Extracting High-Level Context from Low-Level Sensor Data

Yanyong Zhang
Bernhard Firner

Rutgers University, Winlab

May 10, 2011
Collecting Data Is Easy

- Many existing sensor networks have been deployed
- Often these networks collect a dedicated set of data for one environment
  - datacenter
  - hospital
  - wildlife habitat
- Interfacing to sensors is becoming easier
  - Well designed and tested open source protocols and software
  - IPv6 on sensor nodes
Collecting Data Is Easy

- Many existing sensor networks have been deployed
- Often these networks collect a dedicated set of data for one environment
  - datacenter
  - hospital
  - wildlife habitat
- Interfacing to sensors is becoming easier
  - Well designed and tested open source protocols and software
  - IPv6 on sensor nodes
- Is this all we want?
Our Real Goal Is to Extract High-Level Semantic Knowledge

- End-users are interested in high-level semantic events, not raw data
  - “It’s raining” rather than “humidity is 77%.”
- We need to fuse data and semantic events together to discover higher-level semantic knowledge
  - “It’s raining” and “the window is open” fuse into an important event when they occur simultaneously
Next Generation Sensor Networks

- Sensor networks need to extract higher-level semantic information for users
- We need a new kind of system
  - That can store and disseminate higher-level information
  - Can still support a wide variety of hardware layers and sensing information.
  - Can adapt to changing user needs
Sensor networks need to extract higher-level semantic information for users

We need a new kind of system

• That can store and disseminate higher-level information
• Can still support a wide variety of hardware layers and sensing information.
• Can adapt to changing user needs
• Must be general-purpose
  • Supports many kinds of sensor and radio hardware
  • Supports many kinds of applications
Next Generation Sensor Networks

- Sensor networks need to extract higher-level semantic information for users
- We need a new kind of system
  - That can store and disseminate higher-level information
  - Can still support a wide variety of hardware layers and sensing information.
  - Can adapt to changing user needs
  - Must be general-purpose
    - Supports many kinds of sensor and radio hardware
    - Supports many kinds of applications
  - Must be extensible
    - Can modify a running system with new hardware and software
Introducing Octopus

• New open source platform, *Octopus*
• Mimics IP’s hourglass shape
  • Wide variety of sensors on the bottom
  • Wide variety of semantic information on the top
• Supports “live” additions of new hardware/software
• Split into layers
  • Cross-layer communication occurs over TCP
Supporting General Hardware and Information

- Sensor information is accepted through an aggregation layer
  - Sensor hardware is separated from higher layers
- Data is handled in a general way
  - URI strings accompany binary data
  - URIs help analysis software identify data
  - User applications request data by URI
  - The rest of the system treats all data as binary information
Data Analysis Layer

- Impossible to anticipate every type of data
- Solution - use a plug-in system for data analysis
  - Plug-ins are called solvers
  - Solvers run independently
  - User-feedback can be quickly incorporated into new solvers
- Solvers subscribe to sensor data from the aggregation layer
- Solvers generate solutions from this data
- Solutions are sent to the distribution layer
  - Users request data from the distribution layer
  - Users never interact with solvers directly
Creating Higher-Level Semantic Knowledge

- Higher-level semantic knowledge is built from lower-level information
- Solvers can request previous solutions from the distribution layer
  - Simplifies fusing lower-level information into higher-level results
  - Creates a dependency graph of solvers
  - Lower-level solvers do not need to know what uses their data
Deploying an Octopus System

- We have a long-term deployment in Winlab
  - Shows a status map (like a “Marauder’s Map”)
  - Tracks other events of interest, such as coffee brews
- Used pipsqueak radio tags
  - Lifetimes of about a year on a coin cell battery
  - Lab has many available
  - Small enough to put on many objects
There are several kinds of sensors deployed

1. **Switch sensors**
   - Pressure activated chair-use sensors
   - Magnetic door switches
Hardware Sensors

There are several kinds of sensors deployed:

1. **Switch sensors**
   - Pressure activated chair-use sensors
   - Magnetic door switches

2. **Power sensors**

   - For RSS-based localization
   - For RSS-based mobility detection
There are several kinds of sensors deployed:

1. **Switch sensors**
   - Pressure activated chair-use sensors
   - Magnetic door switches

2. **Power sensors**

3. **Temperature sensors**

   ![Image of a coffee maker with sensors labeled as Mobility Sensor and Temperature Sensor]
There are several kinds of sensors deployed

1. **Switch sensors**
   - Pressure activated chair-use sensors
   - Magnetic door switches

2. **Power sensors**

3. **Temperature sensors**

4. **Radio only sensors**
   - For RSS-based localization
   - For RSS-based mobility detection
Higher Level Information

- Sensor data is turned into higher-level knowledge.
  - Localization: Where is the tape?
  - Propped Door: Did someone leave the back door propped open?
  - Coffee Brewing: Is there any fresh coffee?
  - Room In Use: Is the conference room in use?
  - Calendar Events: Are there upcoming talks?
  - Tea Time: Is everyone having tea?
Many Solvers Depended Upon Other Solvers
Adding New Solvers is Easy

- Example: discover when a talk starts
- We just need to connect to the *room in use* and *calendar events* solvers.
  - Reusing other solvers makes this simple
  - Don’t need to worry about raw sensor data
    - Chair sensor switches
    - Locations of chairs
    - Power consumption of the projector
Information is Available in Your Browser

Recent events:

- Wed Apr 6 2011 03:18:51 PM: Coffee Pot has coffee brewing
- Wed Apr 6 2011 03:26:46 PM: winlab.mugs.Ben is at teatime
- Wed Apr 6 2011 03:26:46 PM: winlab.mugs.James is at teatime
- Wed Apr 6 2011 03:26:46 PM: winlab.mugs.Rich is at teatime
- Wed Apr 6 2011 03:26:46 PM: teatime is happening
And in Twitter

What's happening?

@summerfrog2
1 cup poured --message at Fri Mar 25 10:59:13 EDT 2011
25 Mar

@summerfrog2
Coffee is ready! --message at Fri Mar 25 10:52:46 EDT 2011
25 Mar via LabTesting Favorite Reply Delete

@summerfrog2
Coffee is ready! --message at Fri Mar 25 10:46:03 EDT 2011
25 Mar

@summerfrog2
3 cups poured --message at Fri Mar 25 10:51:18 EDT 2011
25 Mar

@summerfrog2
2 cups poured --message at Fri Mar 25 10:50:07 EDT 2011
25 Mar

@summerfrog2
1 cup poured --message at Fri Mar 25 10:47:26 EDT 2011
25 Mar

@summerfrog2
Coffee is ready! --message at Fri Mar 25 10:59:13 EDT 2011
25 Mar
Runtime Statistics

- System uptime over 5 months
- More than 40 transmitters and sensors
- 17 Receivers
  - Over 600 packets processed per second
- Low latency performance
  - About 1-second latency (the sensor transmission interval)
Conclusions

• High-level context is more important to users than raw data
• A system focused on creating semantic information better suits a user’s needs
• *Octopus* addresses this
  • Easy to extend with new hardware/software
  • Easy to reuse low-level information to create high-level semantic information

• Future Directions
  • Grow deployments to span large areas and multiple buildings
  • Extend into higher-level application spaces