Challenge Problems in Sensornet Research

Feng Zhao

Microsoft Research

http://research.microsoft.com/zhao

Keynote at NSF NOSS PI meeting and Distinguished Lectures at Johns Hopkins and Princeton, October 18-21, 2005

Acknowledgment

- Benefited from discussions with my colleagues at MSR, Xerox PARC, Stanford
- Joint work with Jie Liu, Leo Guibas, Jeremy Elson, Alec Woo, Kamin Whitehouse, Elaine Cheong, Prabal Dutta, Zoe Abrams, Siddharth Seth, Ryan Newton, Andrew Parker

Challenge for the Sensornet Community

- Great to see startups begin to tackle practical problems, collecting low-hanging fruits

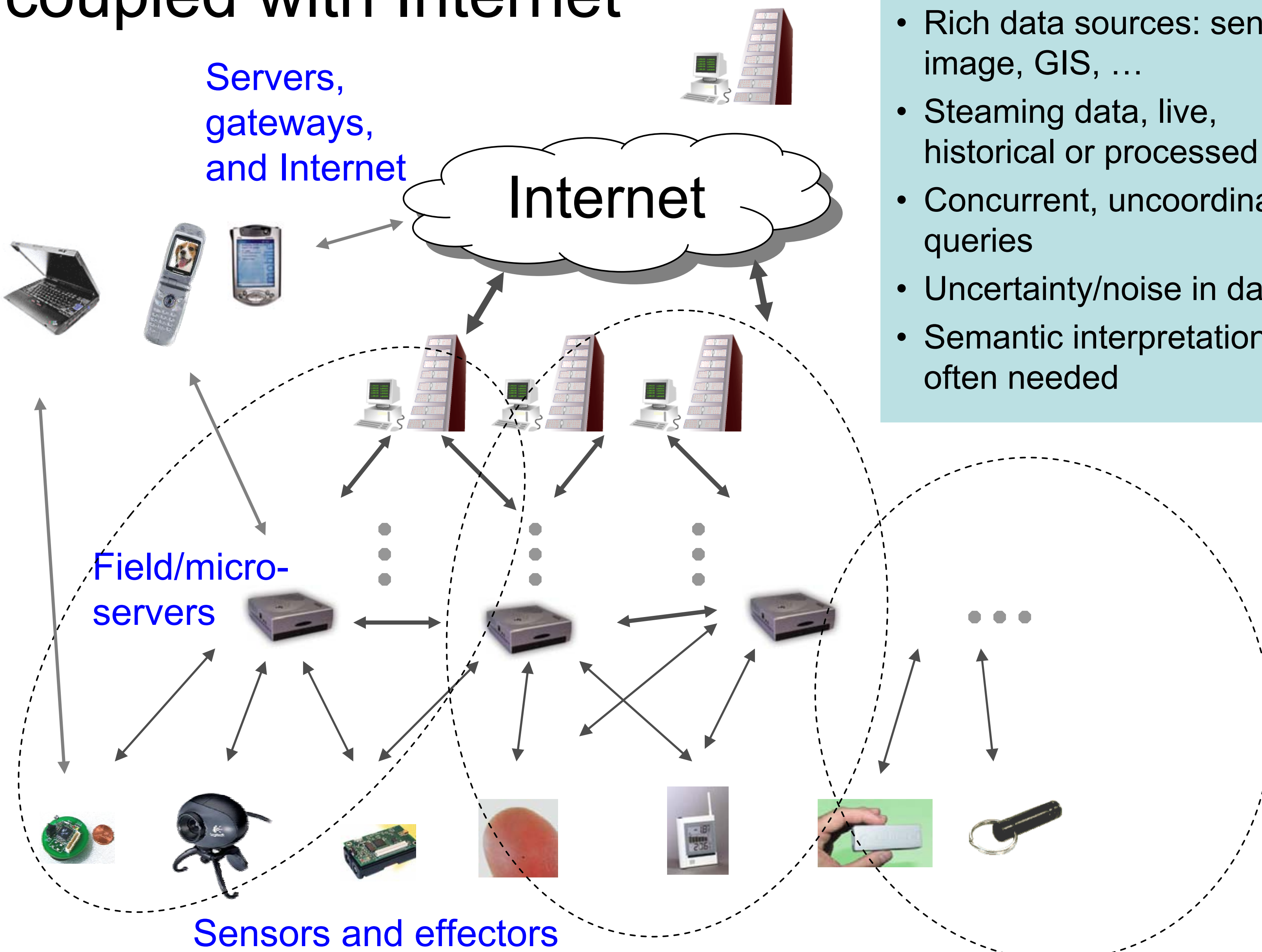
- But, the research community must set the sight further
  - Every field needs grand challenges to focus energy and drive progress, so does sensornet
  - Here is my take on the grand challenges; substitute it with you favorite …

Reality Browser: Query physical world, live and up close, from anywhere

Potential Apps

- Environment
  - Volcano, underwater, rainforest
- Education
  - K-12 interactive learning, virtual laboratory
- Leisure
  - Virtual travel, sunny spot tracker (microclimate), what is the temperature at my favorite beach? what is the water algae level?
- Getting around
  - Where is the nearest available parking space? What is the traffic like on the bridge? How long is the queue at the gas station? Where is the bus?

Multi-tier net of sensornets, coupled with Internet



Characteristics:

- Rich data sources: sensor, image, GIS, …
- Steaming data, live, historical or processed
- Concurrent, uncoordinated queries
- Uncertainty/noise in data
- Semantic interpretation often needed

Three Challenge Problems

- Need to standardize **interfaces** between apps/db/nw/dsp/hw (e.g., SP link layer abstraction, SensorML)

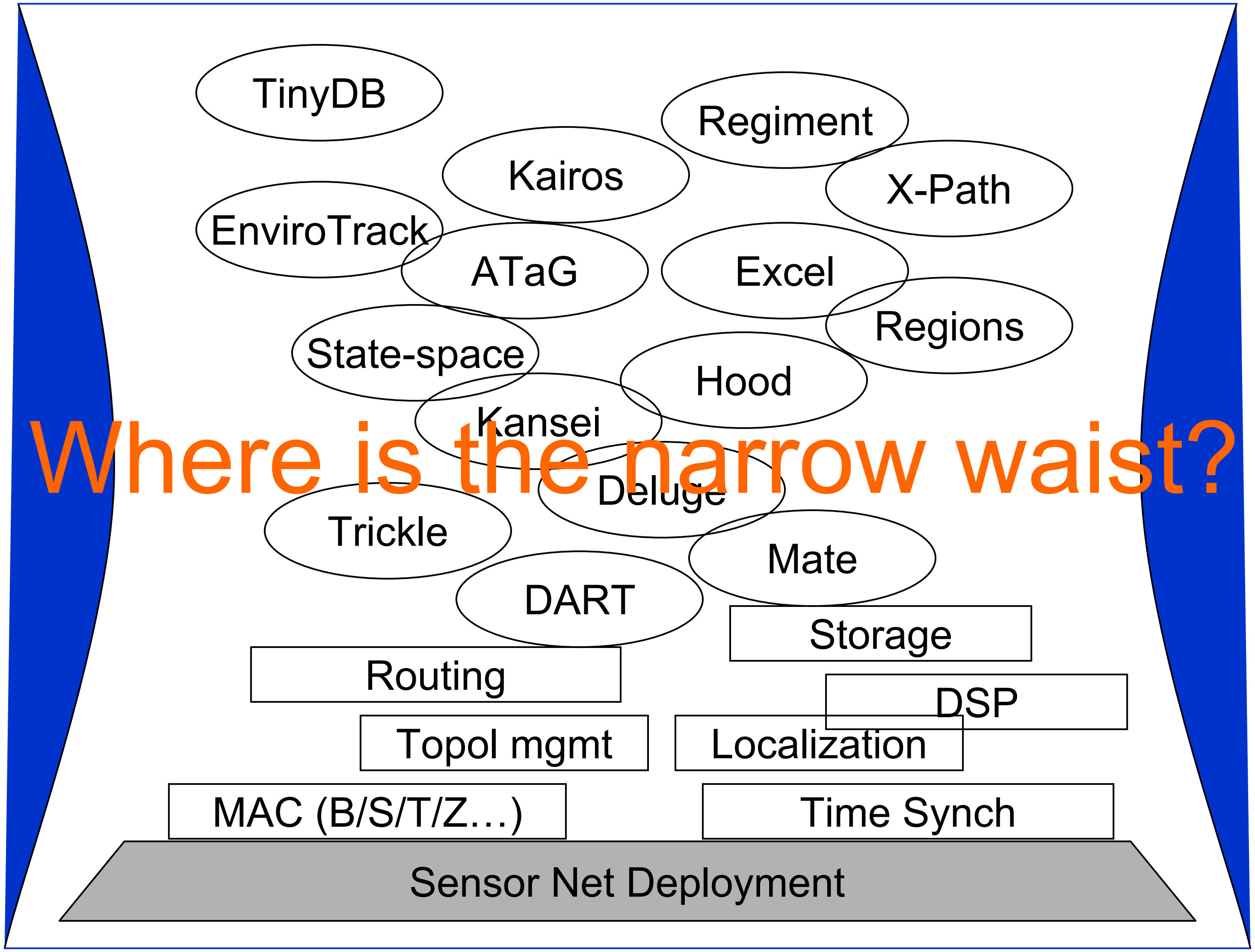
- Need to treat **uncertainty** (in both data and systems) as first-class entity, for reasoning and system mgmt

- Need simple **tools** for sys config/mgmt, for data collection and vis, for in-situ debugging

But the problem is made hard by …

- A great deal of variability
  - Variability in app requirements
    - Use scenarios (data collection, control in the loop, or in-network compression), data rate, hw
  - System dynamism and unreliability
    - Parts of system may be deployed over time by different vendors, using different technologies (e.g., network protocols)
    - Nodes/links come and go
    - Possibly non-replenishable resources
  - Variability in data
    - Uncertain data due to sensor noise, packet loss
    - Incomplete information due to partial observability of the world
- User tasks often specified in high-level semantic queries
  - E.g., “Tell me if you see a red car”, or, “Doing this with that data”.
- As a result, systems are often built vertically
  - With own system abstractions and data models
  - Make sense from efficiency POV, but with relatively small code reuse
  - Phil Buonadonna: “Every sensor net experiment needs 2 PhDs and 5 grad students”, or something like that

Many systems got built over the past few years …

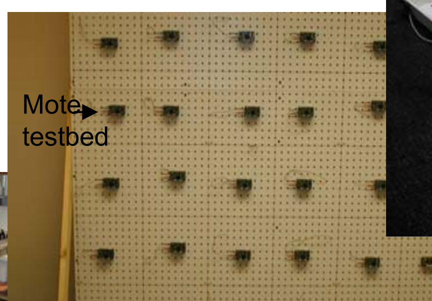


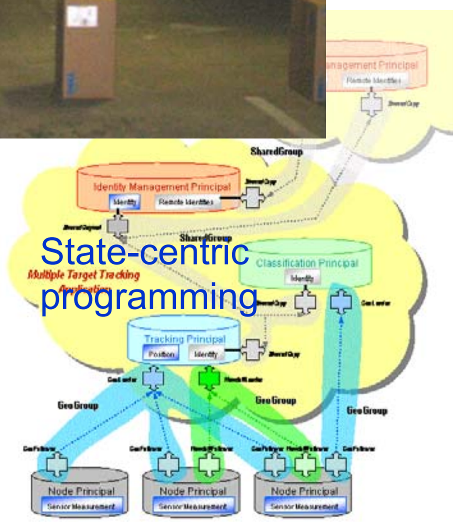
From printer land, Mojave Desert, to parking garage … our own experiences

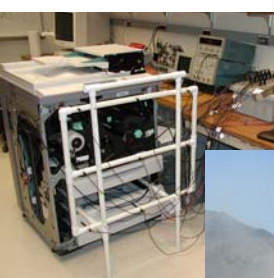
MSR in-building testbed, 2005

MSR parking garage testbed, June 2004

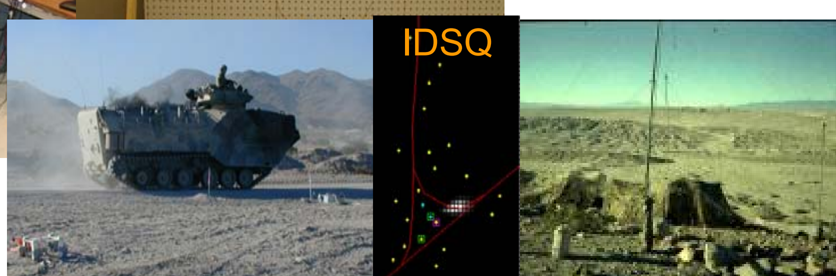
PARC camera testbed, 2003



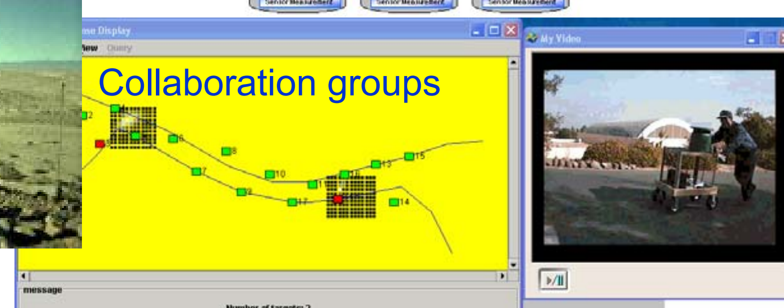




Machine diagnostics, 1997



DARPA SensIT experiments, 29 Palms, 2000-2001



Outdoor experiment, 2002

What we've learned …

- Proposed tracking as a canonical problem for sensorset problem (circa 2000)
  - Dynamic data, close the loop between sensing & decision making (not just data collection)
  - Created SensIT data sets (with ground truth!), app scenarios, experiemental testbed (used by others)
  - Many projects since then: SensIT 29P demo, NEST demo, EnviroTrack, …
  - But the utility of tracking app is often a function of good signal processing

- Developed the IDSQ (Information Driven Sensor Querying) framework/Sensor tasking (circa 2001)
  - Pay more attention to data: value of information
  - Interface btw app ↔ routing: value of info and routing decision at each node
  - Move beyond individual nodes: group/neighborhood management
    - Many other work in this space: Hood, Regions, …
  - But ours (and others) with it own interface assumptions and data representation

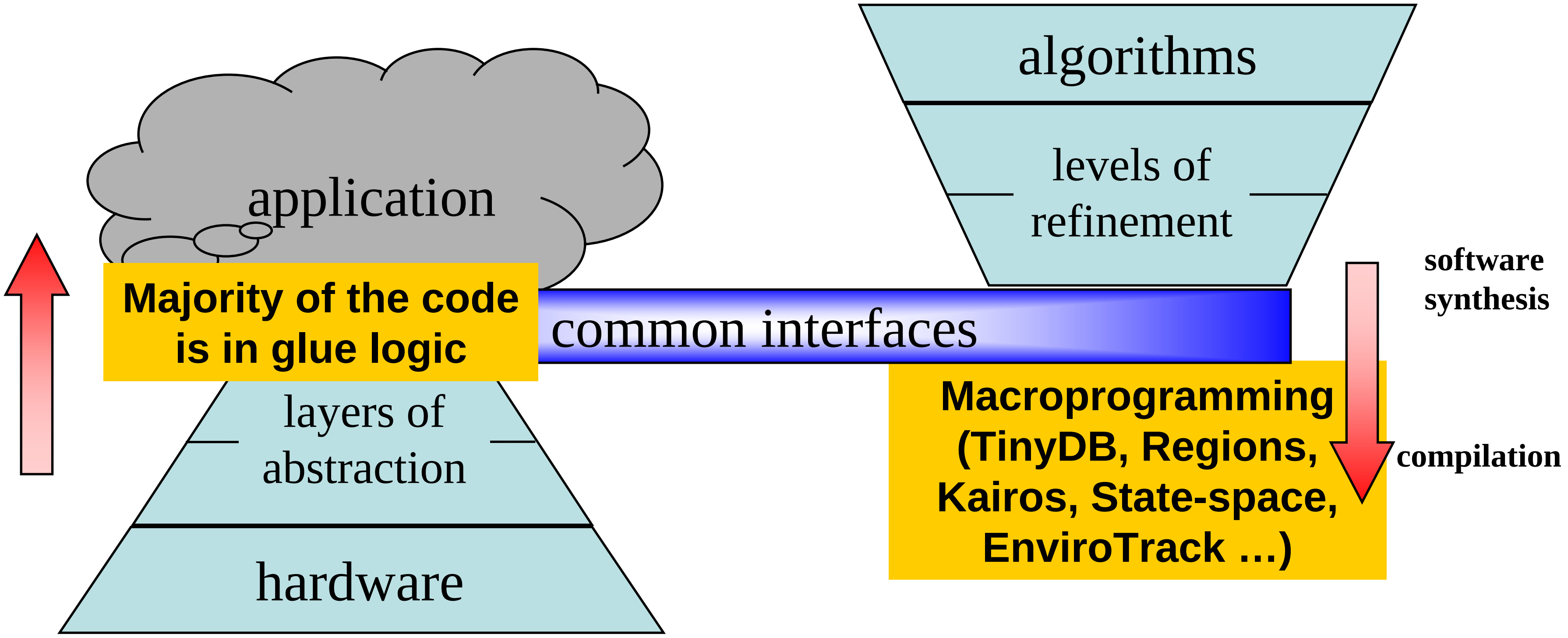
More on Lessons Learned

OS/Network-centric view

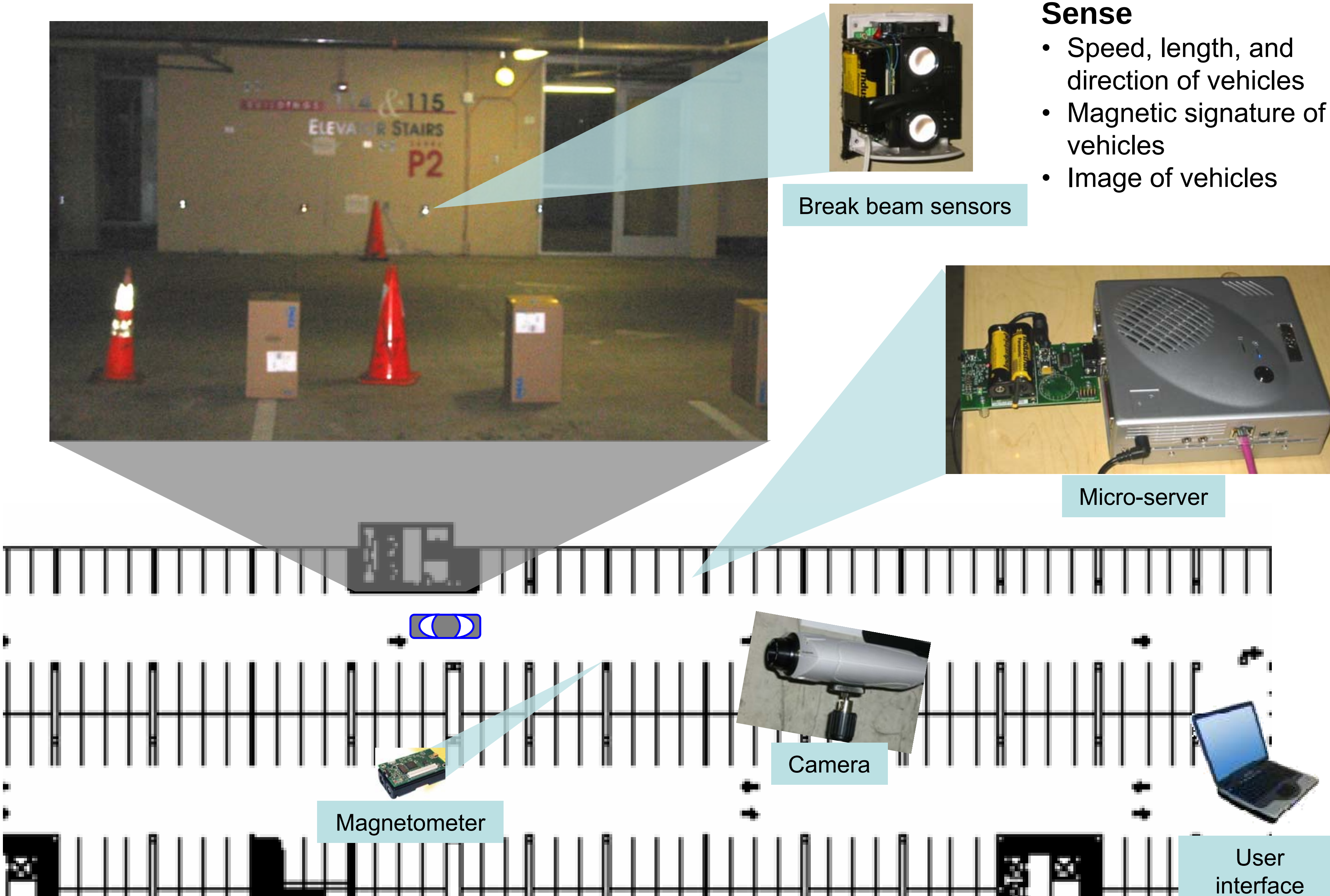
Designing component-level abstractions

Information-centric view

Designing application-level abstractions



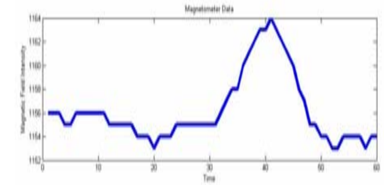
A Parking Garage Example

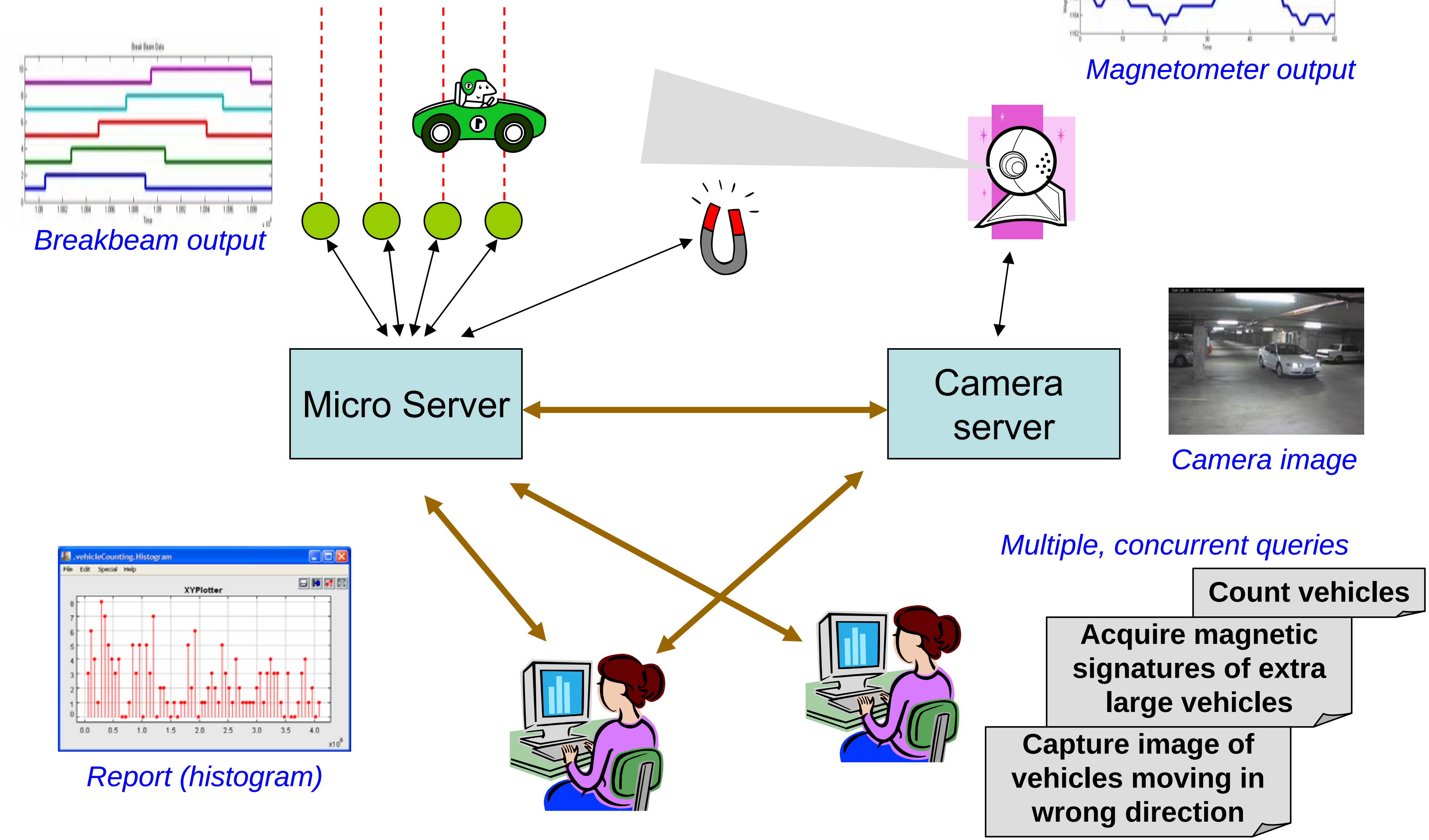


Sense

- Speed, length, and direction of vehicles
- Magnetic signature of vehicles
- Image of vehicles

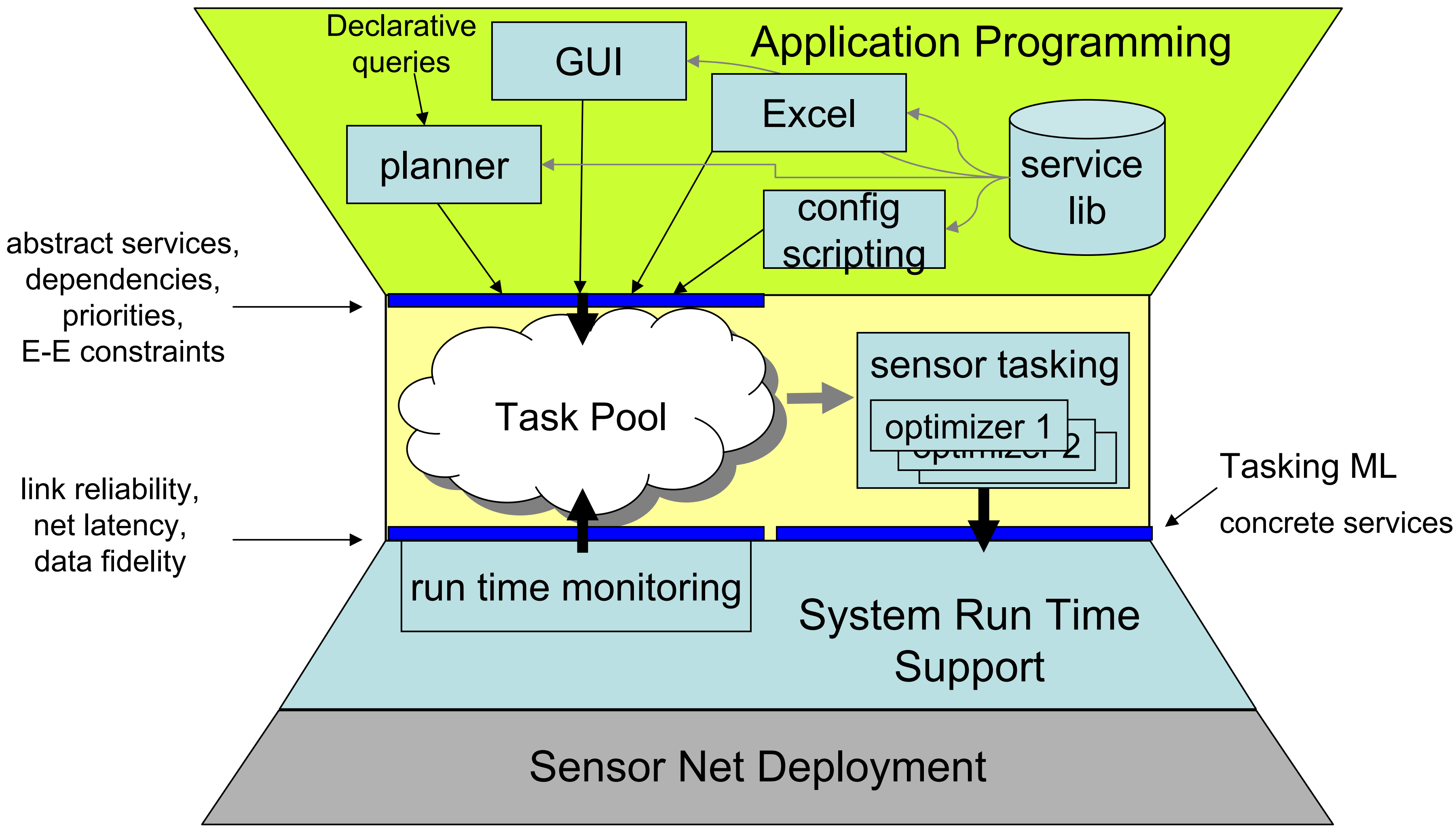
Concurrent Uncoordinated Tasks



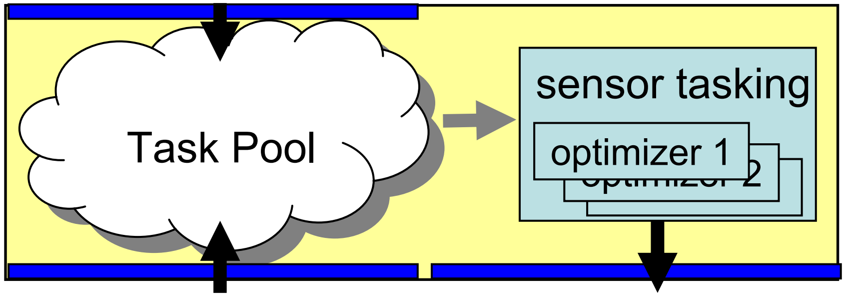


Tasks are sent to microservers at uncoordinated times, running for unpredictable duration.
Tasks may partially overlap.

Interfaces Between Apps and Run-Time



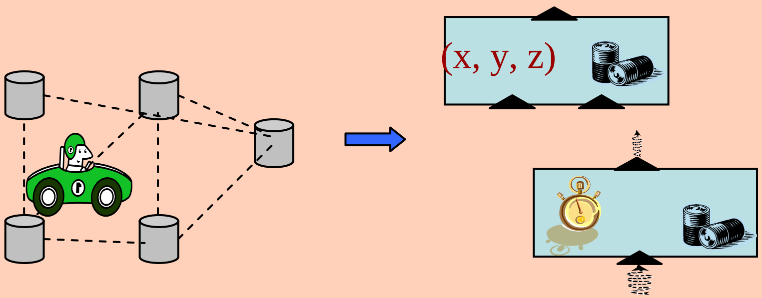
Task Pool Abstraction



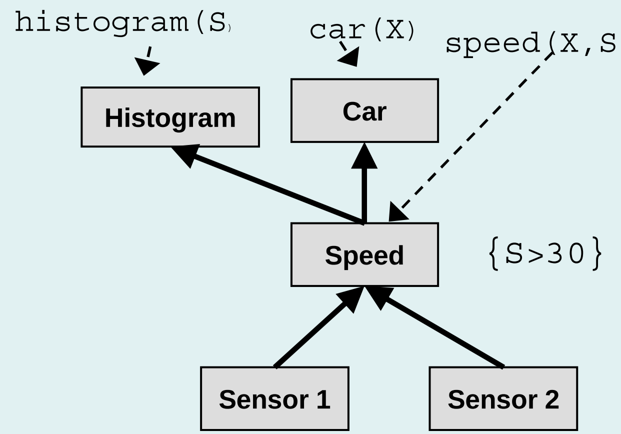
- Application programs inject abstract services and their dependencies into task pool, specifying what/how
  - Services have priorities
  - Applications have end-to-end constraints such as latency, data quality, energy usage
- Sensor Tasking embed services onto nodes, instantiate where/when
  - Need information about physical topology, link quality, latency, data fidelity from run time
  - Search for optimal assignment satisfying EE constraints
    - Using a variety of tools such as CLP(R), LP, or Monte Carlo
  - Tasking ML describes concrete services instances
- Task pool is agnostic to application programming environments and run-time system support, provided that
  - Services are described in a common intermediate language
  - Run-time provides system and data reliability info

SONGS: Service Oriented Networked ProGramming of Sensors

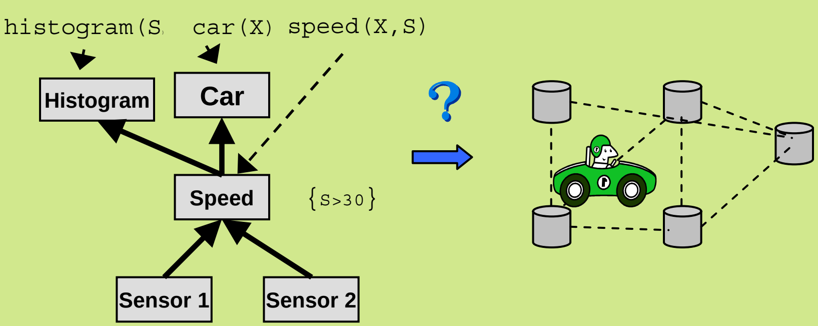
Service Abstraction and Interface



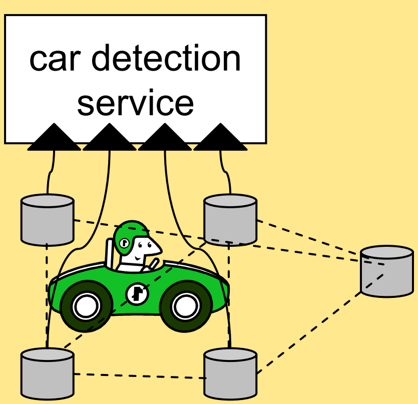
Service Planning



Service Embedding



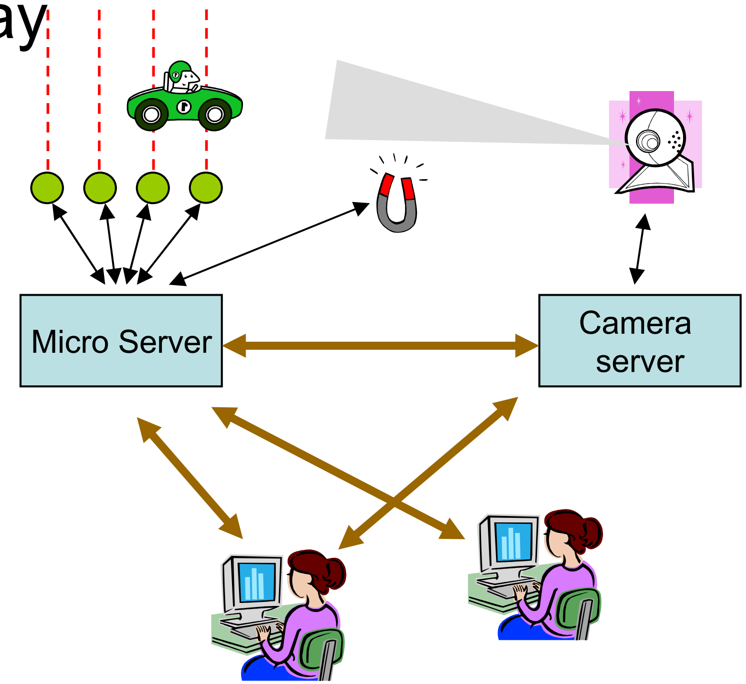
Service Scheduling and Execution

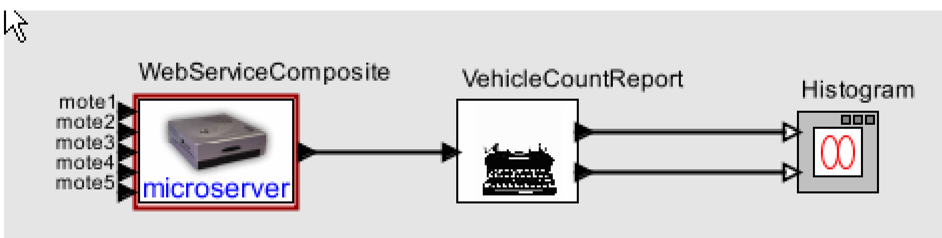


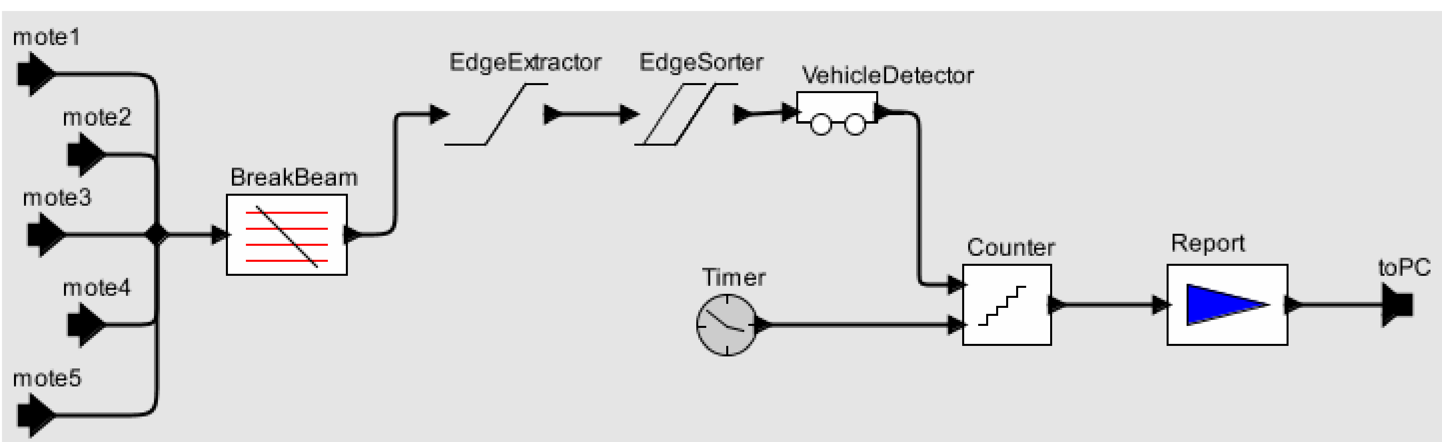
Example of services and their composition

Counting vehicles with a sensor array

- Extract edges from break beam detections
- Sort edges into consecutive detections
- Detect vehicles based on timing relations among detections
- Count vehicles
- Generate an arrival histogram report

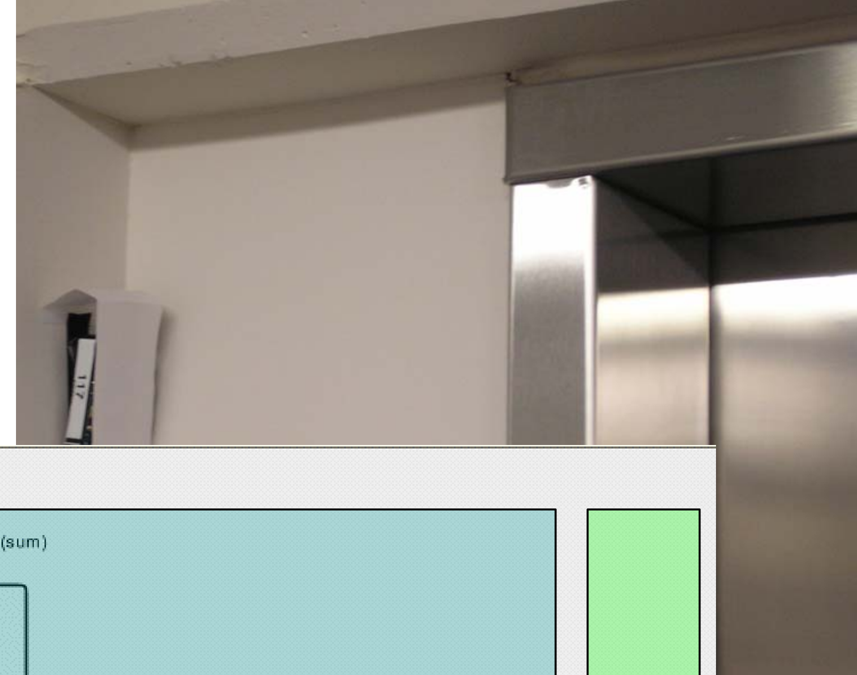


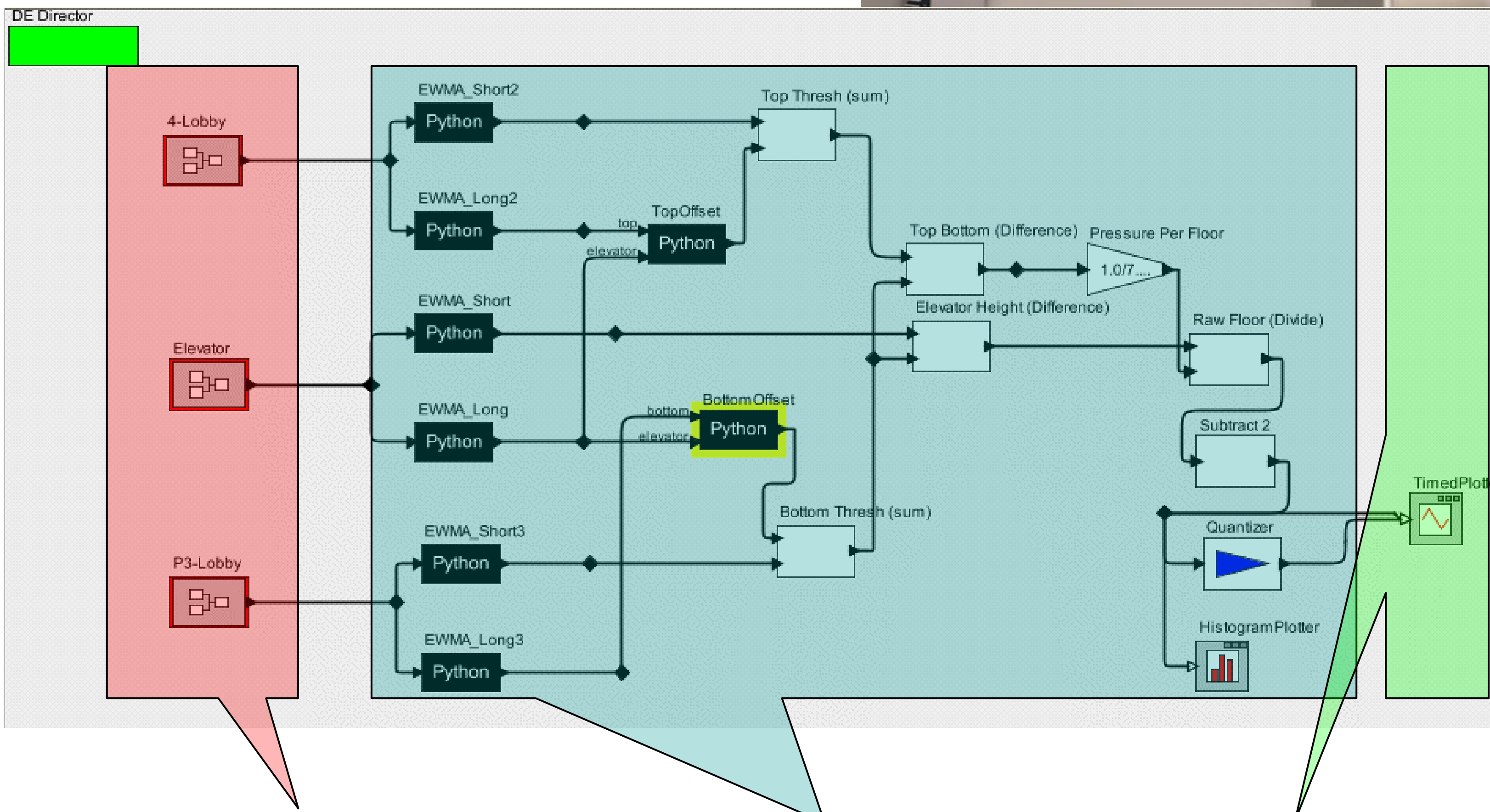




Elevator app

Jeremy Elson and Andrew Parker

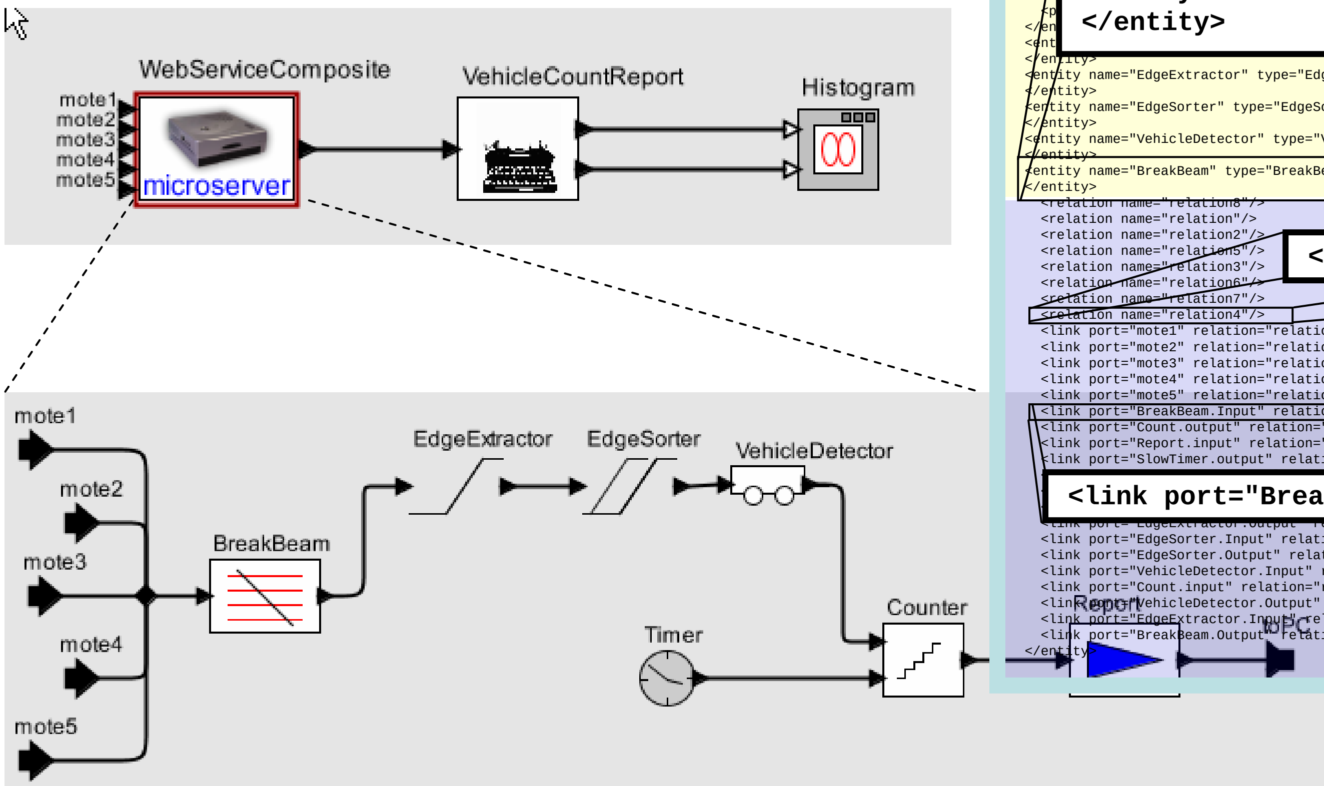


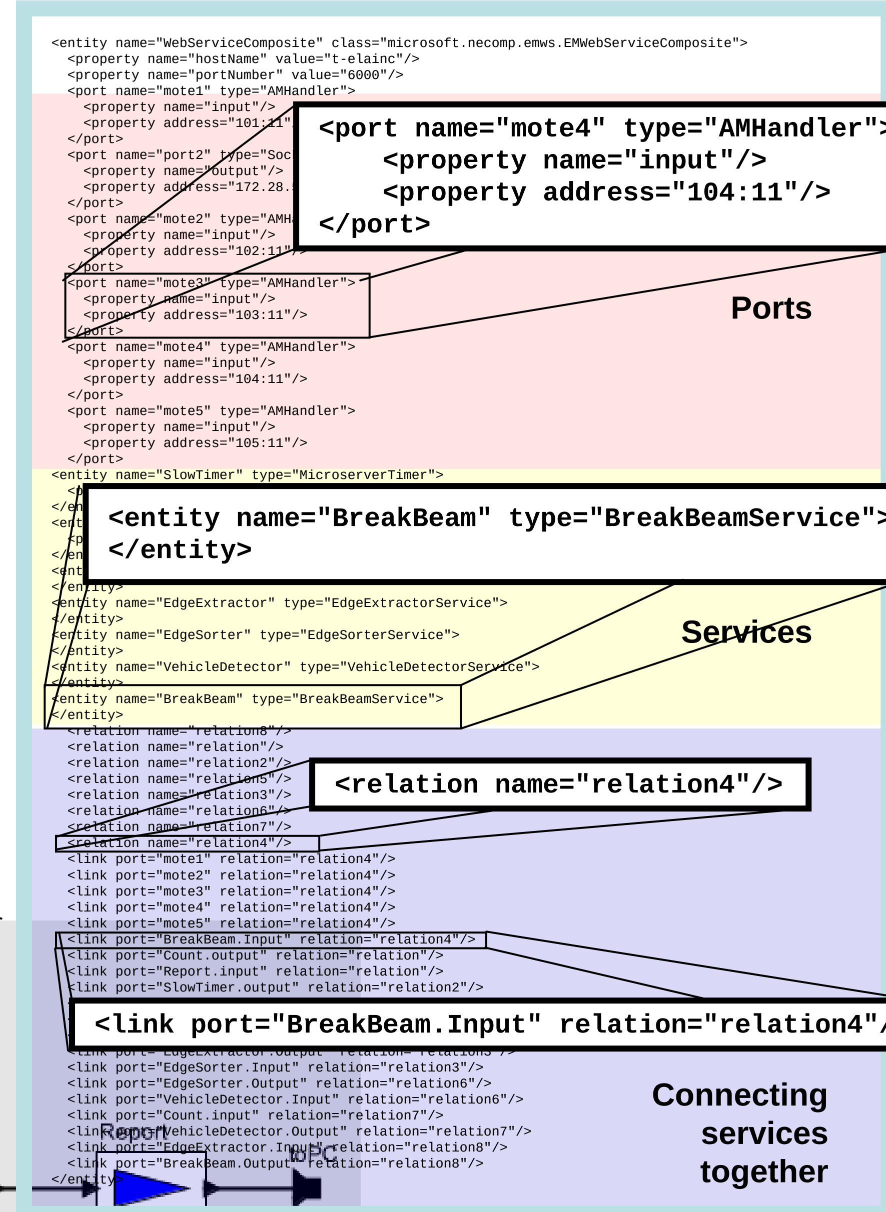


Data Sources Intermediate Processing Output

An Example of TML

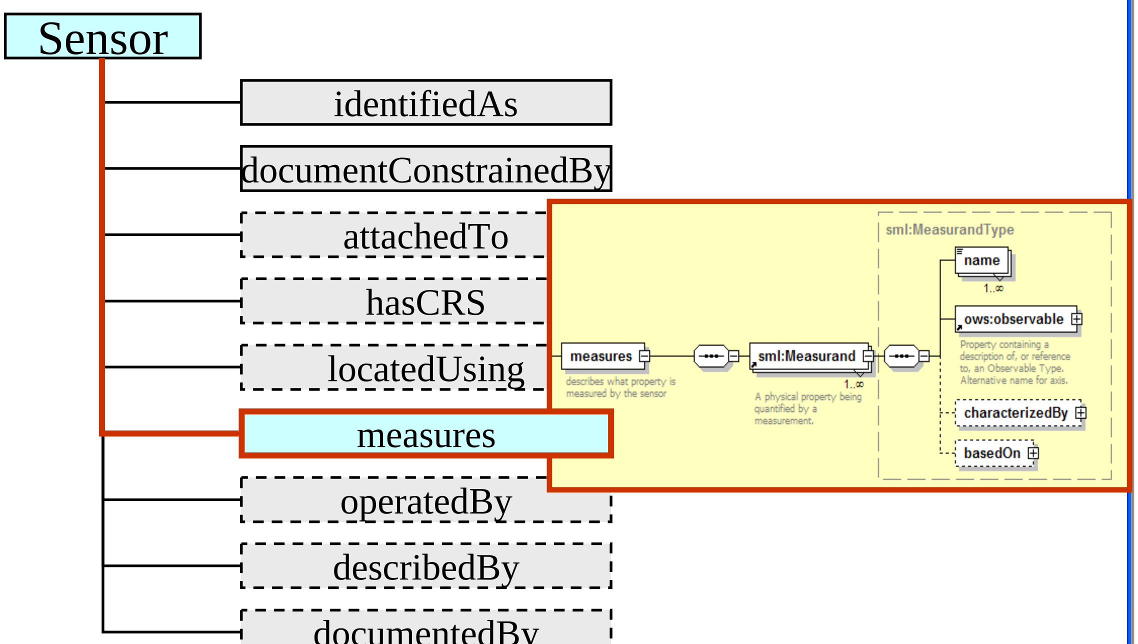
- Task graph for counting vehicle query
  - Ports
  - Services
  - Wiring services together
- Description in Tasking ML (micro-server tasking markup language)

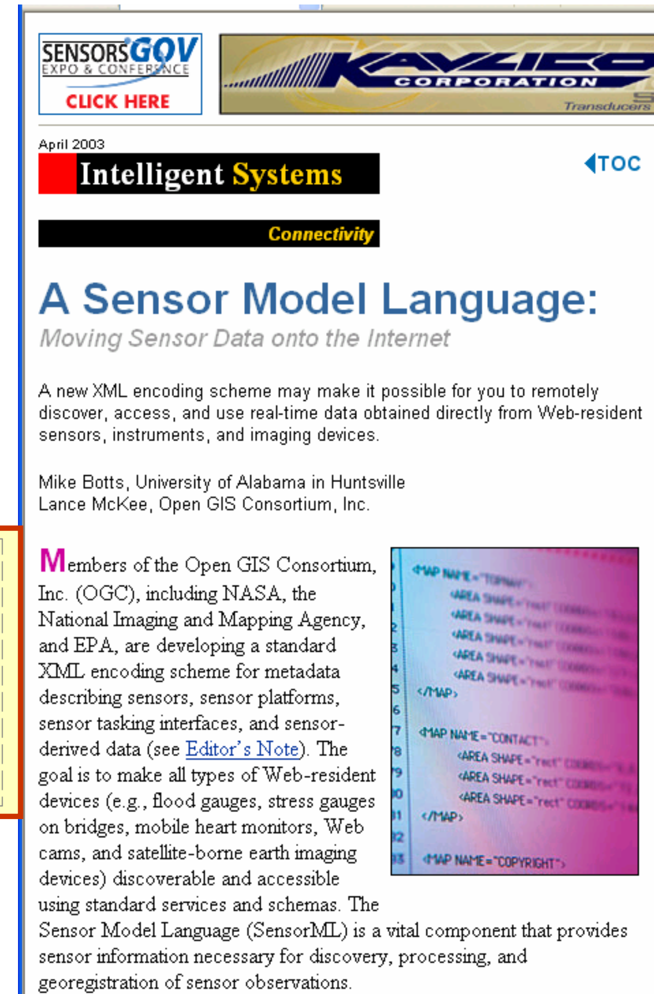




Standards for data/services/interoperation

- SensorML from Open Geospatial Consortium (http://vast.nsstc.uah.edu/SensorML/) ?
- Other interface standards?
  - Service description, data publishing, tasking





April 2003

Intelligent Systems

TOC

Connectivity

A Sensor Model Language:

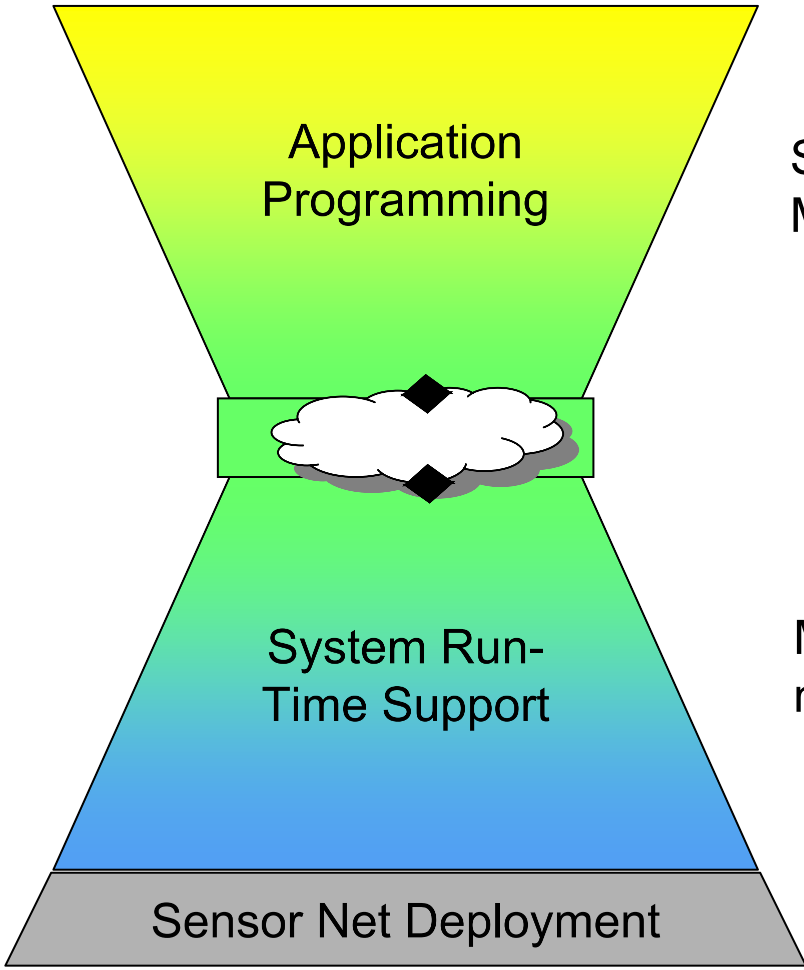
Moving Sensor Data onto the Internet

A new XML encoding scheme may make it possible for you to remotely discover, access, and use real-time data obtained directly from Web-resident sensors, instruments, and imaging devices.

Mike Botts, University of Alabama in Huntsville
Lance McKee, Open GIS Consortium, Inc.

Members of the Open GIS Consortium, Inc. (OGC), including NASA, the National Imaging and Mapping Agency, and EPA, are developing a standard XML encoding scheme for metadata describing sensors, sensor platforms, sensor tasking interfaces, and sensor-derived data (see Editor's Note). The goal is to make all types of Web-resident devices (e.g., flood gauges, stress gauges on bridges, mobile heart monitors, Web cams, and satellite-borne earth imaging devices) discoverable and accessible using standard services and schemas. The Sensor Model Language (SensorML) is a vital component that provides sensor information necessary for discovery, processing, and georegistration of sensor observations.

Narrow waist should allow applications to be specified independent of system configurations

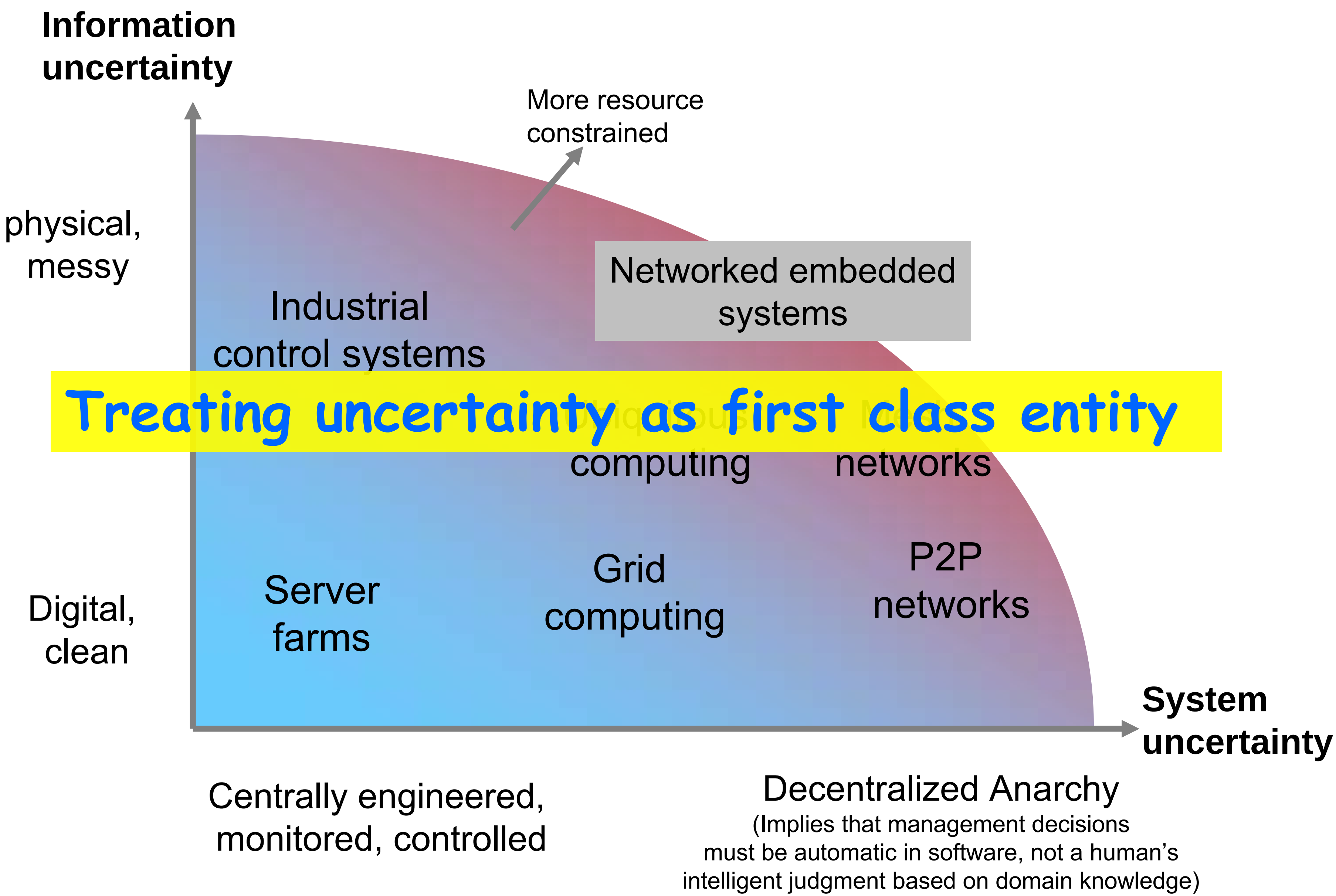


Spreadsheet, logic/functional, visual, MATLAB, SQL

Mote programs, microserver programs, network protocols, …

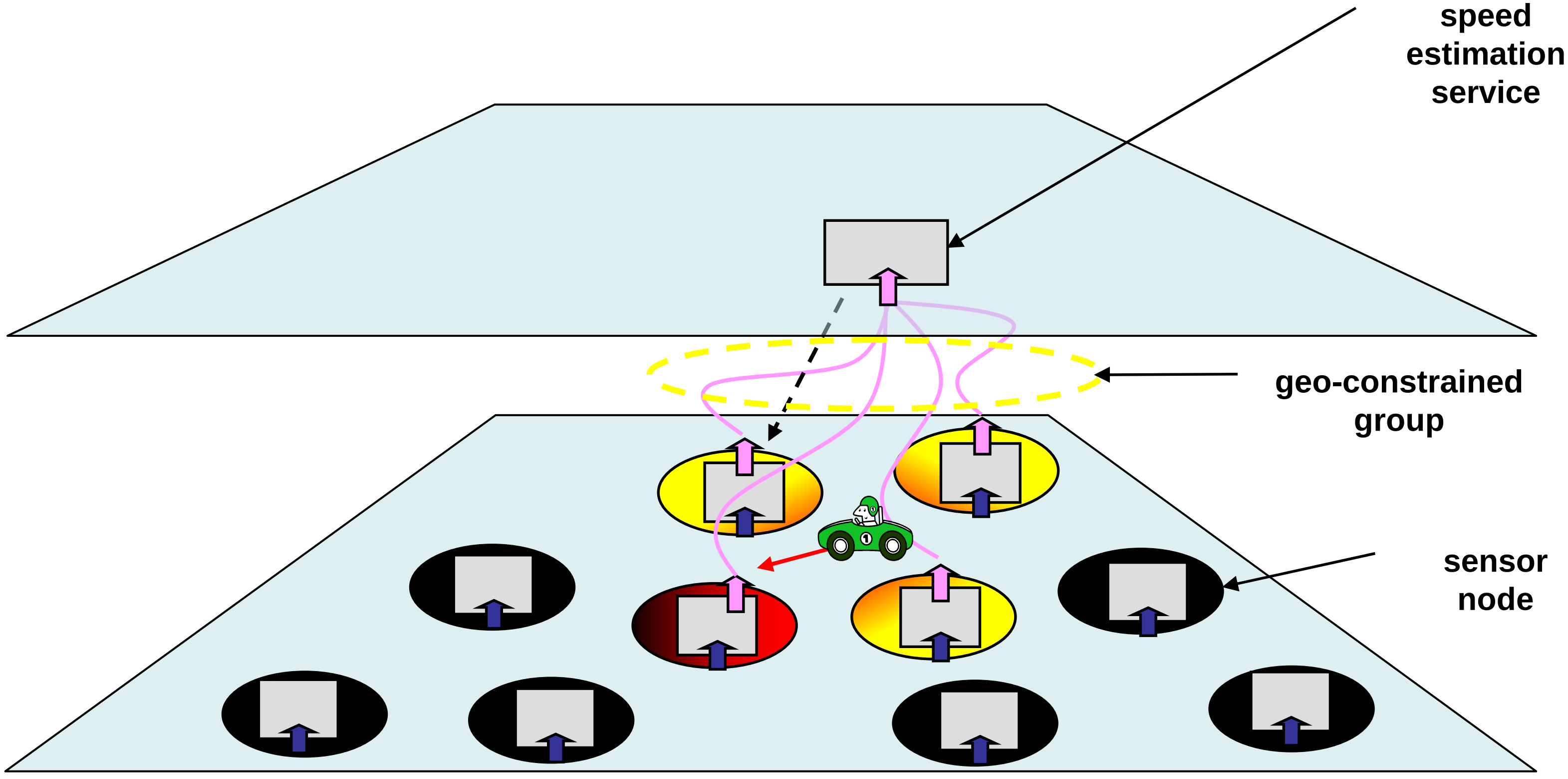
How to engineer reliable behaviors out of unreliable components?

- Sensor net today: reliable components, unreliable networks/systems, and unpredictable apps
- Biology: unreliable components, reliable systems, predictable behaviors


Information uncertainty
More resource constrained
physical, messy
Industrial control systems
Networked embedded systems
Treating uncertainty as first class entity
computing
networks
Digital, clean
Server farms
Grid computing
P2P networks
System uncertainty
Centrally engineered, monitored, controlled
Decentralized Anarchy
(Implies that management decisions must be automatic in software, not a human's intelligent judgment based on domain knowledge)

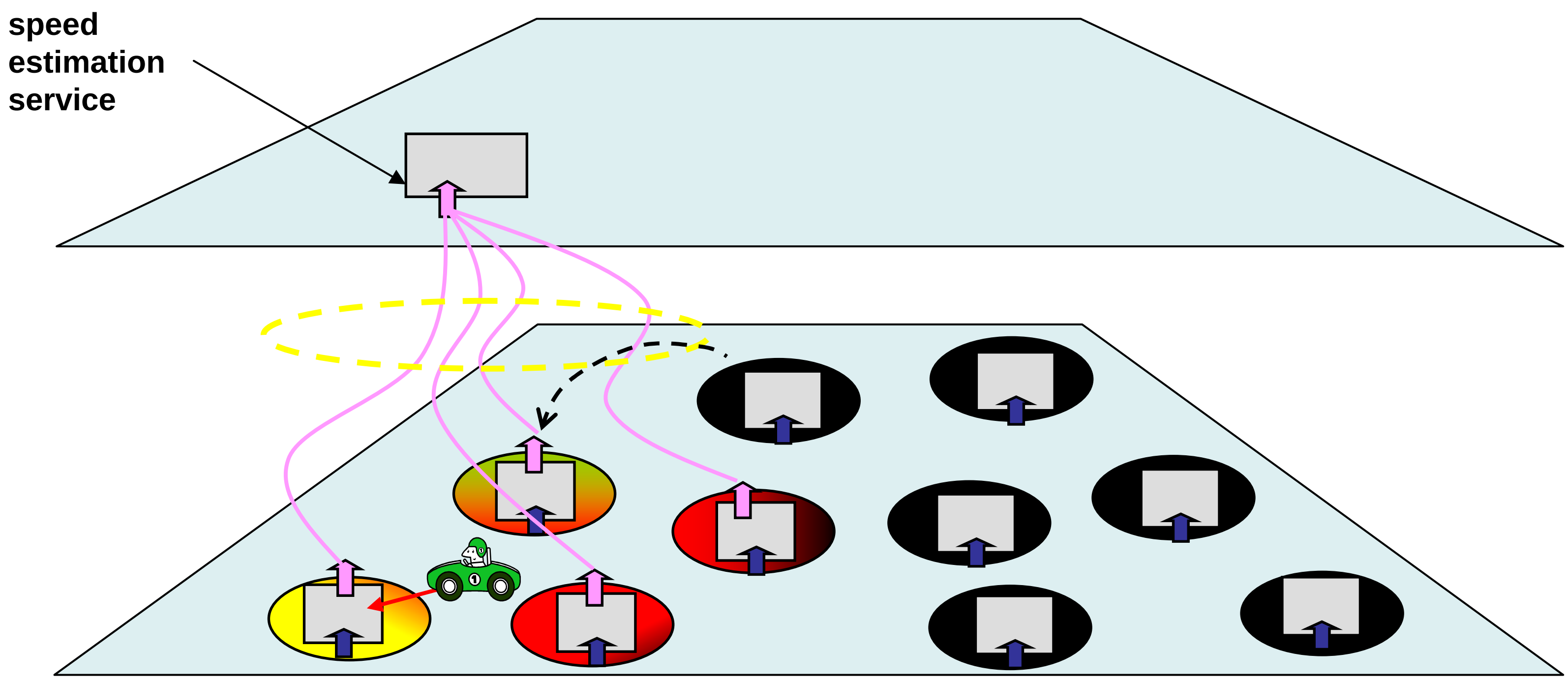
Manage both system and event uncertainty

A tracking example



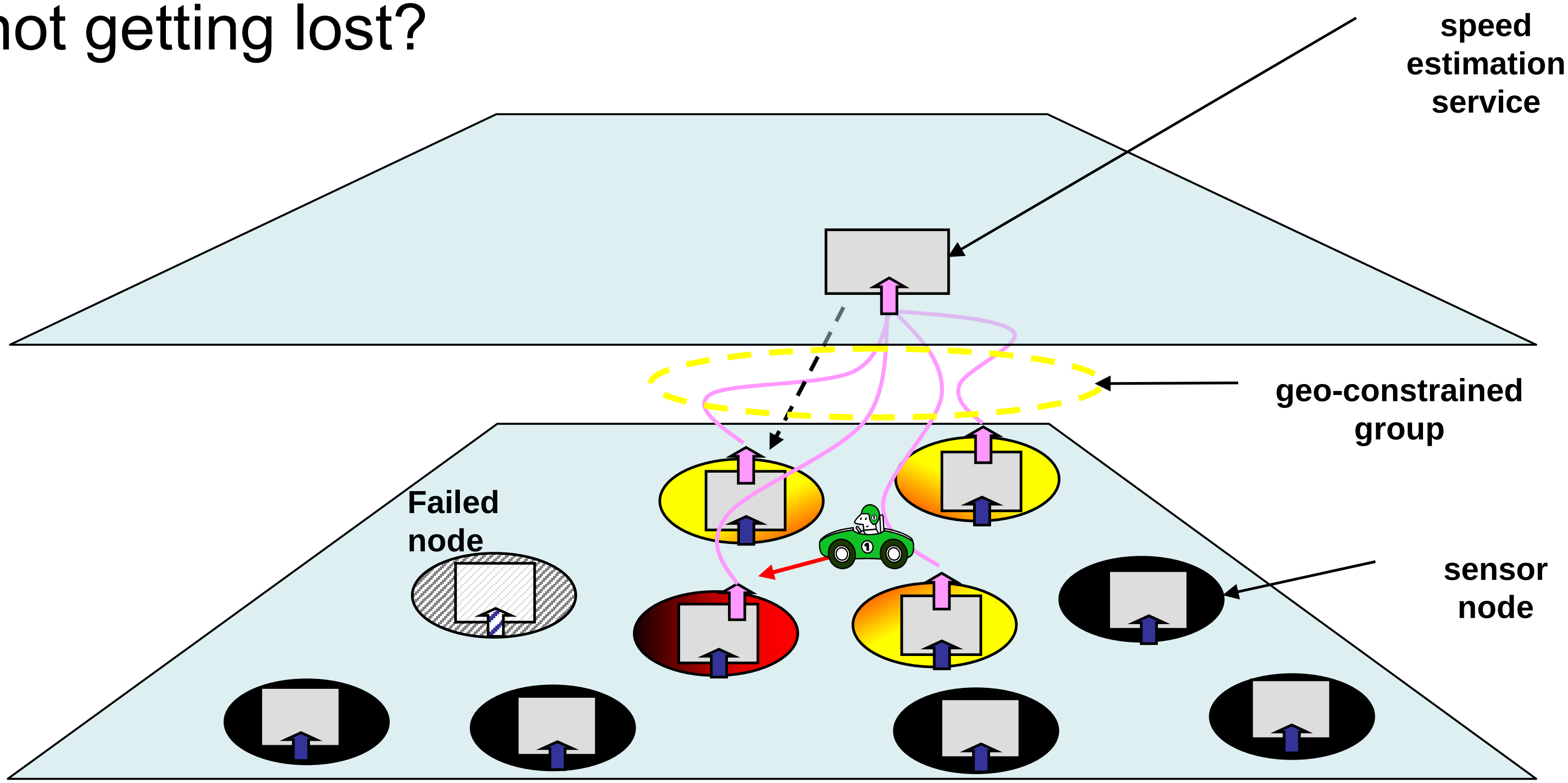
Migration …

As the event moves, one needs to move the code and/or state



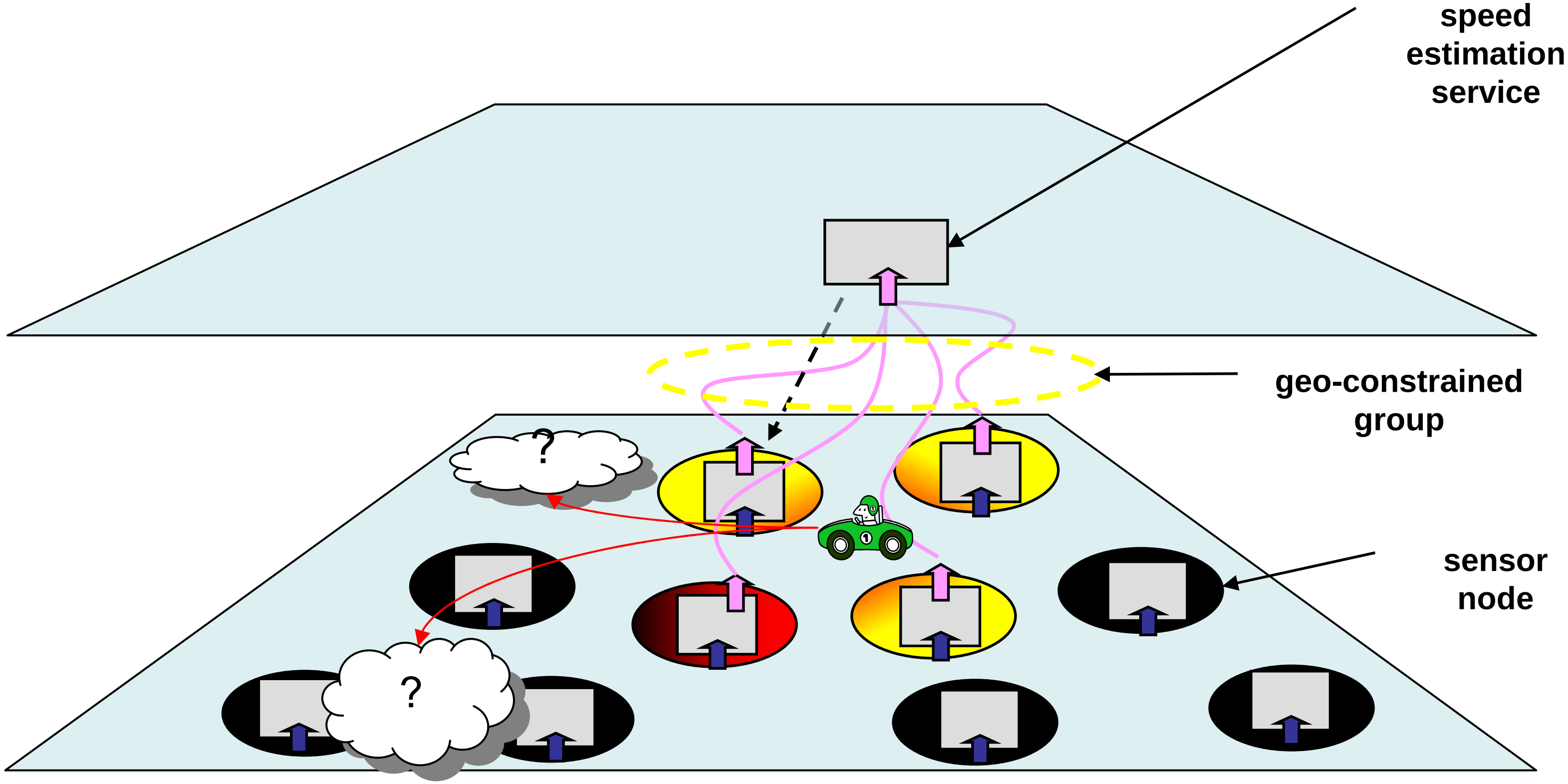
What if nodes fail?

- Contingency plan?
- How many copies do I send around, to maximize odds of not getting lost?



… and world knowledge is incomplete?

- Follow the clouds
- Probabilistic tasking?
- Proactive or reactive?

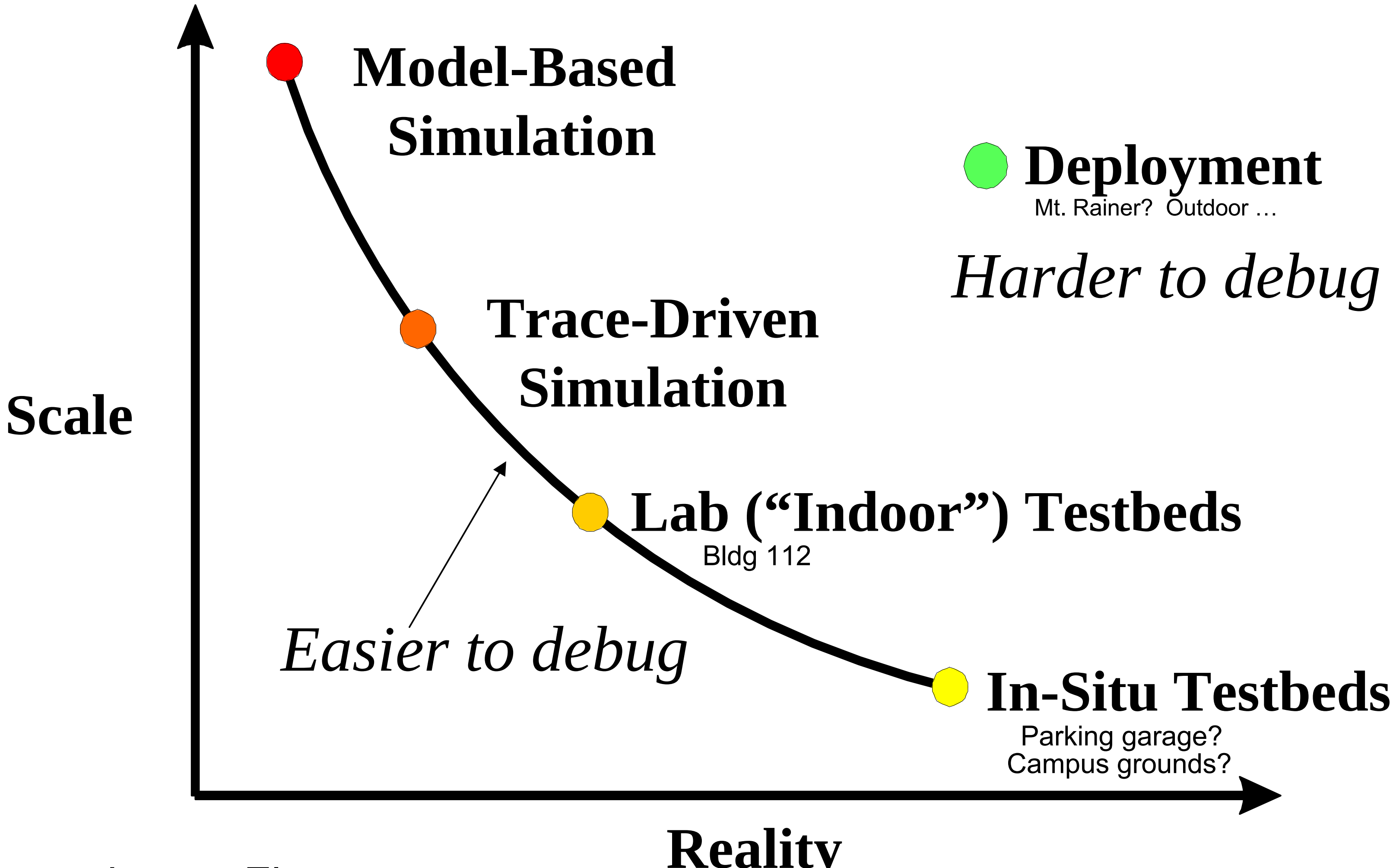


Manage system and event uncertainty

- Event uncertainty
  - Build predictive models of events (e.g., probabilistic)
- System uncertainty
  - Build models of system behaviors (e.g., component failure, processing latency, link quality)

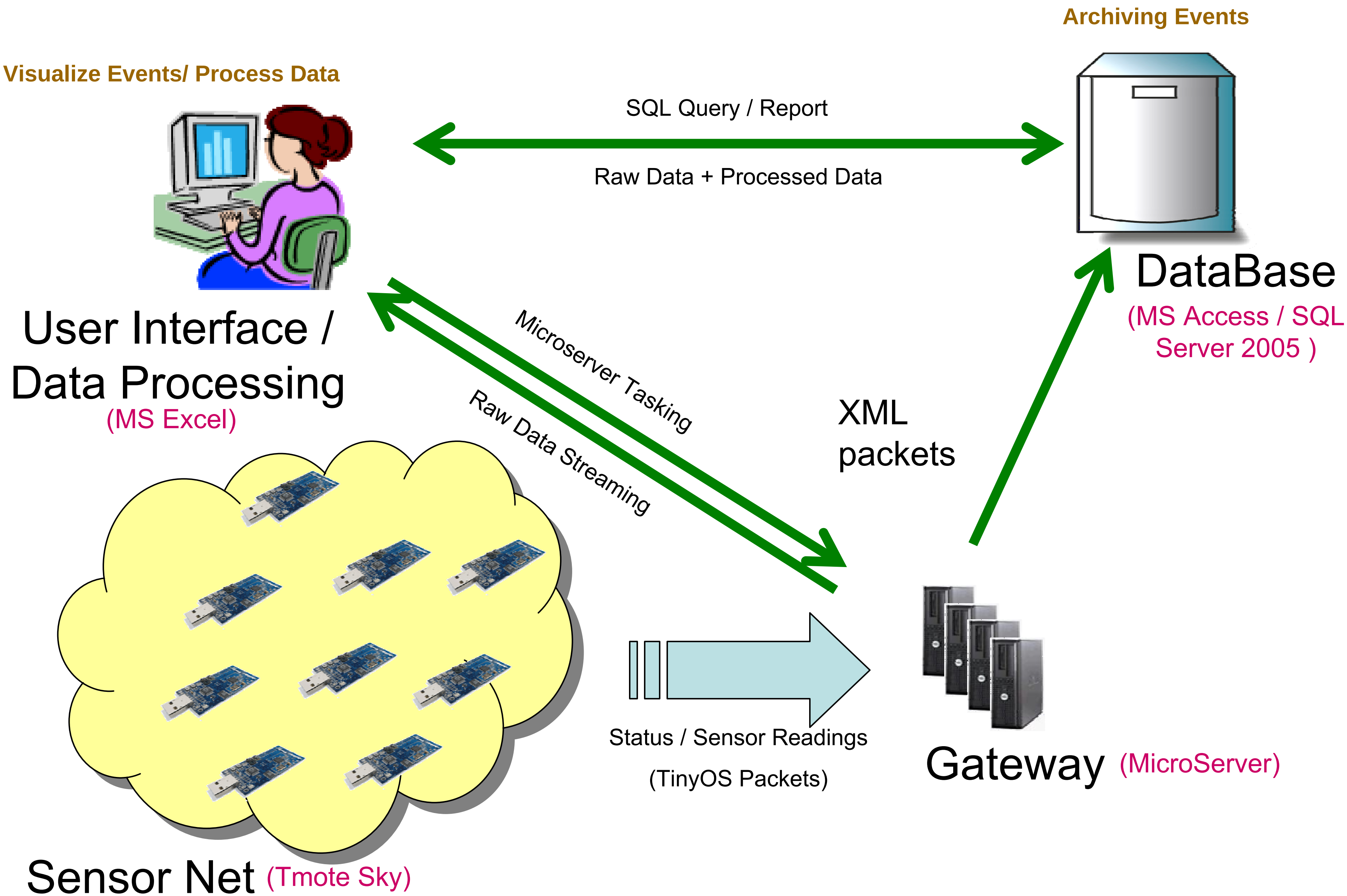
Visibility to systems

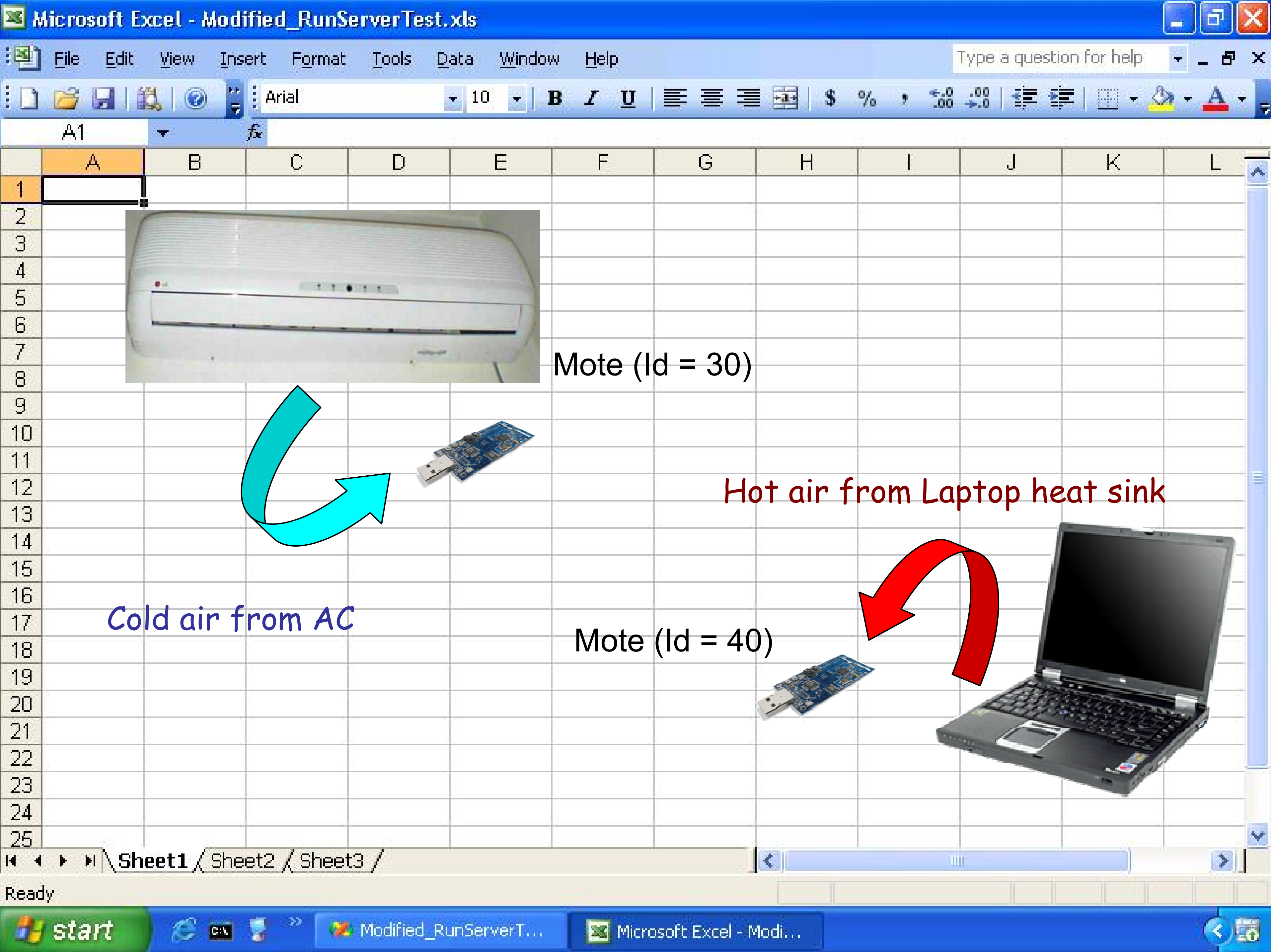
A spectrum allows high-visibility debugging before jumping into low-visibility deployment



Source: Jeremy Elson

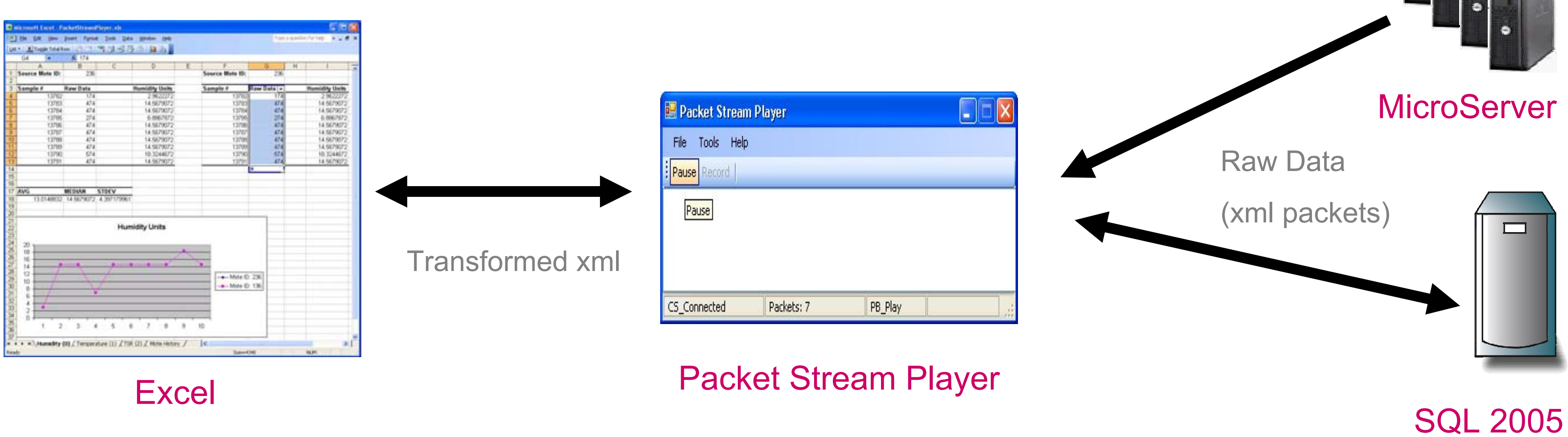
Data Collection

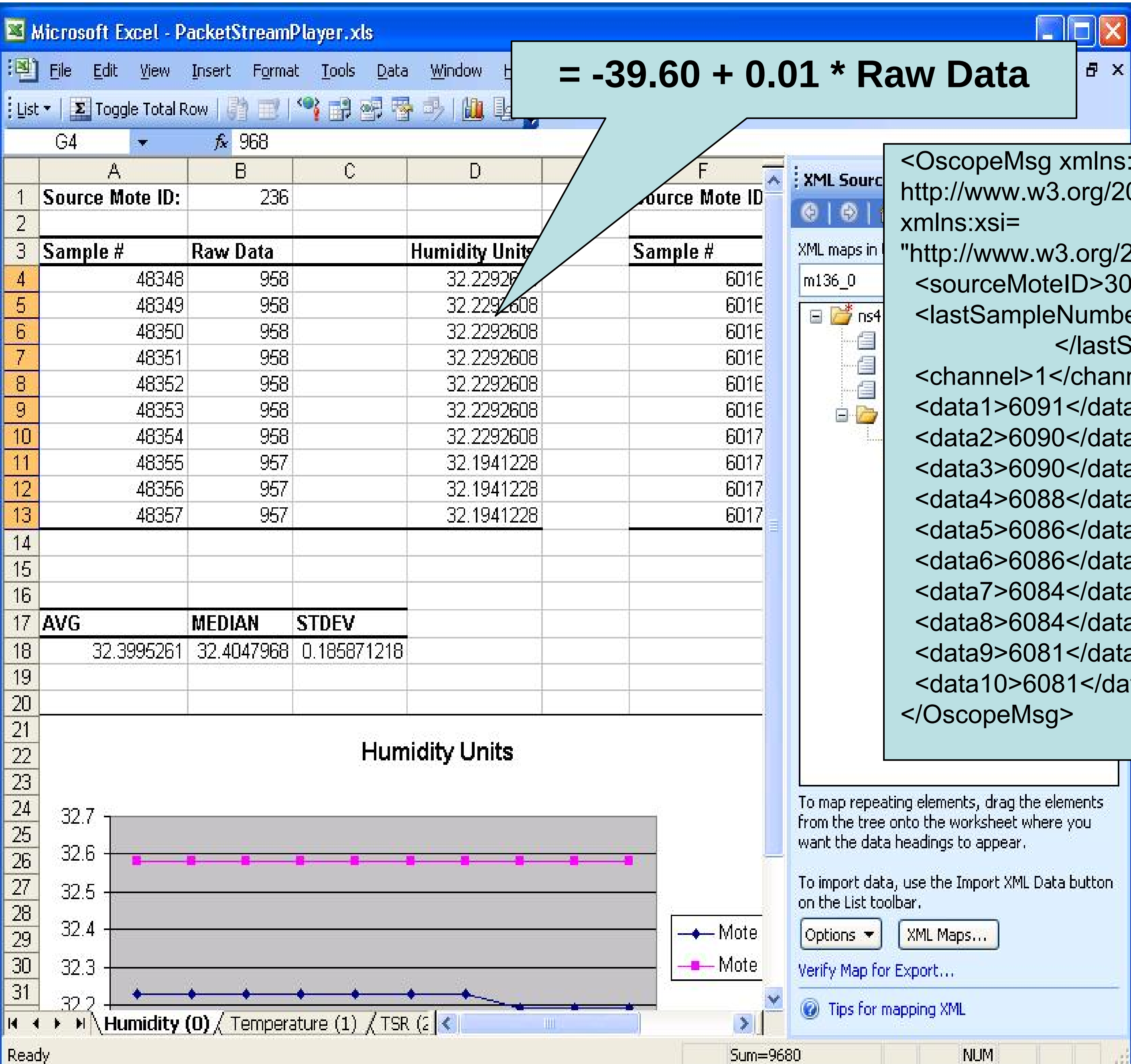



Microsoft Excel - Modified_RunServerTest.xls
File Edit View Insert Format Tools Data Window Help
Type a question for help
Arial
10
A1
Mote (Id = 30)
Hot air from Laptop heat sink
Cold air from AC
Mote (Id = 40)
Sheet1
Sheet2
Sheet3
Ready
start
Modified_RunServerT...
Microsoft Excel - Modi...

Features

- Excel 2003
  - Worksheets
  - Xml Maps
  - Cell Functions
- Packet Stream Player
  - Familiar, simple interface for streaming data
  - Similar to other media-centric players, i.e., Connect, Play, Record, Next, Previous, etc.
- Packet Database
  - Session data
  - Packet data
- Microserver
  - Data provider





= -39.60 + 0.01 * Raw Data

<OscopeMsg xmlns:xsd=
http://www.w3.org/2001/XMLSchema
xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance">
<sourceMoteID>30</sourceMoteID>
<lastSampleNumber>40550
</lastSampleNumber>
<channel>1</channel>
<data1>6091</data1>
<data2>6090</data2>
<data3>6090</data3>
<data4>6088</data4>
<data5>6086</data5>
<data6>6086</data6>
<data7>6084</data7>
<data8>6084</data8>
<data9>6081</data9>
<data10>6081</data10>
</OscopeMsg>

Our tools will be available

- In source code from http://research.microsoft.com/
- Oct. 2005 (alpha)
  - Microserver execution environment (μSEE v0.1)
    - On Windows XP or XP Embedded, .NET Framework 1.1
    - MSTML 1.1 based on MoML
    - Interfacing with MicaZ or Telos (802.15.4 radio packets)
    - Per-node tasking
    - No runtime service sharing
  - Microserver Interaction Console (μSIC v0.1)
    - Generic microserver configuration
    - MSTML editing and data collection
    - Runs locally on the microserver machine
  - Service composition GUI based on Ptolemy II
    - Released as a patch to Ptolemy II 0.5
  - Excel interface for data collection and visualization
    - Require Visual Studio 2005 BETA and .NET Framework 2.0
    - Require SQL Server 2005 BETA
- Dec. 2005 (beta)
  - Move to .NET 2.0, VS 2005, SQL2005 when they are final.

Wrap up: Challenge Problems

- As a community, we need grand challenge problems
- Three focus areas:
  – A community effort to standardize **interfaces** between various sensornet components/layers
  – Build models for **uncertainty** that can be utilized by data and system management
  – Create more **tools**, for sys config/mgmt, data collection and vis, debugging

To probe further

- Some of the relevant papers on the SONGS architecture
  - "Semantic Streams: a for Declarative Queries and Framework Automatic Data Interpretation." Tech Report MSR-TR-2005-45, Microsoft Research, April 2005 (Also Sensys'05 Poster)
  - "Semantics-Based Optimization Across Uncoordinated Tasks in Networked Embedded Systems," EMSOFT 2005, Sept 2005
  - "Towards Semantic Services for Sensor-Rich Information Systems," Basenets 2005, October 2005

- Conferences/Journals
  - ACM Sensys05, San Diego, Nov 2005
  - ACM/IEEE IPSN06, Nashville, April 2006
  - IEEE DCOSS, SECON, EWSN, …
  - New ACM Trans. Sensor Networks, http://www.acm.org/tosn

- MSR Project: http://research.microsoft.com/nec