CarTel: A Distributed Mobile Sensor Computing System

by Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen Miu, Eugene Shih, Hari Balakrishnan and Samuel Madden

Presenter: Ionut Trestian
CarTel - Introduction

- Mobile Sensor Computing System
- Uses CARTEL nodes – mobile embedded computer coupled to sensors.

Functioning:
- Nodes gather sensor readings
- Local data processing (prioritization, summarization …)
- Deliver to a central portal
- At portal, data can be analyzed and visualized
CarTel - Motivation

- Emerging of mobile sensor networks
- “Technology push” thousands of sensors available embedded in computers and phones.
- “Application pull” – the need for cheap monitoring applications.
- Mobile solutions have advantages over static ones
  - Bigger geographical areas covered
  - No need for static infrastructure put in place
CarTel - Applications

- Environmental monitoring (chemical and pollution)
- Civil infrastructure monitoring (vibrations and other sensors to monitor state of roads, bridges)
- Automotive diagnostics (obtaining information from a vehicle’s on-board sensors)
- Geo-imaging (capturing images from locations)
- Data muling (using cars and people as delivery networks)
CarTel – Overview & Contributions

• Reusable software platform that can be used to build many mobile sensing applications.

• High-level goals:
  • Provide a simple programming interface
    • Apps can be written as web apps – distribution and mobility are hidden)
  • Handle large amounts of heterogeneous sensor data
    • Any kind of sensors could be plugged,
    • Large amounts of data can be buffered and processed on nodes
  • Handle intermittent connectivity
    • Opportunistic wireless, eg WiFi passing by and therefore intermittent
    • Other storage devices (USB keys or flash drives) used as data mules
CarTel - Components

- ICEDB (intermittently connected database)
  - A delay-tolerant continuous query processor.
- CafNet (carry-and-forward network)
  - a delay-tolerant network stack
- Portal
  - Sink of the data
  - Data visualization
CarTel – Portal programming model

- Applications running on the portal issue continuous queries using an API exported by ICEDB

- Queries specify:
  - Sensor data to be acquired (rate, sampling, filtering ..)
  - Query results stream across an intermittently connected network and populate a relational database at the portal
  - Applications issue snapshot queries that run on whatever data is currently available.
CarTel - Contributions

• CarTel builds on previous work on mobile systems, sensor data management and DTNs.

• Therefore the main contribution is the synthesis of ideas.

• Other contributions:
  • Expanding the notion of continuous queries to handle intermittent connectivity.
  • Enabling modular upgrades to integrate new sensors and data types using adapters.
  • The CafNet “carry-and-forward” DTN stack that delivers data in intermittently connected environments (using callbacks across all layers – allows the sender to dynamically prioritize data).
  • Design of portal and visualization interface
ICEDB

- ICEDB distributes query execution and result delivery between the ICEDB server running on the portal and the remote nodes.
- Server
  - Maintains a list of continuous queries submitted by the applications and pushes them to the remote nodes using CafNet
- Nodes
  - Run ICEDB remote to process the sensor data and return the query results using CafNet
ICEDB – Data types

- Supports heterogeneous data types and makes the addition and removal of sensors easy
- Uses a meta-data package called an adapter for handling new sensor types (basically a script)
  - Automatically create local tables to store sensor readings
  - Acquire tuples from the sensor
  - Parse sensor readings to store them in the database and process them by subsequent queries.
- Adapter attributes:
  - ID, name, Type (push or pull data), Rate (for pull), Forwarding flag (raw data delivered to the portal), Schema (name-type pairs, currently PostgreSQL data types), Priority.
- Currently, adapters for node diagnostics, GPS receiver, OBD-II interface, WiFi interface and digital camera.
ICEDB – Continuous Query Model

- usage of RATE clause
- `SELECT carid, traceid, time, loc FROM gps WHERE gps.time BETWEEN now()-1 mins and now() RATE 5 mins`
- clock synchronization via NTP
- Prioritized data delivery
- Prioritization Schemes
  - *Local Prioritization*
    - Cannot receive feedback from portal
  - *Global Prioritization*
ICEDB – Prioritization Schemes

- **Local Prioritization**
  - Uses two language extensions for specifying the transmission order of results: PRIORITY and DELIVERY ORDER.
  - PRIORITY assigns a numeric priority to the query’s result buffer
  - DELIVERY ORDER is the further level of granularity which goes into the query and specifies the order of the attributes.

- **Global Prioritization**
  - Expressed using the SUMMARIZE AS clause which specifies a query that will compute a summary.
  - Mainly based on aggregates
CafNet - Overview

- Network stack for delay tolerant communication
- Uses a message-oriented data transmission and reception API not a stream-oriented connection such as TCP.
- All nodes are named using globally unique flat identifiers that don’t embed any topological or organizational semantics (hashes).
- Network stack does no buffering
  - Applications buffer
- CafNet informs the application when connection is available
- Application can decide what data to send “at the last moment” instead of committing earlier to the data in advance by sending to the net layer.
CafNet – Basic Stack

- CTL can be implemented as a library or separate process which communicates with the application using RPC.
- CTL provides optional delivery confirmation service by setting a flag (NONE or END2END - retransmissions).
- CNL provides a send function.
- Peer discovery happens at the MAL because these mechanisms are media-specific.
CafNet – Optimizations and Enhancements

• Design is “pure”, has no network buffering but performs poorly when average duration of connectivity is not significantly larger than application package time.

• Solution: introduce a small amount of buffering in the stack (CNL).

• Setting CNL buffer large hinders application ability to prioritize data.

• Solution: Let application set the size of the CNL buffer.
The Portal

- Central repository
- Three main components
  - Portal framework
  - ICEDB server to retrieve sensor data
  - Data visualization library to display geo-coded attributes.
- Provides privacy to Cartel Users
- Applications issue continuous queries and view the results. Once issued, the queries are pushed to the nodes. Intermediate results are displayed.
- Uses *Trace*: data collected in a single trip
- Provides two class of functions
  - interface to search traces using spatial queries
  - interface to overlay geographic attributes on a map, for any trace
The Portal

![Map with route highlighted]
The Portal
Case studies – Road Traffic Monitoring

- Road traffic analysis: GPS adapter and continuous queries to keep track of routes and delays
  - Commute time analysis

<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>

- Traffic hot spot heuristics
- Image acquisition
Traffic hot spots – Boston area

<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>16.0 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>635</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.
# Traffic Hot Spots – Seattle Area

## 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>40.7 mph</td>
<td>28.1 mph</td>
<td>69</td>
</tr>
<tr>
<td>2</td>
<td>19.3 mph</td>
<td>22.1 mph</td>
<td>91</td>
</tr>
<tr>
<td>3</td>
<td>56.0 mph</td>
<td>21.9 mph</td>
<td>39</td>
</tr>
<tr>
<td>4</td>
<td>23.2 mph</td>
<td>18.5 mph</td>
<td>52</td>
</tr>
<tr>
<td>5</td>
<td>16.5 mph</td>
<td>17.7 mph</td>
<td>51</td>
</tr>
<tr>
<td>6</td>
<td>24.9 mph</td>
<td>17.3 mph</td>
<td>197</td>
</tr>
<tr>
<td>7</td>
<td>11.8 mph</td>
<td>15.7 mph</td>
<td>111</td>
</tr>
<tr>
<td>8</td>
<td>25.7 mph</td>
<td>15.6 mph</td>
<td>186</td>
</tr>
<tr>
<td>9</td>
<td>16.4 mph</td>
<td>15.2 mph</td>
<td>128</td>
</tr>
<tr>
<td>10</td>
<td>14.9 mph</td>
<td>14.9 mph</td>
<td>2770</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.
Image acquisition
Wide area WiFi Measurements

- 65% of on-line households have installed Wi-Fi access points Jupiter Research.
- WiFi sharing?
- Collect over 290 “drive hours” of data about Wi-Fi connectivity in urban environments.
- Data:
  - Wi-Fi scans – reports of nearby APs
  - Wi-Fi associations - attempts to establish link-layer connectivity with APs
  - IP address acquisitions – attempts to acquire an IP address using DHCP.
  - Ping and upload statistics.
- 32000 Aps, 5000 associations, 2000 gave IPs.

<table>
<thead>
<tr>
<th>Mean association duration</th>
<th>25 seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean time between connections to Internet</td>
<td>260 seconds</td>
</tr>
<tr>
<td>Median upload throughput</td>
<td>30 KBytes/s</td>
</tr>
</tbody>
</table>
Automotive Diagnostics

- Federal Test Procedure (FTP75) rates cars for fuel economy and emission levels.
- Test criticized so EPA introduced US06 that has harder acceleration and moving speeds (not used for fuel economy purposes).
- OBD Data
  - Emissions, engine status, fuel consumption, troubleshooting codes, engine load, fuel consumption and pressure, engine RPMs, engine timing, air intake temperature, engine throttle position and oxygen sensor status.
Conclusions & Future work

- Hundreds of millions of automobiles (to which embedded computers can be attached) and over a billion-phone equipped people. HUGE Potential
- Cartel provides software to collect, process, deliver and visualize data from sensors located on mobile devices to a portal.
- Future work:
  - Aggregate information across users.
  - Process and analyze more data obtained from OBD sensors.
  - Incorporate simple routing algorithms.
  - More queries and users.