Networked Embedded Systems and the ZebraNet Project:

Experiences and Challenges

Prof. Margaret Martonosi

Dept. of Electrical Engineering

Princeton University

Acknowledgments

- Yong Wang
- Other ZebraNet students: Ting Liu, Chris Sadler, Pei Zhang
- Other Princeton colleagues: Prof. Li-Shiuan Peh, Prof. Steve Lyon, Prof. Dan Rubenstein
- Other folks: Sushant Jain, Kevin Fall

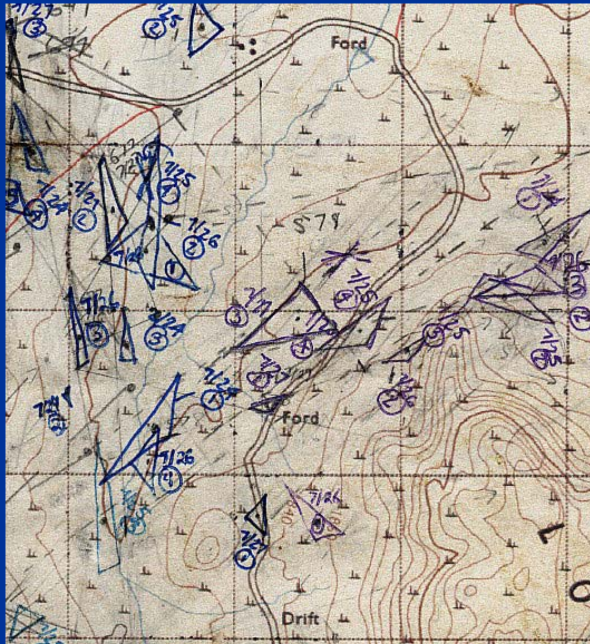
- NSF ITR since 2002

Challenges in Researching Networks of Embedded Sensors

- Constrained optimization on highly-varying apps
  - Good system performance under energy, size, bandwidth constraints
- Fast-changing hardware
  - Difficult to manage software changes, amortize software development efforts
- Inevitably-changing software (Learn->change->learn->change…)
  - Manage on-the-fly updates post-deployment
- "New world" → new metrics for success
  - Autonomy, reliability matter a lot
  - Need good infrastructure to evaluate…

Our work: zebras + …

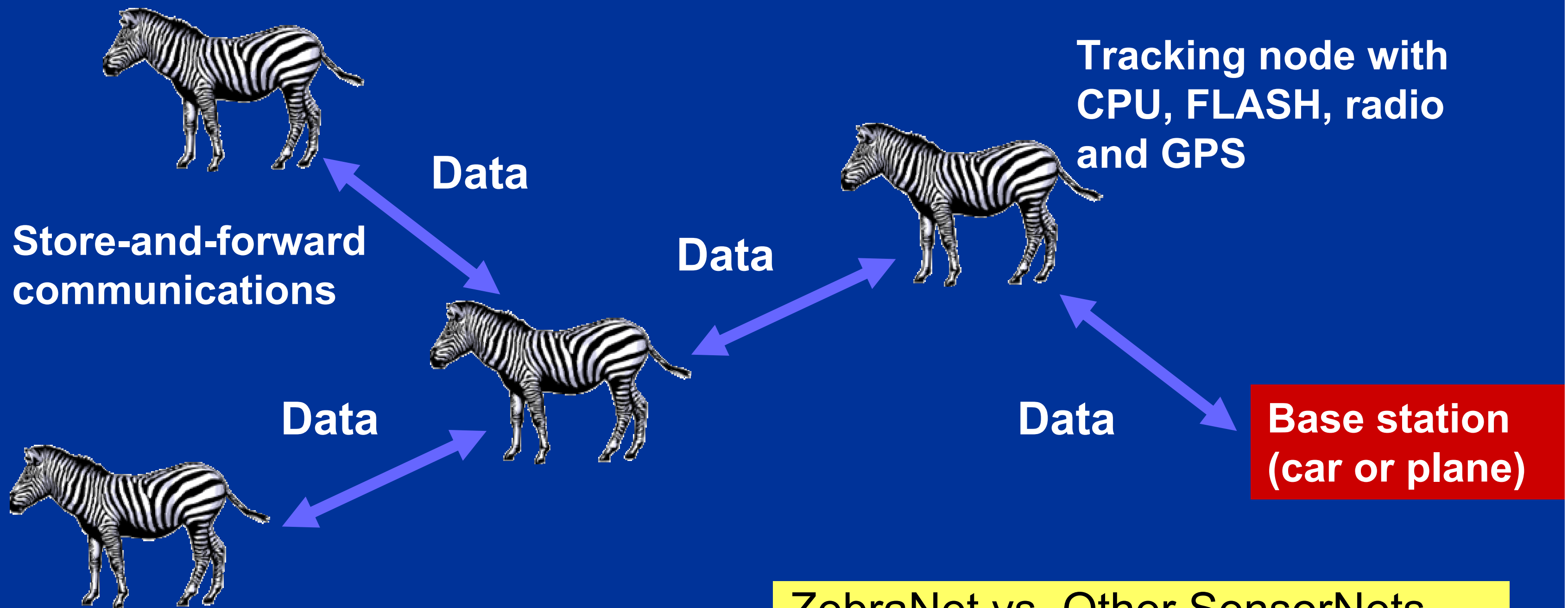
ZebraNet as Biology Research



- Goal: Biologists want to track animals long-term, over long distances
  - Interactions within a species?
  - Interactions between species?
  - Impact of human development?

- Current technology is limited:
  - VHF Triangulation is difficult & error-prone
  - GPS trackers limit data to coarse sampling and require collar retrieval
- Overall, energy and info retrieval are key limiters
- Peer-to-peer offers opportunity to improve

ZebraNet as Computing Research



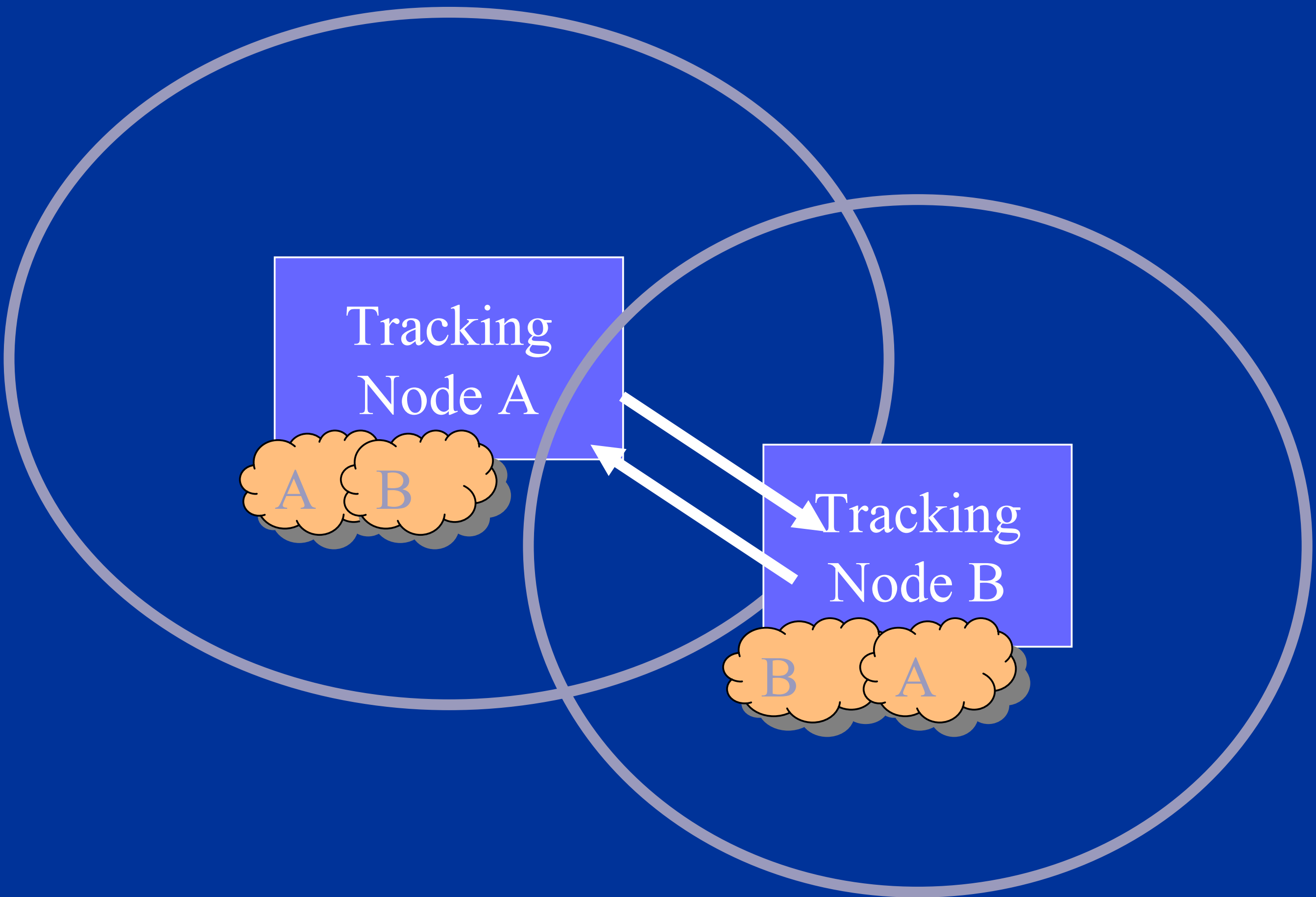
ZebraNet vs. Other SensorNets

- All sensing nodes are mobile
- Large area: 100's-1000s sq. kilometers
- "Coarse-Grained" nodes
- GPS on-board
- Long-running and autonomous

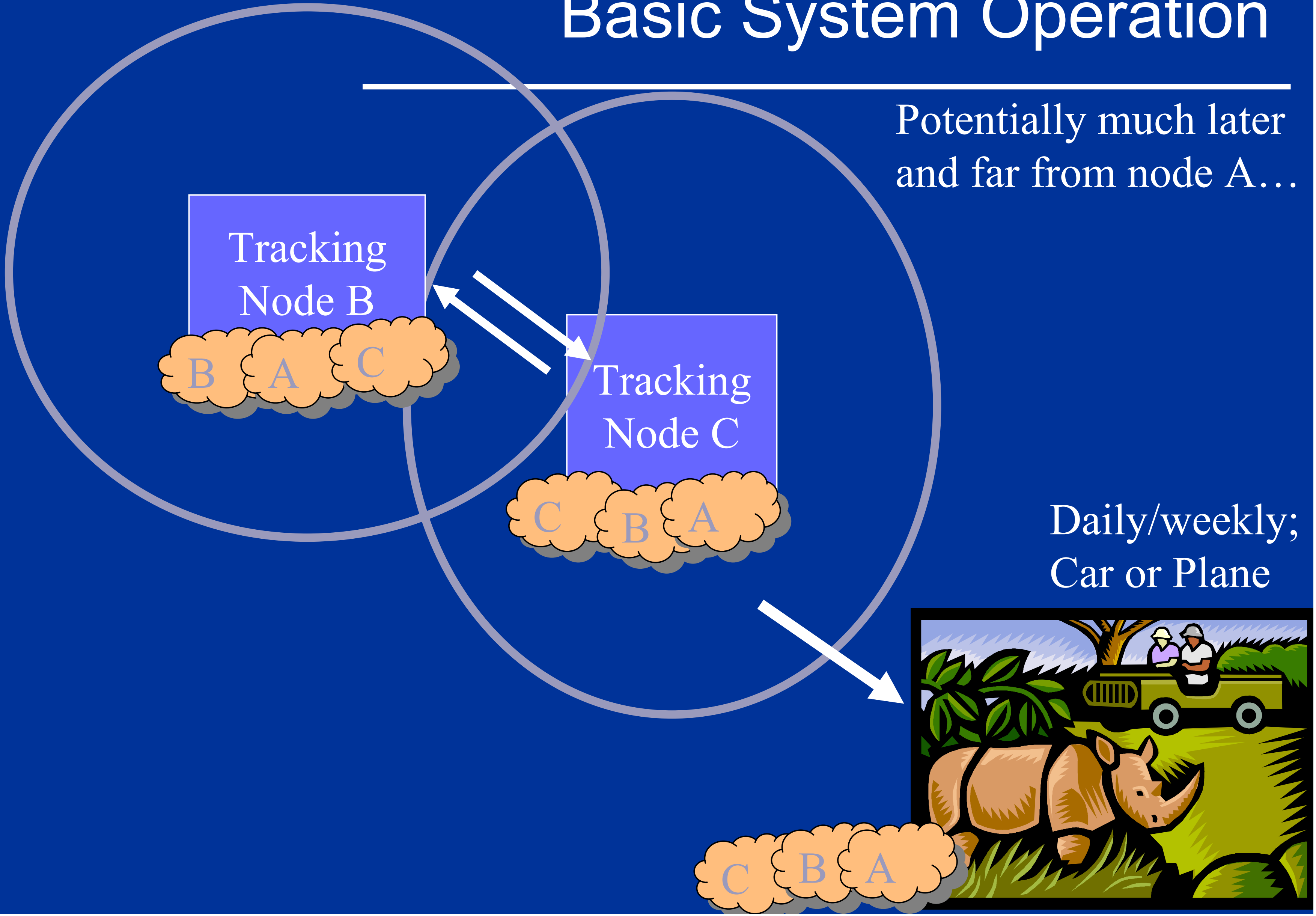
Research Questions

- Protocols and mobility?
- Energy-efficiency?
- Software layering design?

Basic System Operation



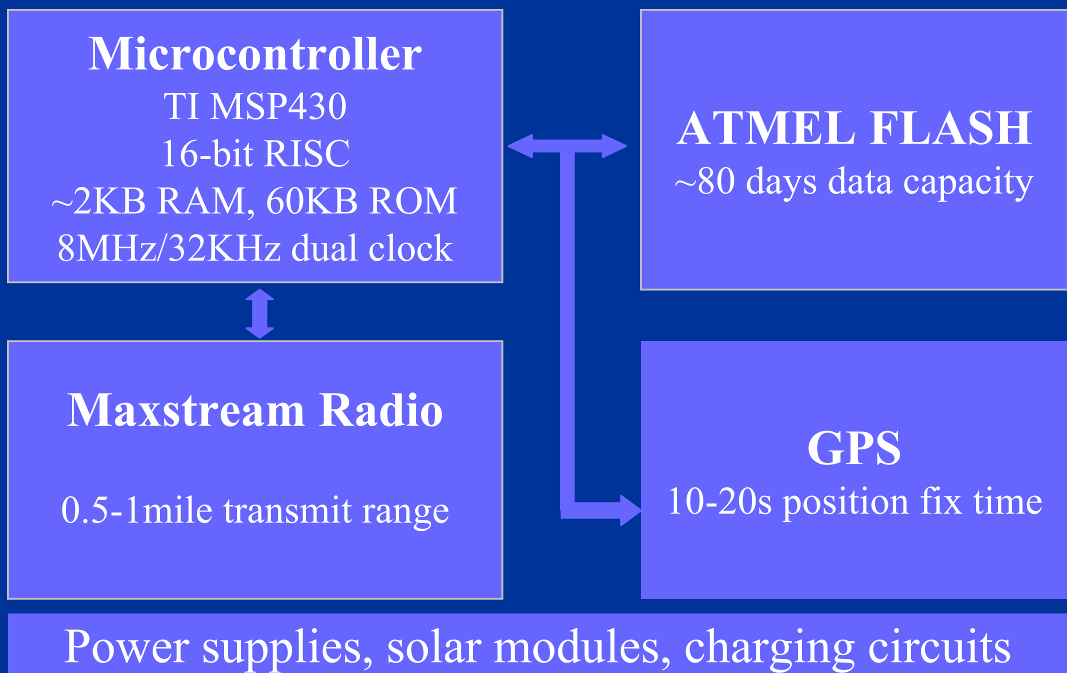
Basic System Operation



Abstraction Layers and Research Challenges

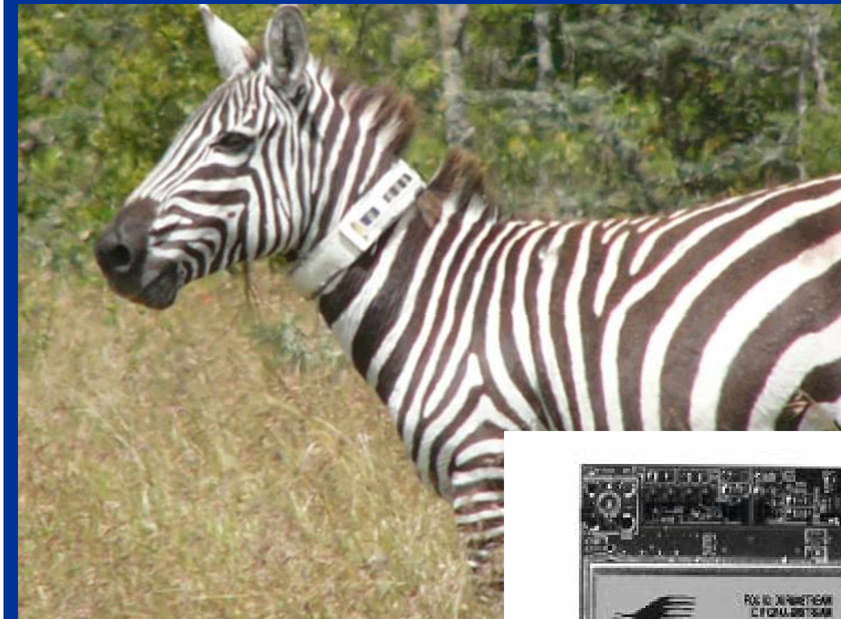
- Impact of app constraints and needs on design?
  - energy, weight, lifetime, datarate
  - latencies, data rates, accuracy
- Managing constant change of applications and hardware
- Info passing between interface layers?
- Portability across apps?

ZebraNet Hardware Design



Mode	Power
32Khz CPU	9.6 mW
8MHz CPU	19.32 mW
8MHz w/ GPS	568 mW
8MHz + radio xmit	780 mW
8MHz + radio rcv	312 mW

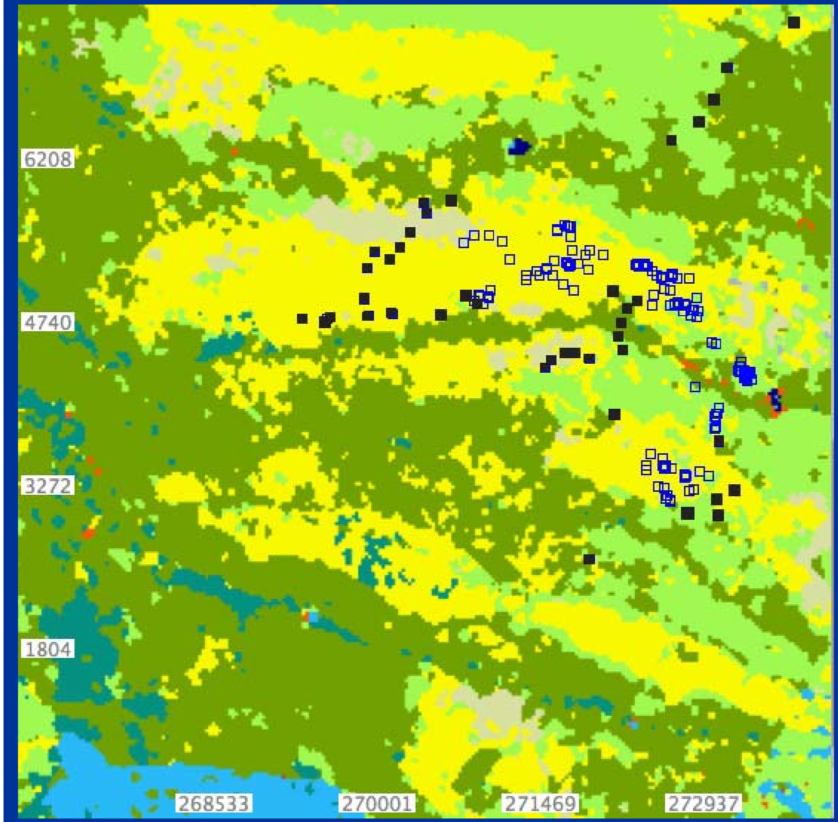
January, 2004 ZebraNet Hardware



January, 2004: Initial test deployment

- Eat/sleep/work at Mpala Research Centre near Nanyuki Kenya
- Deploy collars at Sweetwaters Preserve, also near Nanyuki Kenya

First Deployment Results



Biology

- Zebras affected by collar first day (head shaking) but little thereafter
- First night-time zebra movement data: animals appear to explore more wooded areas and gullies at night

Engineering

- Radio range: <1km in final collar packaging,
  - Disappointing vs. NJ tests
  - Answer: Choose cobbler well…
- Communication protocols: generally worked as plans, although duplicated ACK packets improved their performance

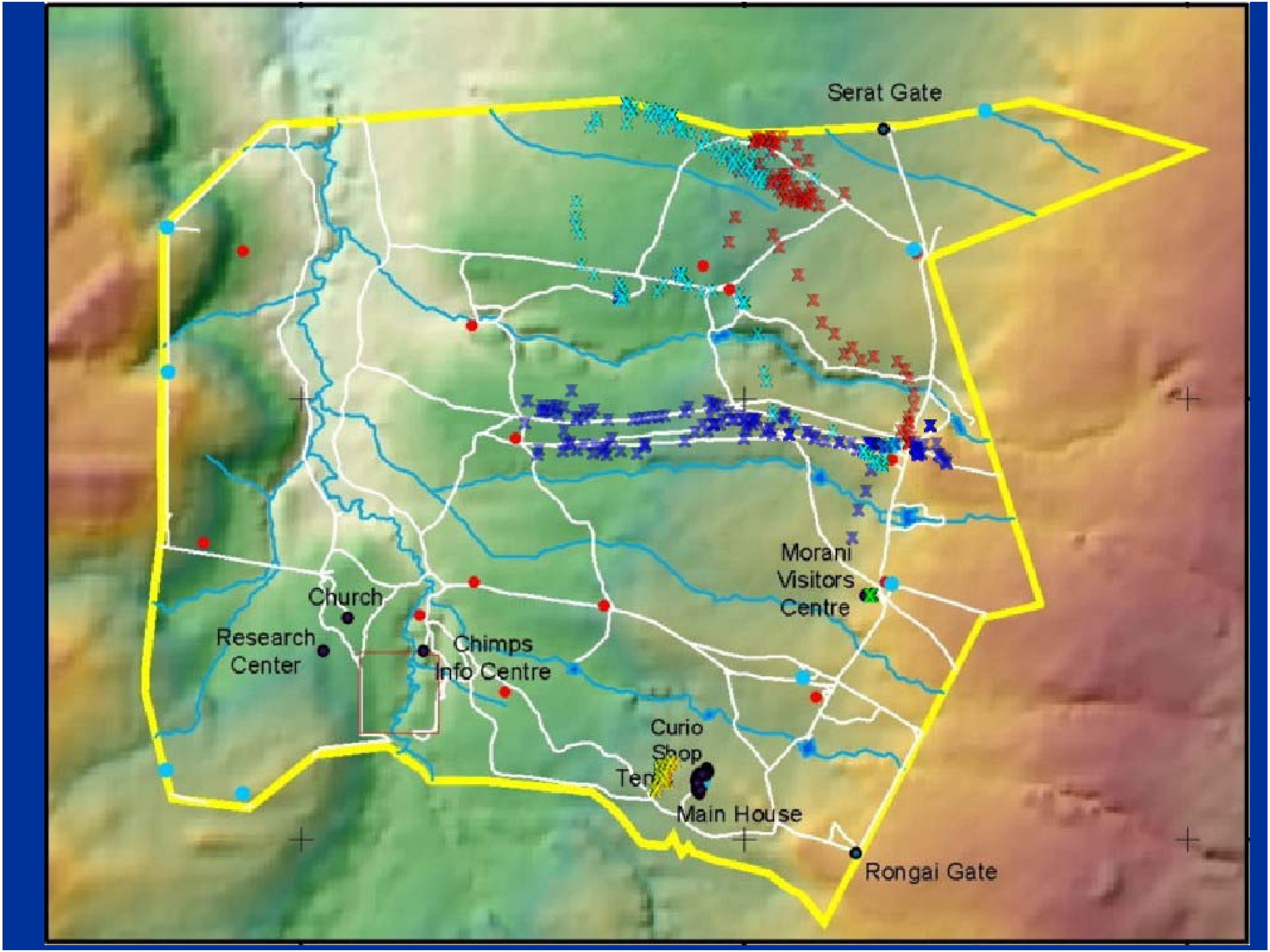
15 months later…

- Collar on Zebra for roughly 1.25 years
  - Hot sun, tough weather.
- Zebra attacked/killed by a lion (not unusual, probably not related to collar)
- How did the collar do?

Next Steps

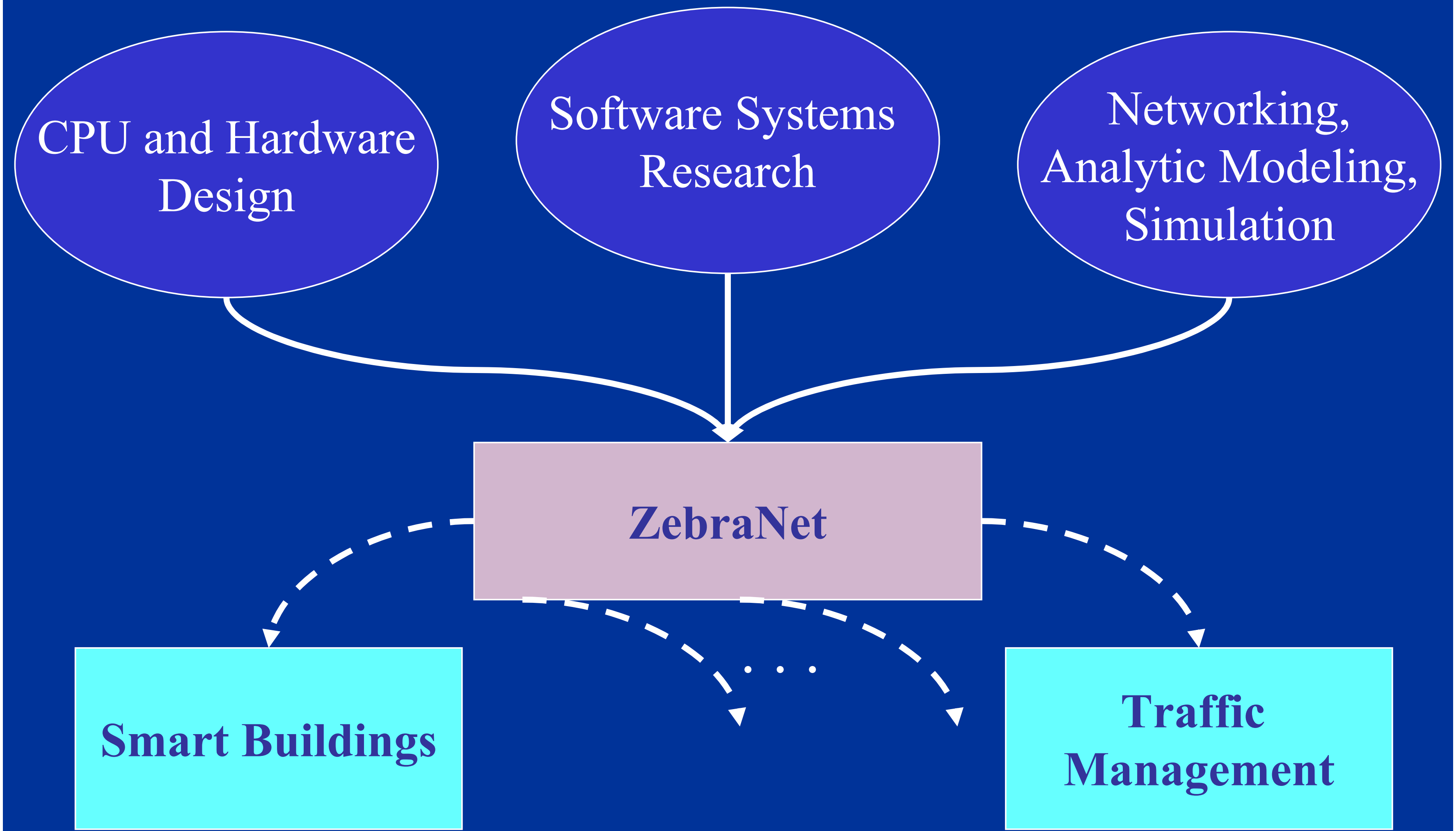
Second Deployment: June/July, 2005 (NOW!!)

- Amorphous silicon solar cells with Tefzel coating
- Leather rather than butyl rubber collar
- New version of msp processor
- New radio
- New GPS
- New collar design
- Improved code (latencies, interrupt handling….)


Serat Gate
Morani
Visitors
Centre
Church
Research
Center
Chimps
Info Centre
Curio
Shop
Ten
Main House
Rongai Gate

Beyond zebras…

Systems research feeding into mobile wireless application domains…



Challenges: Another look…

- Lack of stable application drivers on which to experiment
  - Applications very much on-the-move ri…t now. Hard to pin down general …cteristics
  - … stability in underlying hardware a… …ftware
  - What are the default hardware and software platforms to assume?
  - How to support their ongoing change?
- Other: Radio range, security, …
- Lack of good experimental infrastructure
  - Testbeds
  - Simulators

Unlikely to change

Look at these first!

Unlikely to change

Unlikely to change

Likely to change

Changing Applications, changing platforms

- Ongoing change is the mark of all dynamic/interesting areas of computing
- Rather than wait for change to slow down, embrace it and look for ways to manage/encourage it

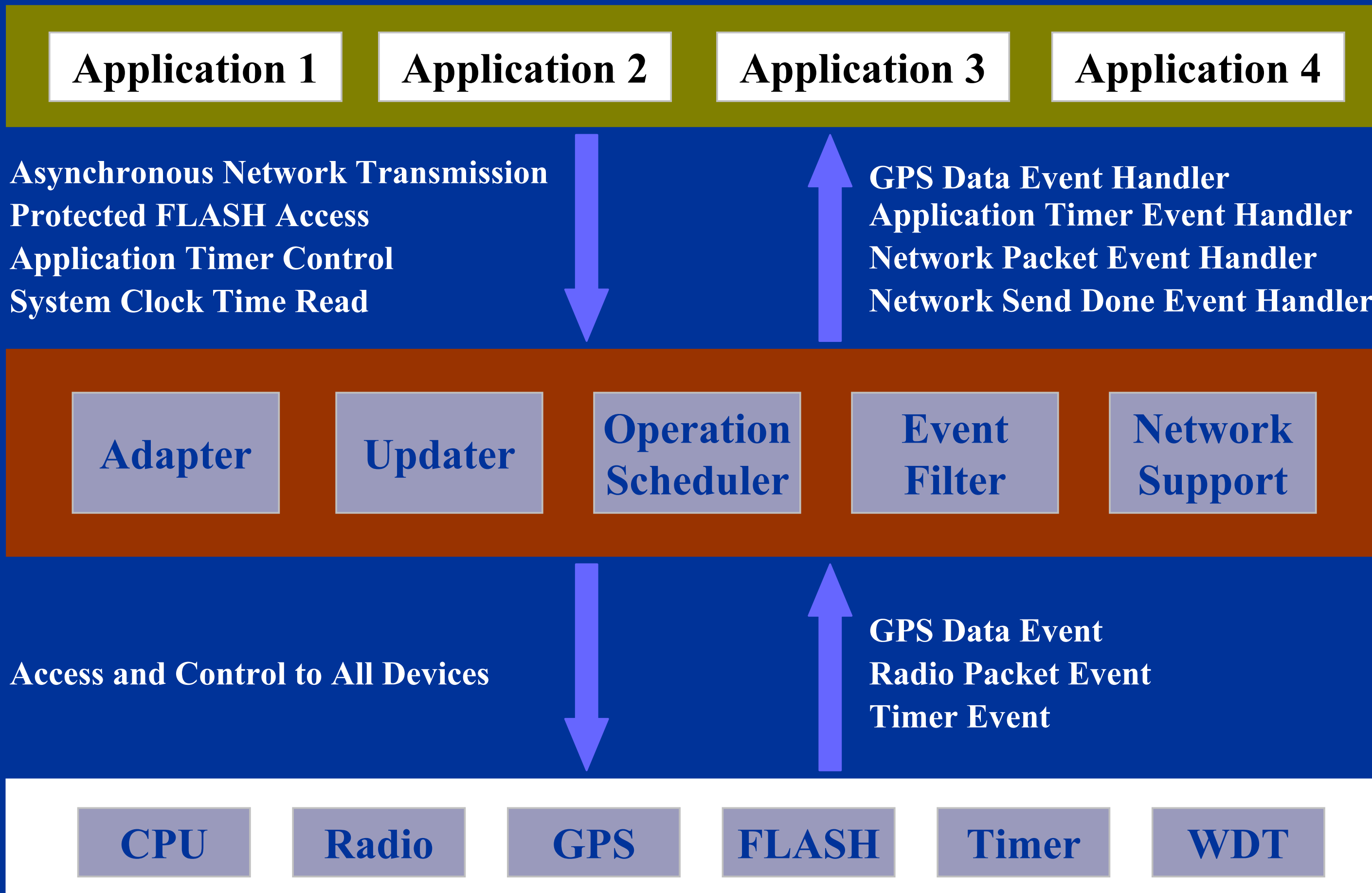
- Abstraction layers
- Dynamic and model-driven adaptation

Abstraction Layers and the Impala OS

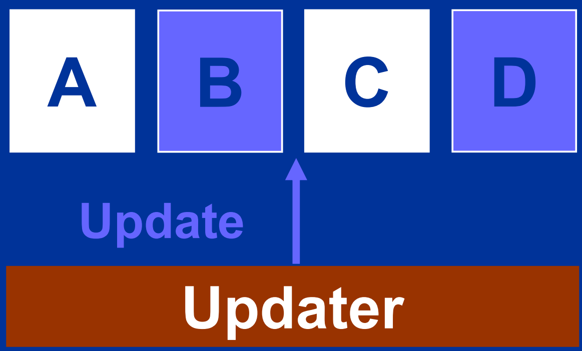
Abstraction Layers and Research Questions

- Portability across apps?
- Impact of app constraints and needs on design?
  - energy, weight, lifetime, datarate
  - latencies, data rates, accuracy
- Info passing between interface layers?

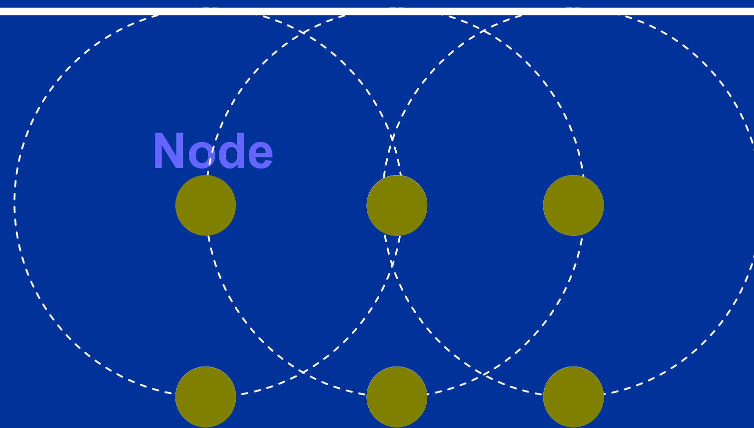
Impala Middleware/OS Layers



Impala Code Updates



On a single sensor node



Full network

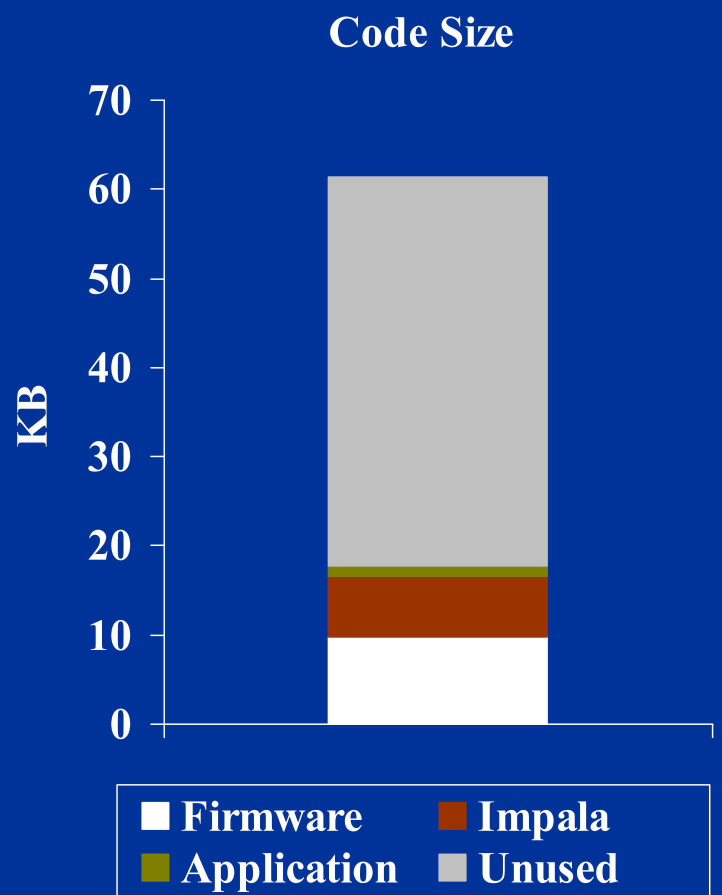
ZebraNet Characteristics

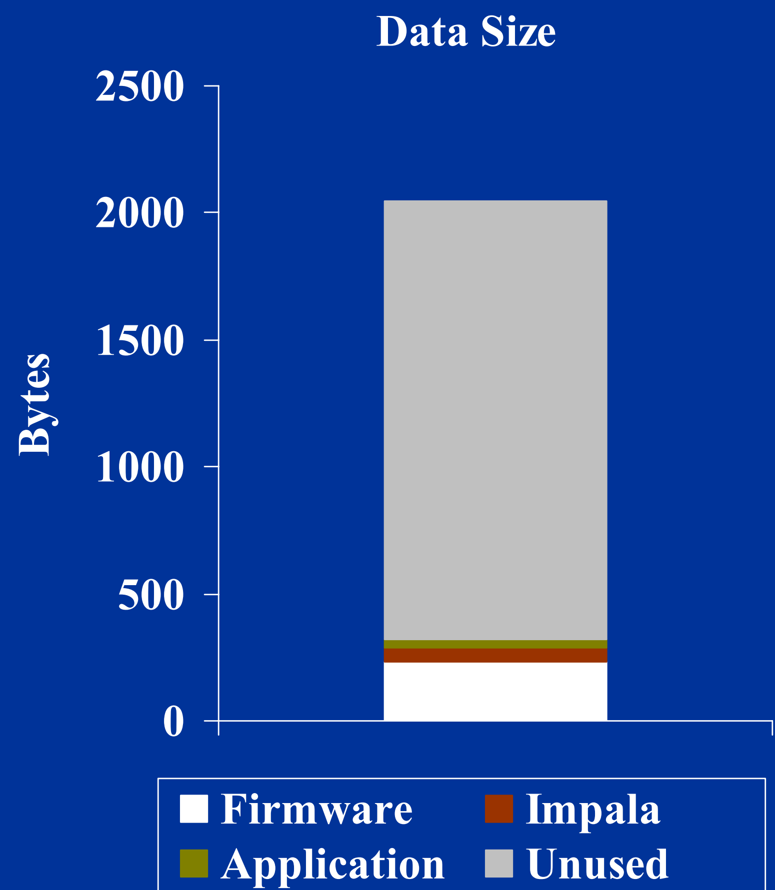
- High Node Mobility
- Constrained Bandwidth
- Wide Range of Updates

Design Implications

- Incomplete Updates
- Updates vs. Execution
- Out of order Updates

Code and Data Size: Memory Footprint of Impala Layers



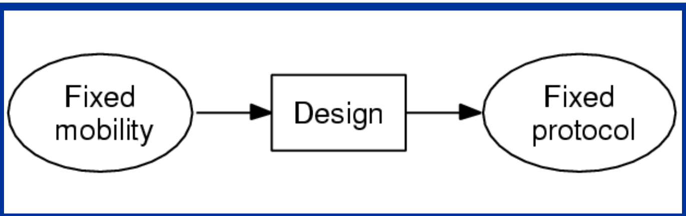


Dynamic & Model-Driven Adaptation

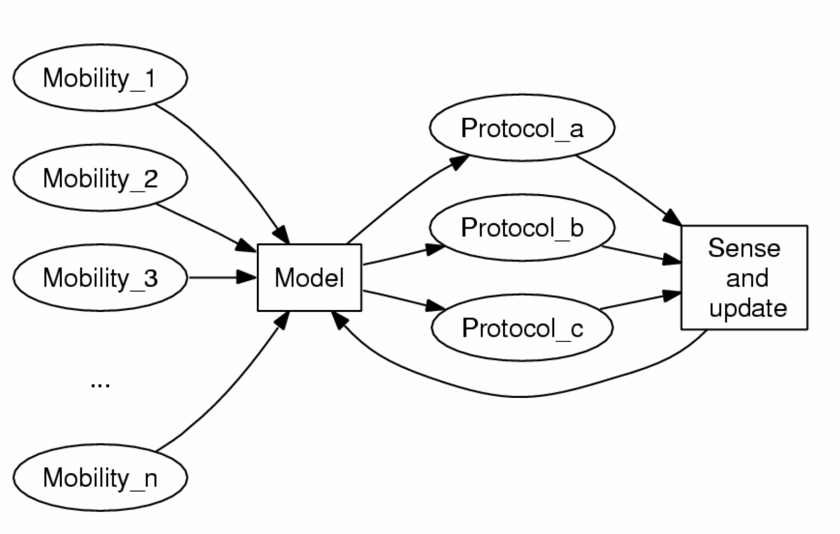
Adjusting to uniquely mobile characteristics

- Mobile networks have gained a lot by building off techniques from static wired networks.
- But they have lost a lot too
  - Their ability to do broadcast-for-free is rarely leveraged as fully as it could be.
  - Recognizing disconnected or disrupted optimization and customizing for it
  - Ability to adapt to changing network and exploit it intelligently

Model-based Mobility-Adaptive Protocols

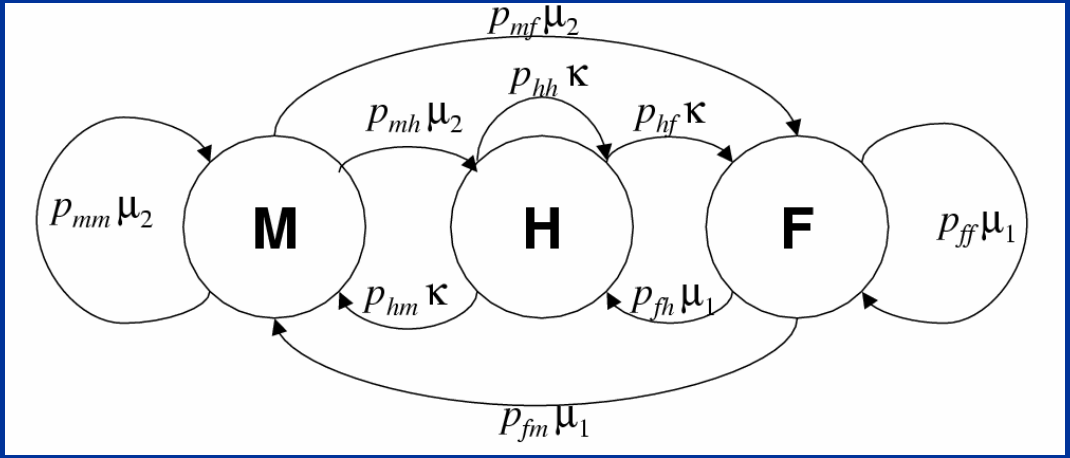


Traditional Approach



Our Approach

Example:
Markov Model based on route cache lifetimes



Markov route cache model

Traffic Rate	Original (DSR)	Markov approach	% latency improvement
2 pkts/s	6.2	5.2	16%
4 pkts/s	4.1	3	27%
8 pkts/s	2.7	2.1	22%

With Yong Wang, LS Peh, under submission

Tolerating Sparse and High-Disruption Wireless Networks

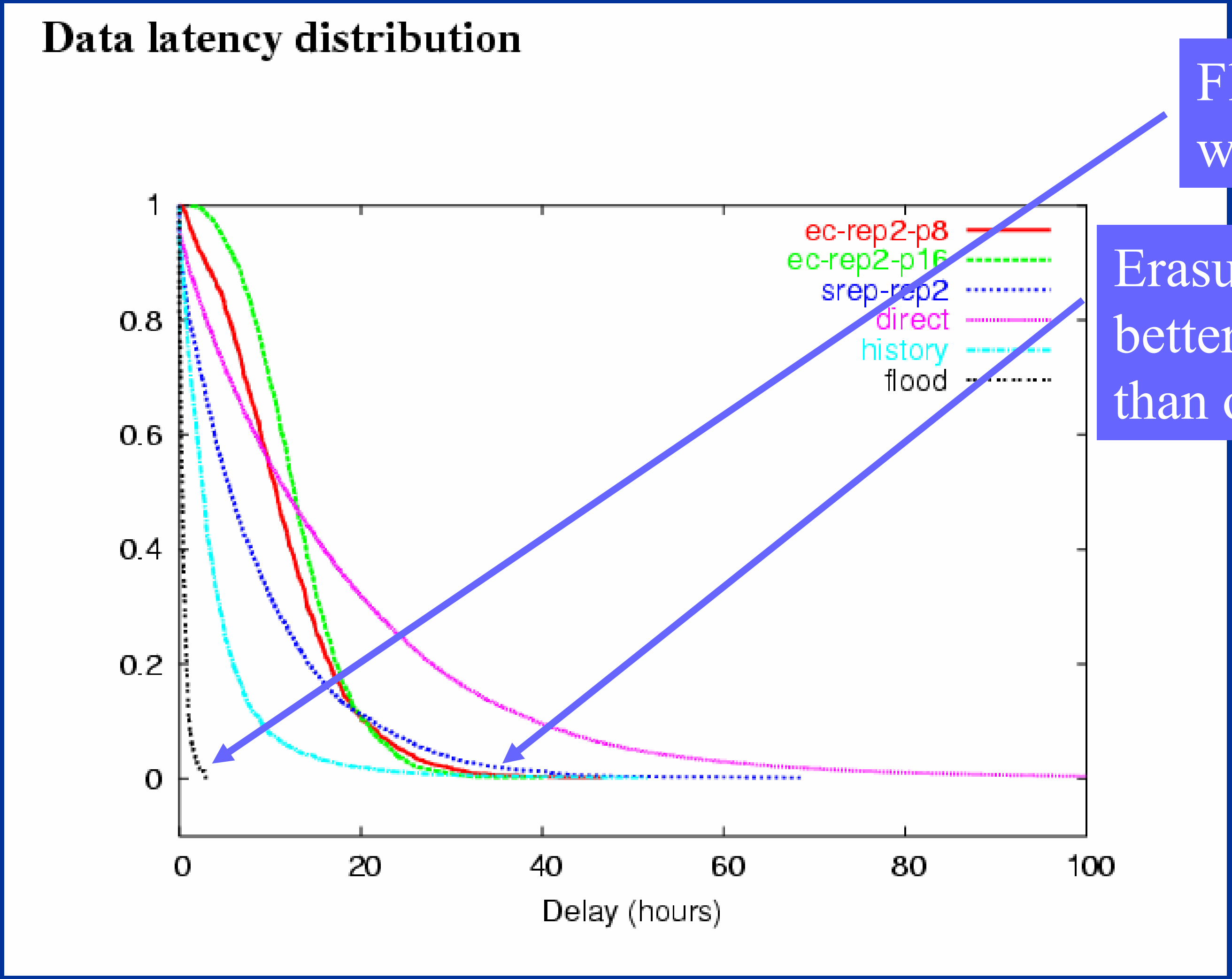
- What if a source->dest route never exists? Or rarely exists?
- Need to engineer more opportunistic approaches
- Model-based: if you can learn about mobility patterns, you can exploit this knowledge to know how to route
- Cover-your-bases: If not, use erasure-coding to reduce risk

Erasure Coding for Disruption-Tolerant Wireless Networks

- Basic Approach
  - For given “replication budget”, replicate via erasure-coded packets, rather than basic redundancy.
  - Partial packet delivery can be used to reconstruct original message
- Intuition
  - Erasure-coding reduces “risk” due to outlier bad links
  - Erasure-coding based approach has the lowest worse-case pkt delivery latency

With Yong Wang, Sushant Jain, Kevin Fall, Sigcomm WDTN 2005

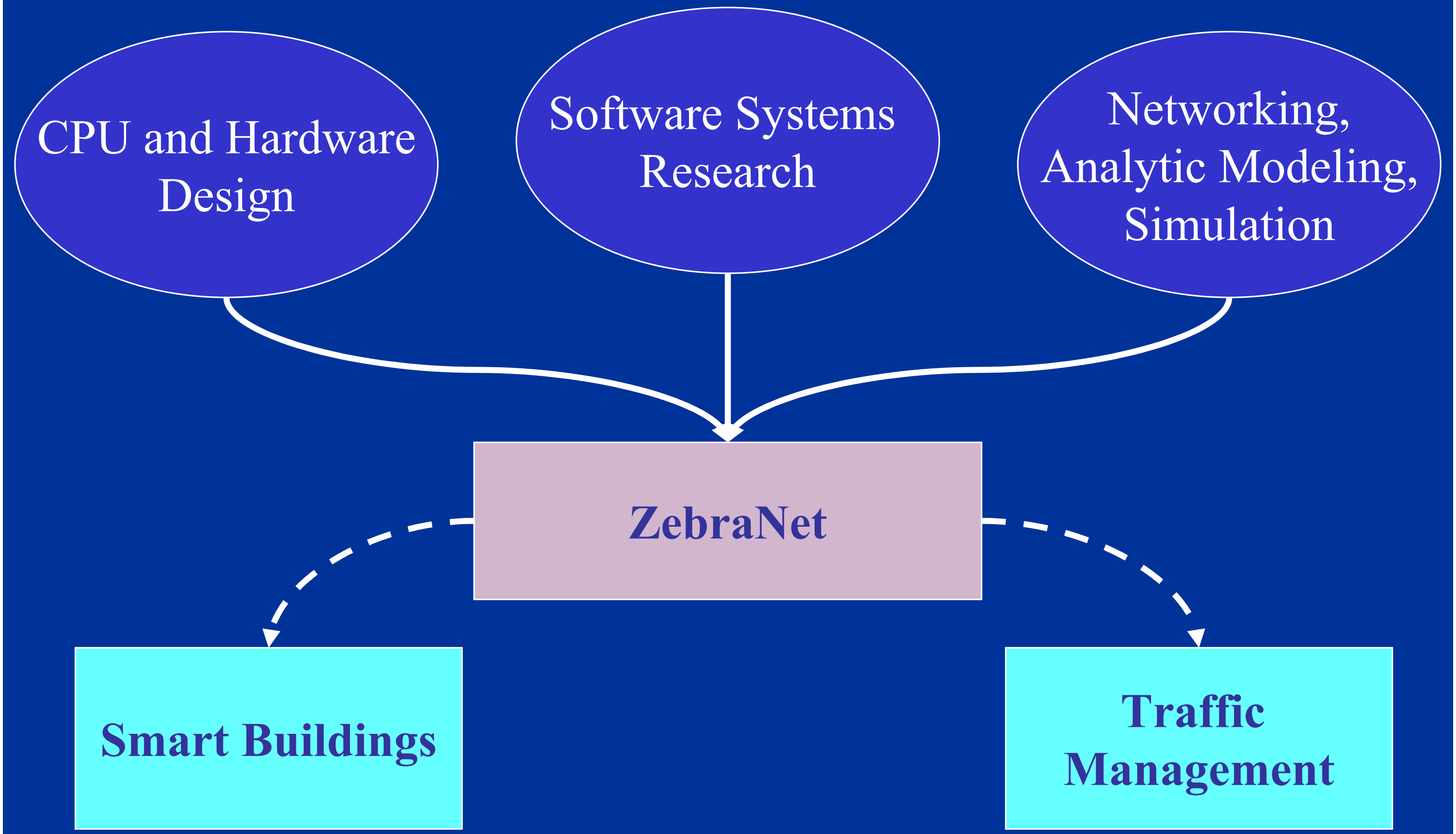
Packet delivery latency for Erasure-Coding and other approaches



Flooding has best worst-case time

Erasure-coding: better worst-case than other replication

Beyond zebras: Final challenge is leveraging lessons/infrastructure over many apps…



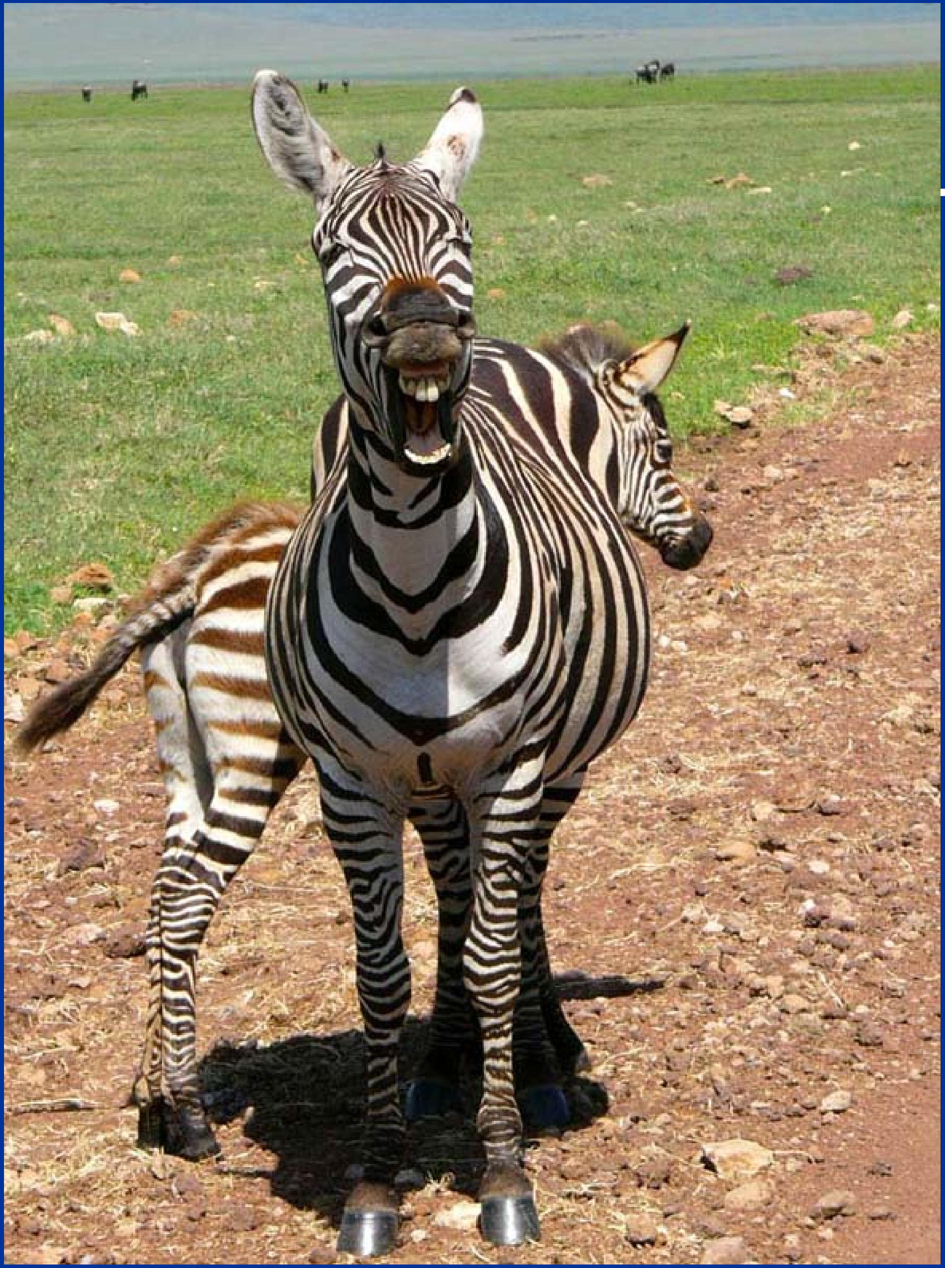
Summary and wishlist

- Better built-in infrastructure for real-system wireless measurements
  - Testbeds, especially large-nodecount and unique physical placements
- Increased availability of “real-world” mobility traces
- Continued and increased research focus on:
  - composition of algorithms into real systems
  - and on how real-systems behaviors impact theoretically-good algorithms
  - Portable hardware/software interfaces and clean models for managing complex parallelism in these systems

The Princeton ZebraNet Project: Mobile Sensor Networks for Wildlife Tracking

ZebraNet Folks at Mpala Research Centre, near Nanyuki, Kenya. January 2004.

- *Grads: Pei Zhang, Chris Sadler, Ting Liu, Ilya Fischoff,* Yong Wang, Philo Juang.
- Profs: *me, Dan Rubenstein, Steve Lyon,* Li-Shiuan Peh, Vince Poor.
- Undergrads: Julie Buechner, Chido Enyinna, Brad Hill, Kinari Patel, Karen Tang, Jeremy Wall
- Departments of EE, CS, and Biology at Princeton
- Funded by NSF ITR since 9/2002

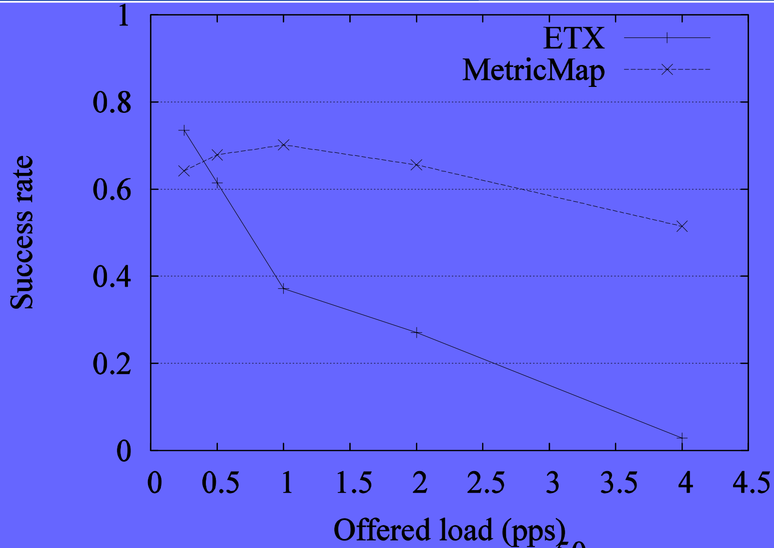


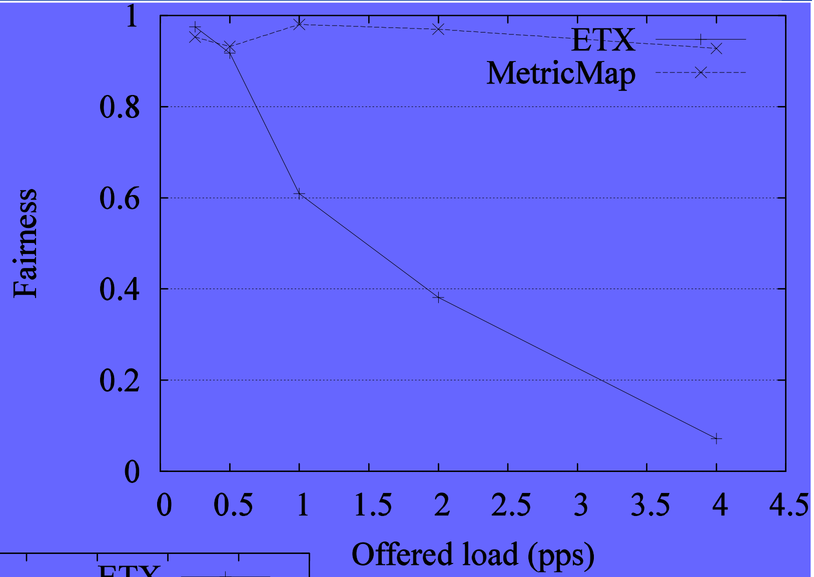
More questions?

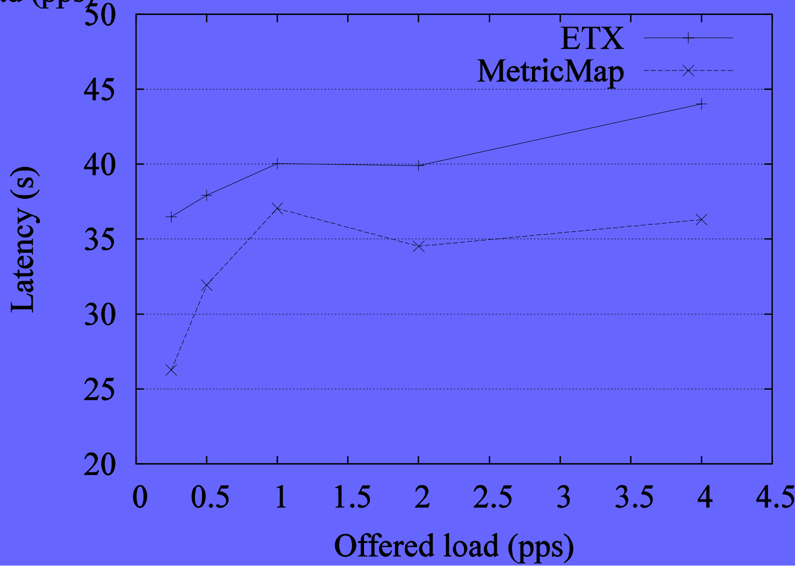
For more info, see papers:
ASPLOS02, PPOPP03, Mobisys04,SenSys04

… and my webpage
www.princeton.edu/~mrm

Preliminary results







Why routing in disrupted network is hard (I)

- Node mobility is very unpredictable and highly varying for many emerging applications
  - Zebras move in phases with very distinct mobility characteristics
  - Data mule links may have very long delays
  - No model available for these emerging movement traces yet
    - Interesting and challenging for the modeling and measurement community
    - Hard to generalize
      - Statistics models alone may not be suitable since mobility shows very different features in different states
    - From the system perspective, such model/measurement is also great importance to in understanding the design space and drive new designs

Why routing in disrupted network is hard (II)

- Communications maybe asynchronous and contemporary due to intermittent connections
  - Traditional routing is based on end-to-end connection
  - Store-and-forward is the most widely used forwarding algorithm in current DTNs
  - Based on various replication schemes

    _flooding _direct

    _simpleReplication _history/probabilistic
  - Performance heavily dependent on the relays selected
    - Delay
    - Data delivery rate

Why routing in disrupted network is hard (III)

- Constrained device and varying quality links
  - Nodes are energy constrained
    - Cannot afford expensive operations like period broadcast or neighbor probing
    - Especially true when workload is high
  - Wireless bandwidth is limited
    - Radio range is very limited in our testbed environment
  - Strong interference and unpredictable quality
    - Radio transmission quality varies with environment, traffic workload, to name a few
    - No golden rule that a link is always good
    - Shortest-path based on link hops may not be perform well

Why routing in disrupted network is hard (IV)

- Traffic is application-dependent and unpredictable
  - When traffic workload increases, such as from a demand for high-sampling rate in wireless sensor networks, some of the underlying assumptions used by current routing may break
  - One example: ETX routing
    - The method to measure link quality by snooping data packets or sending probing packets fail.
    - Effective adaptation requires situation-aware so that mechanisms that work in current scenario will be used.

Why routing in disrupted network is hard (V)

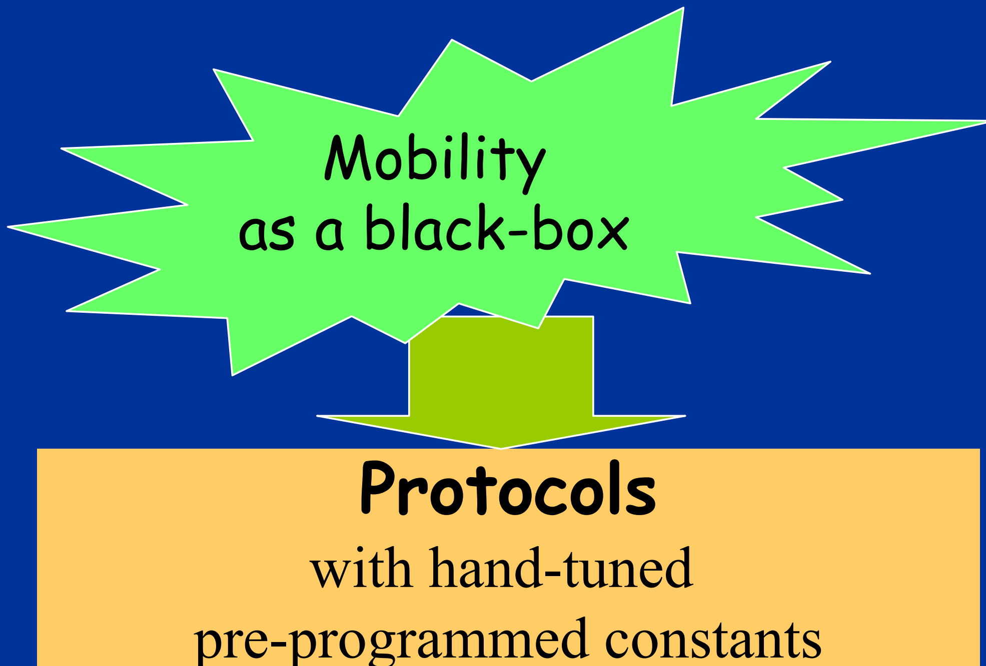
- Evaluation
  - Lack of *testbed* and *simulation* tools
  - Partly due to the lack of models for such emerging applications
  - No open testbed for widely use, like motelab or mirage in static sensor network area
    - It will be cool to have a live znetlab for experiment

Our Approach: Put intelligence into decision-making

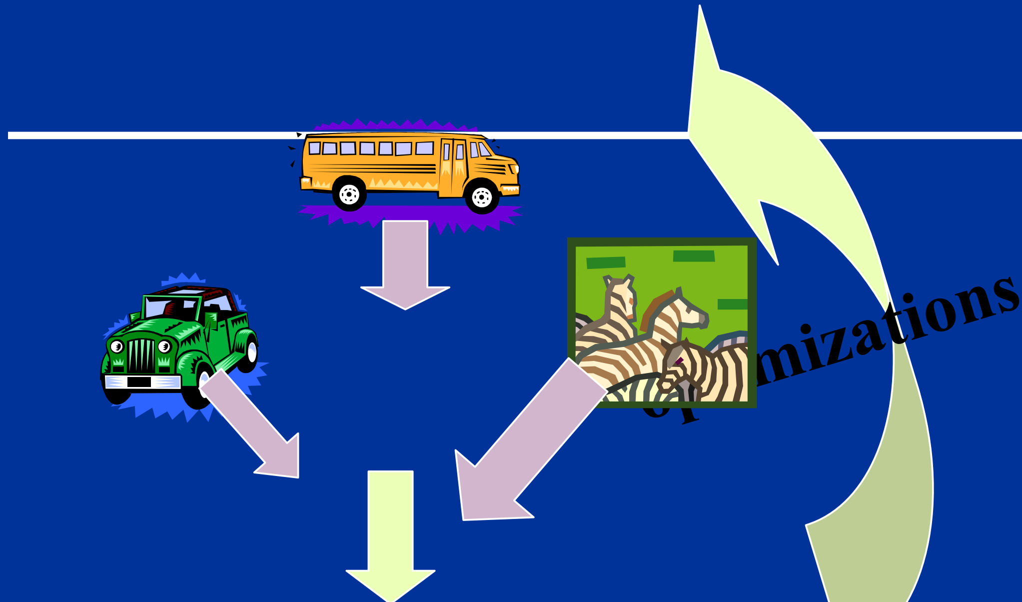
- Smart adaptation based on accurate understanding of mobility, traffic and link quality measurement, etc.

- Ia. *MARio*
- Ib. *Model-based routing*
- II. *Erasure Coding based DTN Routing*
- III/IV. *MetricMap*

Mobility Adaptive Routing



Traditional Approach



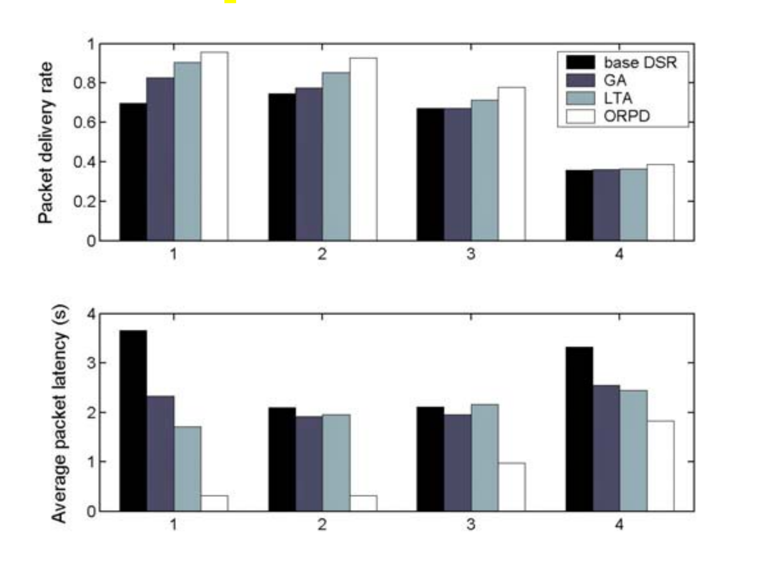
Route lifetime abstractions (RLA)

MARio

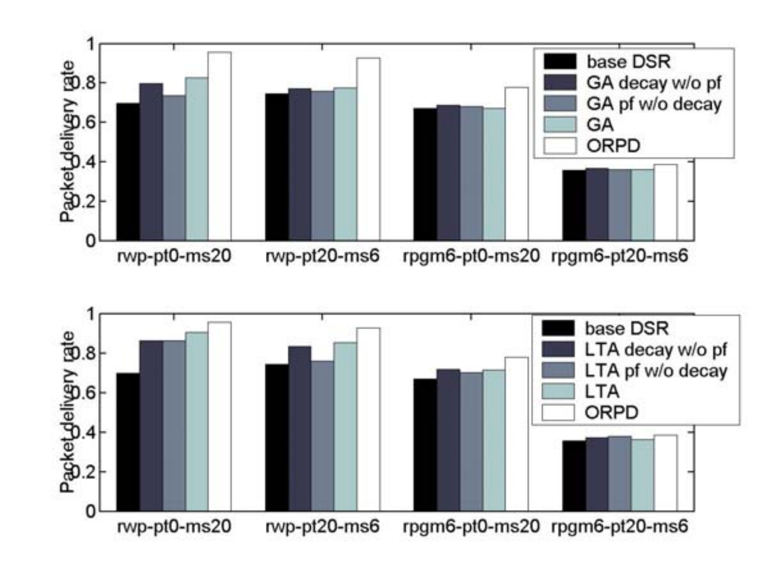
with mobility-adaptive strategies
(this is different from recently proposed ***mobility-controlling*** approach)

Our Approach

Raw performance



Performance breakdown

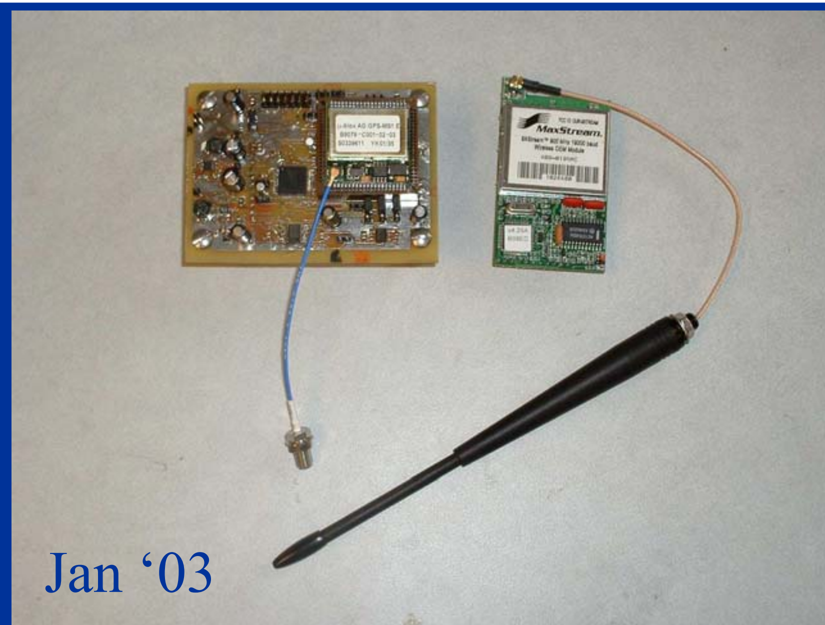


- We use DSR as the base protocol and implement MARio atop it
- The performance comparison is among the following protocols:
    - Base DSR, Offline (GA), Online (LTA) and ORPD (Oracle)
- Performance breakdown demonstrates operation (prefetch/decay) accuracy:
    - Prefetch only, Decay only, Prefetch and decay

With Yong Wang, LS Peh, Mobihoc poster 2004, mc2r 2005

ZebraNet Hardware: Time-Lapse View…

Aug '02

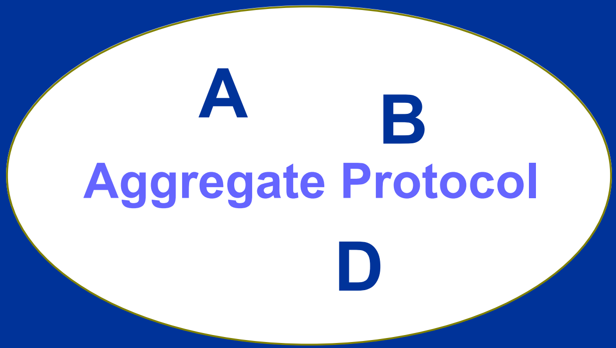


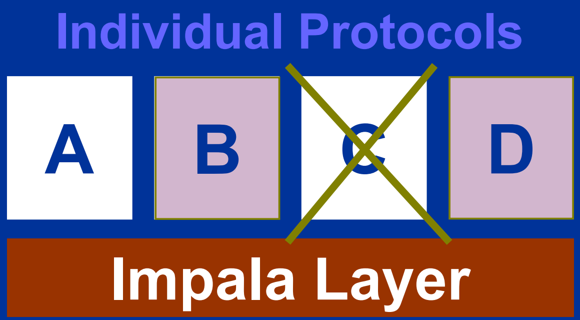
Jan '03


Aug '03


Oct '03

Impala: Middleware/OS Support for Application/Protocol Modularity





Monolithic Approach:

- Source code hardwired to particular situations
- App responsible for adapting situation/version choices
- Difficult to debug, maintain
- Difficult to update on the fly
- Difficult for other apps to reuse

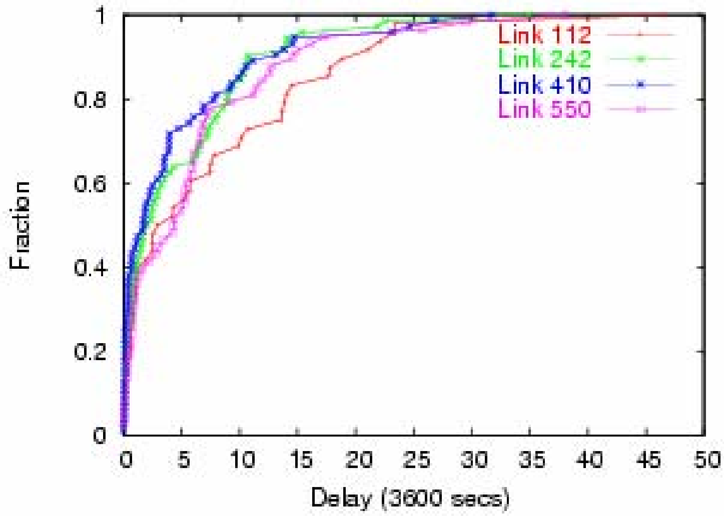
Layered Approach:

- ✓ OS provides network and event-handling services
- ✓ Middleware adapts, updates apps, protocols dynamically
- ✓ New protocols can be plugged in at any time

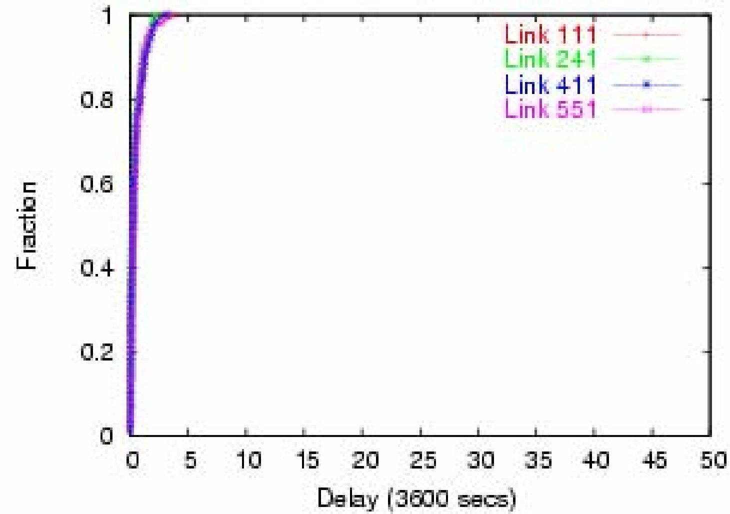
Forwarding Algorithms Comparison

Algorithm	Who	When	To whom
flood	All nodes	New contact	All new
Direct	Source only	Destination	Destination only
Simple replication(r)	Source only	New contact	*r* first contacts
History(r)	All nodes	New contact	*r* highest ranked
Erasure Coding (ec-r)	Source only	New contact	*kr (k>=1)*first contacts (k is related to coding algorithm)

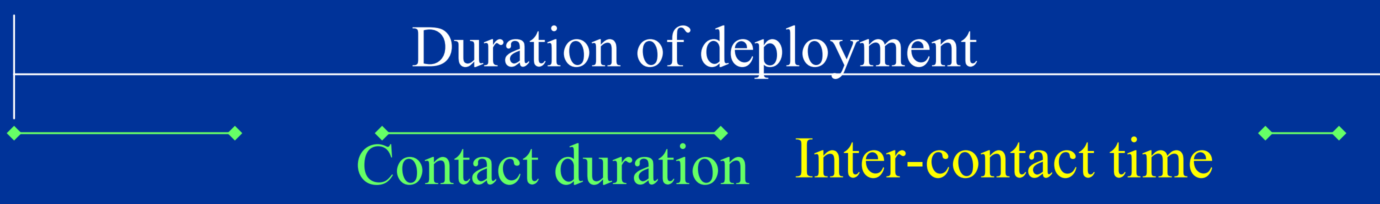
Analyzing node contact times from zebra mobility trace



(a) Inter-Contact time distribution



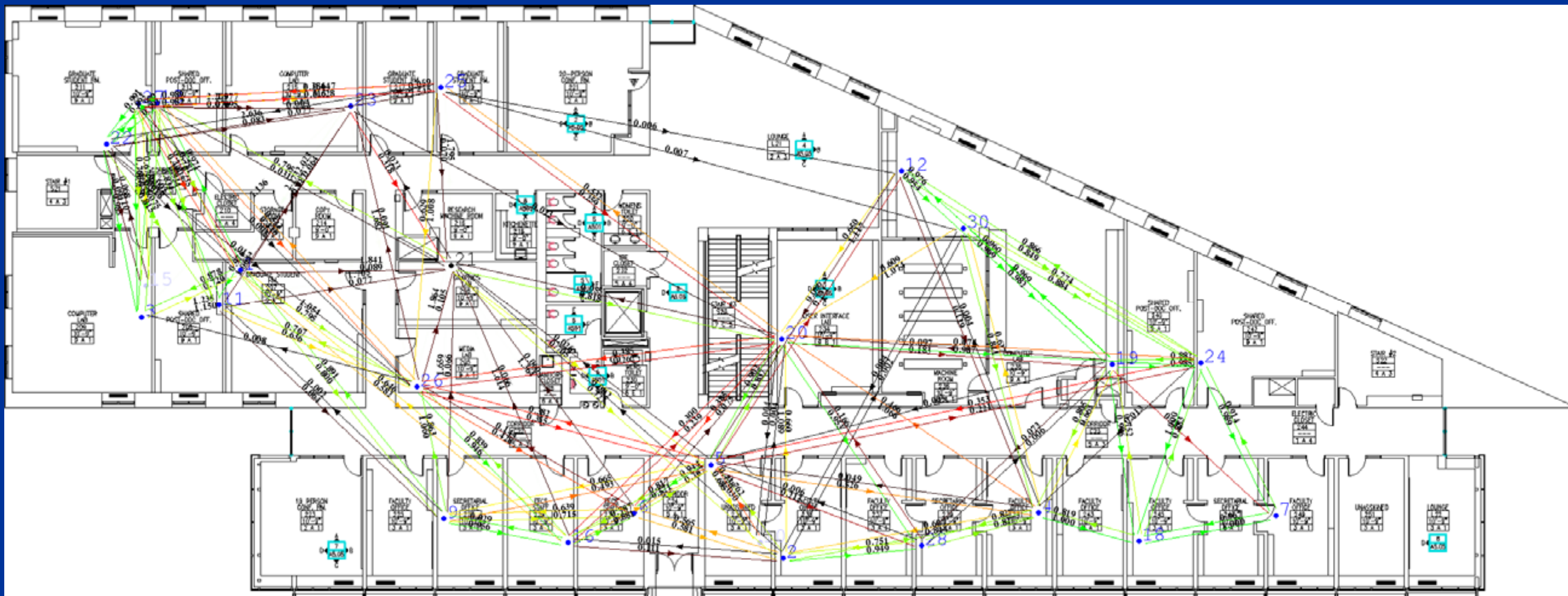
(b) Contact duration distribution



MetricMap: a machine learning approach to estimate link quality

- Discover correlations between local metrics (*feature vector*) and link quality indicator (*goal*)
  - Buffer size, channel clear access, depth, RSSI, etc.
  - Hardware-provided link quality indicator
    - Closely related to received signal strength indicator
    - Also reflects other factors
- Derive inference rules using standard machine learning algorithm
  - Weka package
  - C4.5 (decision tree) and JRip (decision rules)
  - training sets collected by running on a ***Berkeley-MicaZ*** based real-world sensor network testbed

Deployed on MoteLab and Mirage



With Y. Wang, L.S Peh, under preparation

Preliminary Results

- MetricMap improves on previous approaches:
  - 10-40% in terms of packet latency
  - ~80% in terms of packet success rate
  - Also improves fairness properties
- Benefits especially pronounced at high offered loads.
  - MetricMap is good at distinguishing packet drops due to link quality from packet drops due to congestion