• Motivation
  – Need to sense the environment at much finer fidelity and higher scale.

• Technology push

• Application pull
Applications

- Traffic Monitoring
- Environmental monitoring
- Civil infrastructure monitoring
- Automotive diagnostics
- Geo-imaging
- Data muling
What does CarTel offer?

• a mobile sensor computing system
  – to collect, process, deliver, and visualize data from a collection of remote, mobile, and intermittently connected nodes.
  – Provides a reusable software platform to build mobile sensing applications.
CarTel System Architecture
Design Goals

• Provide a simple programming interface

• Handle large amounts of heterogeneous sensor data

• Handle intermittent connectivity
CarTel Software Architecture
CarTel Components

- **Portal**
  - Centralized
  - Visual user interface
- **ICEdb**
  - Delay-tolerant query processing
- **CafNet**
  - Delay-tolerant network stack
IceDB (Intermittently connected DB)

• Centralized declarative queries
  – Executed in distributed fashion by mobile nodes

• IceDB server and IceDB remote
IceDB (Intermittently connected DB)

• Data Model
  – Handle large amounts of heterogeneous data
  – Solution: Meta-data package describing attributes of sensor called *adapter*
  – Create local tables for sensor readings
  – Acquire tuples from sensor
  – Parse sensor readings
IceDB (Intermittently connected DB)

- Continuous Query Model
  - SQL Extensions for
    - Continuous Queries using the *rate* clause

```sql
SELECT carid, traceid, time, location FROM gps
WHERE gps.time BETWEEN now()-1 mins AND now()
  RATE 5 mins
```

- Handling Intermittent Connectivity
  - Using Prioritization (Local and Global)
Contd...

• Local Prioritization
  – PRORITY
  – DELIVERY ORDER
  – User-defined function

• Global Prioritization
  – SUMMARIZE AS
CafNet: Delay-Tolerant Network Stack

- **CTL**
  - Callback mechanism for dynamic prioritization

- **CNL**
  - Routing (best effort)

- **MAL**
  - Media-independent interface
  - Two types of callbacks

---

**Diagram: Delay-Tolerant Network Stack**

- **Device Driver**
  - To **Transport Layer**
    - **Transport Layer**
      - Registers data to be transmitted
      - Delivers incoming data
      - Requests data from the application
      - Notifies application of successful delivery
  - To **Network Layer**
    - **Network Layer**
      - Notifies transport layer of free buffers
      - Schedules data for transmission
      - Selects routes
      - Buffers data for transmission
  - To **Mule Adaptation Layer**
    - **Mule Adaptation Layer**
      - Provides uniform neighbor discovery
  - To **Device Driver**
Portal

• Portal Framework
  – To build CarTel Applications
  – Navigate sensor data using web-based application

• IceDB

• Data visualization interface
  – Trace – data segmentation abstraction
  – For searching traces using spatial queries
  – Overlaying geographical attributes on a map
Spatial Queries
Overlaying geographic attributes
Case Study: Road Traffic Analysis

- For data collection,
  - Continuous query
  - GPS adapter

- Commute time analysis
  - Observe speed and delay for the particular trace

<table>
<thead>
<tr>
<th>Route</th>
<th>Avg. Dist</th>
<th>Avg. Time</th>
<th>Std – dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freeway</td>
<td>9.94 miles</td>
<td>19:52</td>
<td>02:14</td>
</tr>
<tr>
<td>City Streets</td>
<td>9.83 miles</td>
<td>29:34</td>
<td>02:19</td>
</tr>
<tr>
<td>Frontage Road</td>
<td>9.27 miles</td>
<td>31:51</td>
<td>03:54</td>
</tr>
</tbody>
</table>
Case Study: Road Traffic Analysis

Traffic Hotspots

<table>
<thead>
<tr>
<th>Rank</th>
<th>Avg MPH</th>
<th>Std-dev</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25.3 mph</td>
<td>26.7 mph</td>
<td>176</td>
</tr>
<tr>
<td>2</td>
<td>29.3 mph</td>
<td>24.6 mph</td>
<td>315</td>
</tr>
<tr>
<td>3</td>
<td>33.0 mph</td>
<td>22.8 mph</td>
<td>267</td>
</tr>
<tr>
<td>4</td>
<td>39.4 mph</td>
<td>22.5 mph</td>
<td>245</td>
</tr>
<tr>
<td>5</td>
<td>15.0 mph</td>
<td>18.0 mph</td>
<td>729</td>
</tr>
<tr>
<td>6</td>
<td>32.9 mph</td>
<td>17.5 mph</td>
<td>187</td>
</tr>
<tr>
<td>7</td>
<td>20.4 mph</td>
<td>17.1 mph</td>
<td>535</td>
</tr>
<tr>
<td>8</td>
<td>34.2 mph</td>
<td>16.2 mph</td>
<td>53</td>
</tr>
<tr>
<td>9</td>
<td>17.9 mph</td>
<td>15.9 mph</td>
<td>365</td>
</tr>
<tr>
<td>10</td>
<td>17.2 mph</td>
<td>15.3 mph</td>
<td>313</td>
</tr>
</tbody>
</table>

* To zoom in on the map, use the +/- buttons, or position the cursor over the map and press z to define the zoom region and x to zoom out.
Case Study: Wide-area Wi-Fi Measurements

- Experimental Setup
  - 6 cars equipped with CarTel box and software
  - Driving normally in parts of the Boston area
  - ~300 drive hours

- Fast scanning of WiFi access points, caching of AP parameters to speed up connection establishment

- Careful (small and unobtrusive) TCP data transfers to measure throughput, latency, loss rates

- Track performance statistics: connection durations, throughput distributions, etc
Wide-area Wi-Fi Measurements

• Results:
  – Total Number of AP’s discovered – 32k
  – Total Number of associations – 5k
  – Mean Association Duration – 25s
  – Mean time between connection to Internet – 260s
  – Median Upload Throughput – 30KBytes/s
Wide-area Wi-Fi Measurement

**Connection Duration Distribution**

Typical connection durations at vehicular speeds are a few seconds long (incl. optimizations for fast scanning and AP parameter caching)
Pros and Cons

• Framework for building mobile applications
• Provision of simple programming interface
• Dynamic Proritization
• Opportunistic Connectivity options
• Visual interface to specify visual queries
Relevance to SOS Project