CS 525M – Mobile and Ubiquitous Computing Seminar

Mark Figura

About the article

"IrisNet: An Architecture for a Worldwide Sensor Web"

Phillip Gibbons, Brad Karp
Intel Research Pittsburg

Yan Ke, Suman Nath, Srinivasan Seshan
Carnegie Mellon University

From *IEEE Pervasive Computing*, Oct-Dec. 2003

Introduction

- IrisNet = Internet-scale Resource-Intensive Sensor NETwork services
  - Doesn't "IrisNet" sound cool?
- World-wide sensor web
  - UI is like a database
  - Users query the sensor-web for the information they are looking for
    - Parking spaces
    - Coastal conditions

Introduction

- IrisNet will basically do everything
  - Alerts – head to the bus stop; A tornado is coming!
  - How much time to wait for stamps at the post office
  - Where’s the nearest parking space?
  - Lost and found – where’s my stuff?
  - Watch-my-child-when-I’m-at-work
  - Health alerts (watch out for the new flu)
  - Homeland defense (watch out for those inbound missiles)
  - Many more!

How is this supposed to work?

- Each sensor will retain it’s own data until it is necessary to transmit – keeps transmission down
- Ability to change sampling rates – don’t sample a lot when nothing is happening
- One single interface for everything – one query tool does parking spaces, coastal oil-spill monitoring, watch-my-child, etc
- Data can be queried from anywhere
- Data integrity / privacy (usually an afterthought)
- Ability to deal with equipment failure
- Ease of writing ‘services’

Writing services

- “A tornado is coming!”
  - Uses many different sensors
- “Is it a nice day today?”
  - Uses many of the same sensors!
- A ‘naïve’ implementation would require redundant sensors
- IrisNet allows the reuse of sensors
  - Cheaper
  - Easier for service authors
  - Provides an interface to sensors that can be queried from multiple services

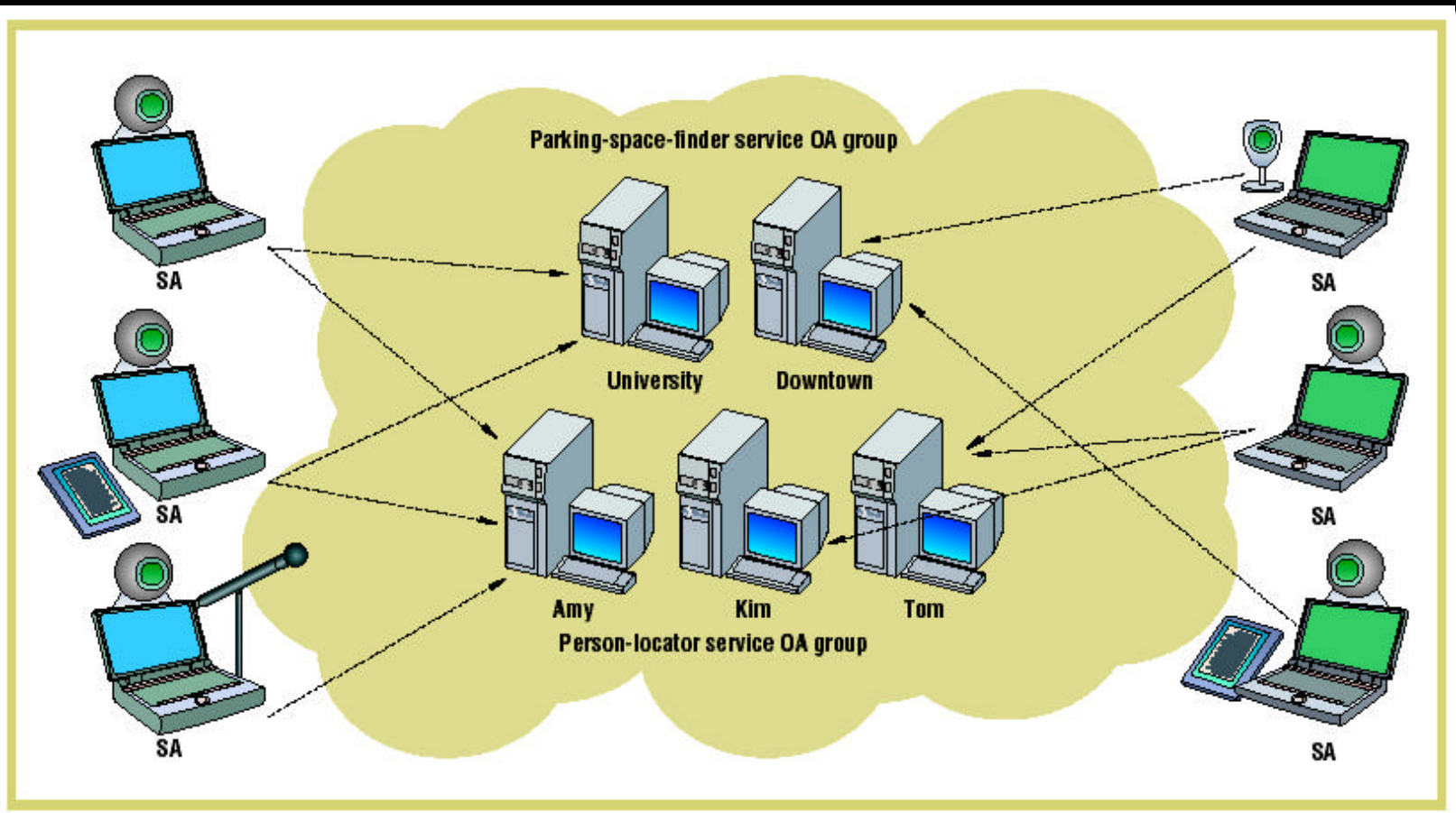
Data -> services

- Services request processed data
  - Instead of receiving video footage, ask for a time-lapse picture
  - Reduces transmission, power, time
- Data is updated often – traditional database systems are less than optimal
  - IrisNet can deal with this
  - ‘Partition’ database across multiple nodes
  - Local data (barometric pressure in Boston) is stored locally (in Boston)

IrisNet architecture

- Two types of nodes on IrisNet
  - Sensing Agents (SAs)
    - Sensors that implement the IrisNet generic data acquisition interface
  - Organizing Agents (OAs)
    - Nodes that store a distributed database of information collected from one or many SAs

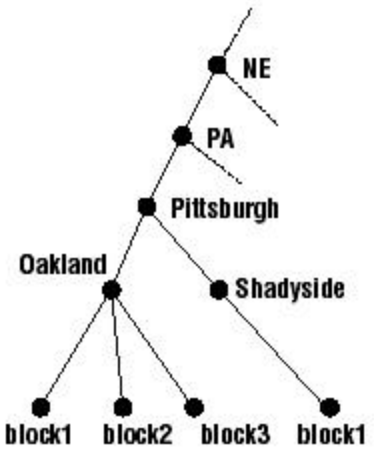
IrisNet architecture



(Note that multiple SAs and OAs can be run on a single computer)

OA architecture

- Each service consists of a number of dedicated OAs.
- Services can share SAs, but not OAs
- Use XML for the database
  - XML provides good structure to the data
  - Another buzzword for the paper



Distributing the database

- Database is partitioned with “a distributed algorithm”
- Use structure of the database along with DNS to locate each node
  - city-Pittsburgh.state-PA.usRegion-NE describes the city of Pittsburgh and can also be registered as a DNS name
  - The Pittsburgh OA’s IP address would be bound to the DNS name
- What about name collisions?

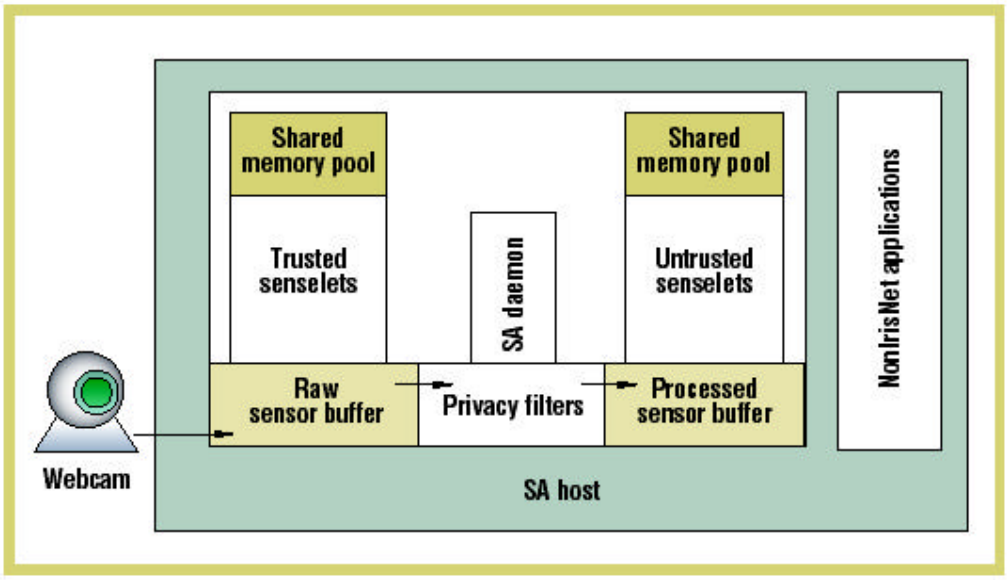
Querying the database

- Distributed nature of the DB makes it difficult to query
  - Send request to the Lowest Common Ancestor (LCA) (look up IP in DNS)
    - LCA = the node that is closest to the bottom of the tree, but can still access all data from its children and/or itself
    - (querying of siblings and parents are not allowed)

Consistency

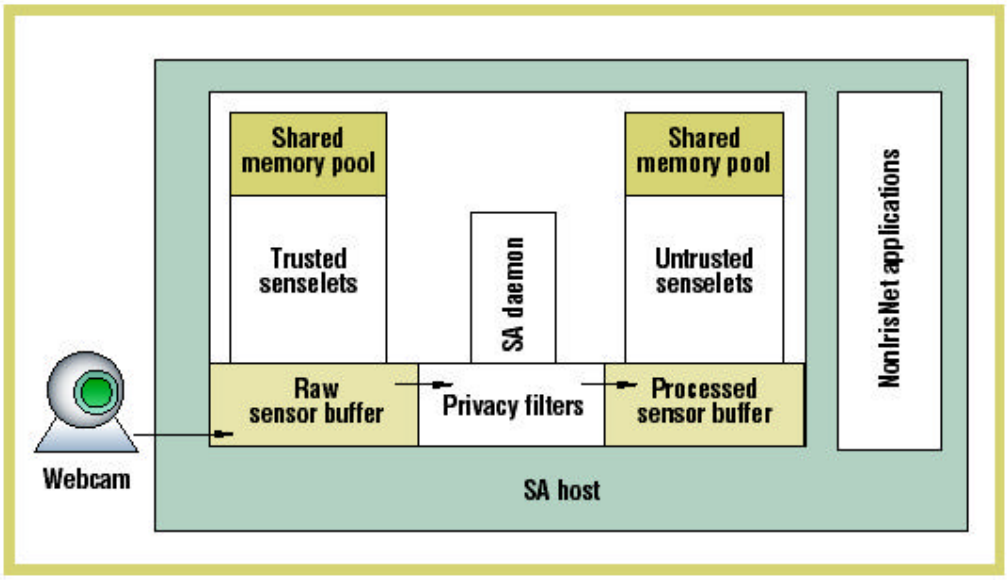
- Data in OAs might not be most current
- For example, an SA might monitor for riptides by sending 10-minute time lapse photos to an OA
- If one starts to develop right after a photo is sent to an OA, there will be about 10 minutes of riptide before lifeguards are notified
- Also, if there is only a small change, data might not be transmitted to cut down on transmissions = energy, time

SA architecture



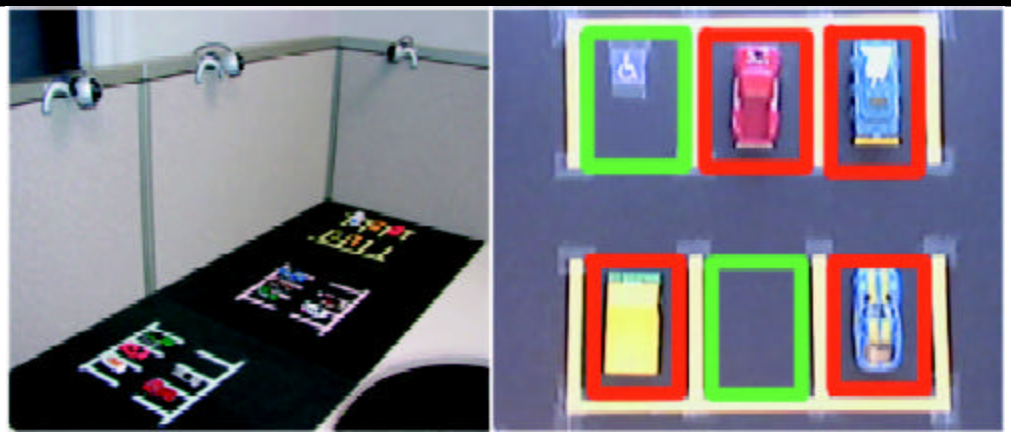
- Senselet = program that filters data into a form useful for an OA
- Protection from buggy or malicious senselets
  - Each senselet runs as its own process
    - Protected memory is great and all, but this is **not any** protection from malicious senselets!
  - Limit resource usage
    - Doesn't solve the problem either – they're malicious after all!

SA architecture



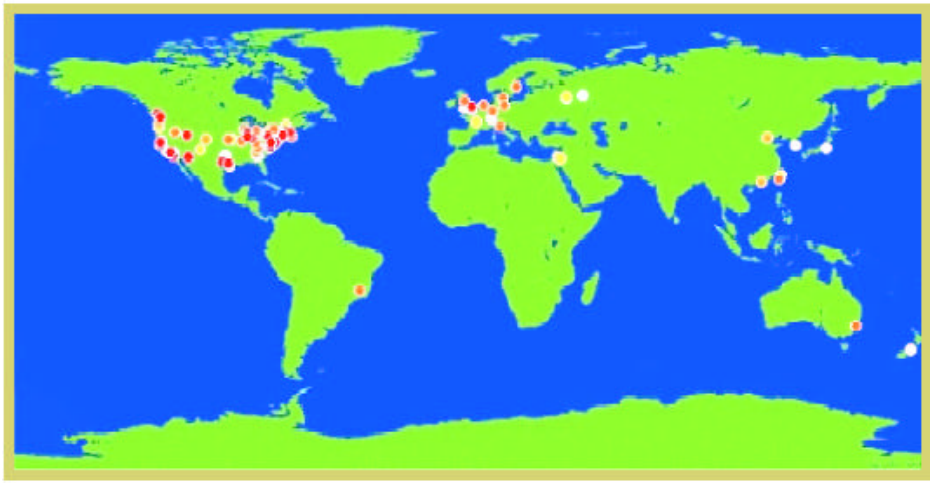
- Privacy
  - Privacy filters remove identifying information
  - ie. Put black boxes over faces and license plates
- Shared memory pools
  - Senselets can work together
  - ie. Many audio-based senselets might have to do a FFT on the audio data. If one senselet does it, other senselets can use it

Cool stuff - parking



- Tested on a mock-parking-lot with Matchbox cars
- Allows queries that include constraints on the parking space – handicapped, covered, etc
- Uses Yahoo! Maps to get directions to the parking spaces

Cool stuff - IrisLog



PlanetLab (AKA "US-and-Eastern-Europe-College-Lab")

- PlanetLab allows monitoring of computer usage through *Ganglia*
- IrisLog supports all *Ganglia* queries and more
- Integrated into PlanetLab
- More efficient thanks to the "distributed algorithm"

Cool stuff – Coastal imaging

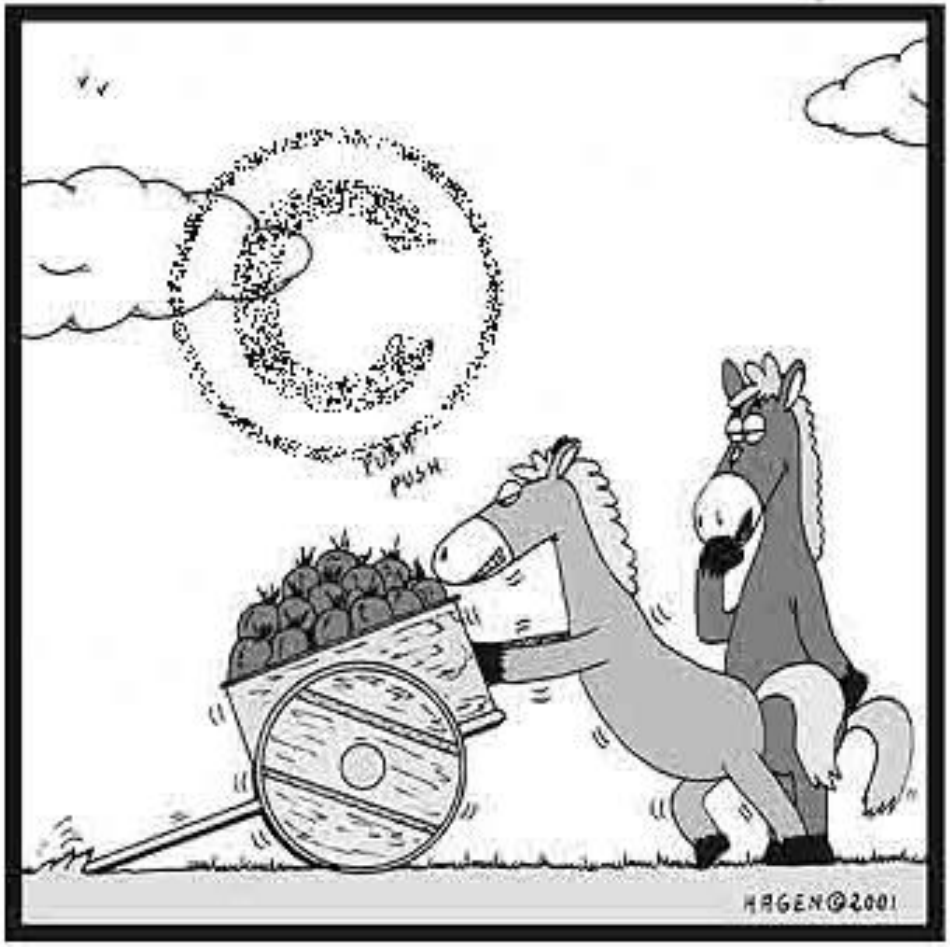
10 minute time lapse photo constructed from a video camera near Oregon State University

- Time-lapse photos are useful for detecting sandbars

Paper's conclusions

- In the past, sensor network research has been on creating sensors
- This paper discusses a software architecture for getting information from these sensors once they're deployed
- "While IrisNet represents an important first step … [i]mportant policy, privacy, and security concerns must be addressed before rich sensors can exist pervasively at a global scale."

My conclusions



Hang on... We must be doing something wrong...
How does the saying go again?

My conclusions

- This is not necessary yet?
  - It will be a while before sensors are ubiquitous
  - Other new technologies will be invented at that point
  - Don’t hold back 2030(?) sensor technology with 2003 software paradigms
- That being said…
  - It is important that these things are thought about before sensors are deployed!

My conclusions

- Most topics in the paper aren't anything novel
  - A description of the "distributed algorithm" for partitioning databases might have been interesting
  - However, it's pretty obvious that something like a query-able distributed database will be involved in a global sensor-web
- SA / OA – interesting extension of OOP to sensors / databases
  - By the time we have sensors everywhere, another programming paradigm might be more popular / better?
- Paper authors just trying to get their names out there?

My conclusions

- In summary…
  - Of course it's necessary to have a nice software architecture to go along with the hardware sensor deployment
  - Right now, we don't need this
    - Parking space finder
      - Neat tool, but it doesn't need IrisNet
      - Such varied tasks such as "inbound missiles!" "find a parking space" and "watch my child" would be awkward on a single interface.
  - A fully-developed software architecture should be developed before sensor deployment, but not now

Questions?