Ubiquitous autonomic management

-or-

How I learned to stop trying to avoid the real world when building ubiquitous and sensor systems

Simon Dobson
School of Computer Science and Informatics
UCD Dublin IE
simon.dobson@ucd.ie http://www.simondobson.org

lero THE IRISH SOFTWARE ENGINEERING RESEARCH CENTRE

CLARITY clarity-centre.org

sfi science foundation ireland fondúireacht eolaíochta éireann

Overview

- Ubiquitous and sensor systems
  - The characteristics that make them the next big challenge
- Towards a more outward-facing network management
  - Uncertain reasoning
  - Component recomposition
  - Autonomic control and deriving control to improve data provenance

The space of opportunities

- Increasing emphasis on sensor-led systems
  - Micro: environmental sensing, e-health
  - Macro: scientific/enterprise/social decisions
- View diverse information as a unified whole
  - Reason, don't (just) program
  - Flexible and autonomic infrastructures
  - Context-aware, adaptive
  - Respond to challenges locally and globally
- A very different landscape for science and computing, that needs particular expertise

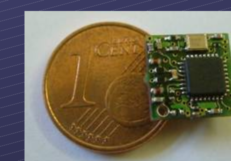
Coutaz, Crowley, Dobson and Garlan. Context is key. Comm. ACM **48**(3). 2005.

Thanks

- The MUCS organisers for letting me think about these ideas
- My students and colleagues at UCD, including but not limited to:
  - Eoin Bailey, Davide Cellai, Adrian Clear, Lorcan Coyle, Mike Hinchey, Joe Kiniry, Stephen Knox, Josu Martinez, Olga Murdoch, Paddy Nixon, Aaron Quigley, Graeme Stevenson, Juan Ye
  - Many of the ideas here are theirs, not mine

Background: ubiquitous sensing

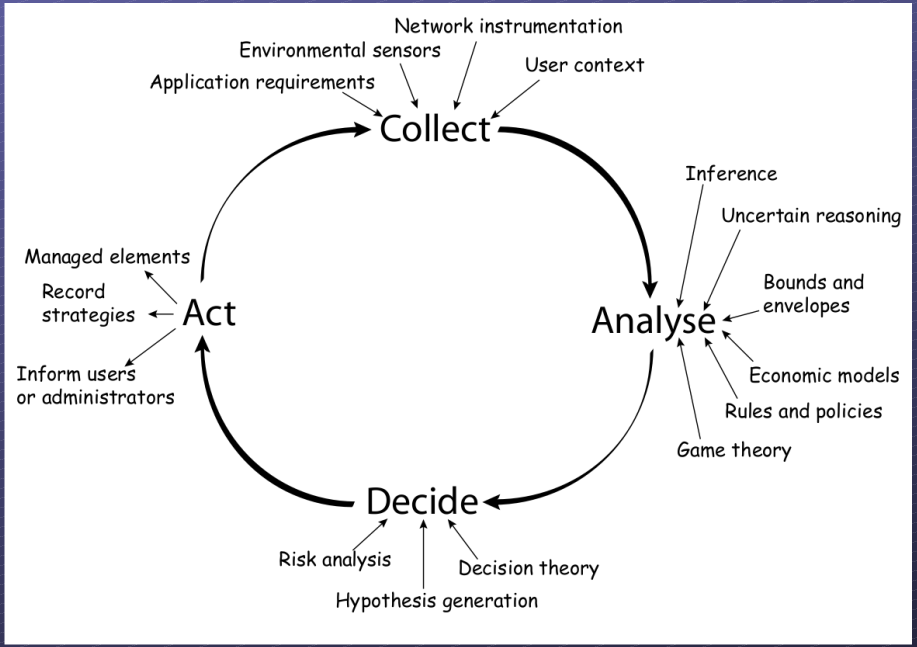
- Sensor networks and pervasive computing bridge real-world facts to in-computer models, to allow decision-making
  - Small, low-power nodes, context-aware, limited capabilities individually



- Fantastic opportunities in systems, and in how we store, process and interpret information
  - Diverse, uncertain, uncommon
  - Network seriously exposed to partial failure, traditional techniques often inadequate

Background: autonomic systems

- Adaptive control
  - Close the control loop
  - Respond to sensors, inference, predictions
  - Little (or no) human-in--the-loop control



From Dobson *et alia*. A survey of autonomic communications. ACM Trans. Auto. Adapt. Sys 1(2). 2006.

- Broad range of techniques
  - Respond to changes in environment, goals, physical models, …
  - Achieving stability, predictability, trust hard to guarantee in the face of uncertainty

Why this affects management

- By *management* we mean the way in which the service of a system is delivered
  - Quality, fault management, instrumentation, reporting
  - ...and now also sensing, adaptation, re-purposing, self-healing, ...
- Ubiquity implies that the network "protrudes" into the real world – and conversely that the real world protrudes into the network
  - Reality intrudes in ways we've tried hard to avoid

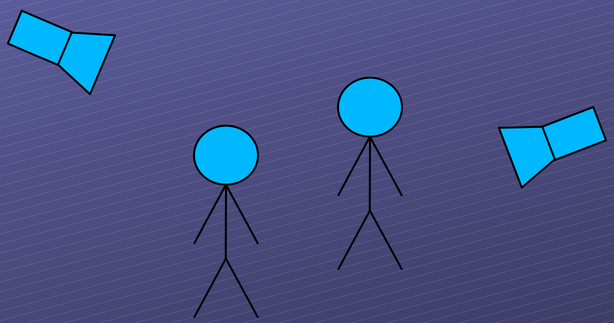
How this affects management

- Adaptive management means that both the network *and* the management system evolve
  - Can't (usually) pre-load all the possibilities
  - “Open-adaptive” behaviour
  - Enormously increases space of possible system behaviours – in good *and* bad ways
- If we accept the entwining of the world and the network, maybe understanding the *world* better will let us understand the *system* better

Key research drivers

- Uncertainty
  - Can’t always be engineered away at source
  - ⇒ Must be *reasoned* away
- Stable adaptive systems
  - System must adapt but guarantee properties
  - ⇒ *Adaptive spaces* as a whole-system model
- Systems engineering
  - Must ensure that systems *are* programmable
  - ⇒ *Theory and practice* meet on an equal footing

Uncertainty



- Uncertainty and inaccuracy are the defining features of inputs
  - Take rapidly-changing *context* data and generate semantically meaningful *situations*

Sensors may see some, all or no people; agree or disagree on their identities; repeat observations; report with different footprints and frequencies

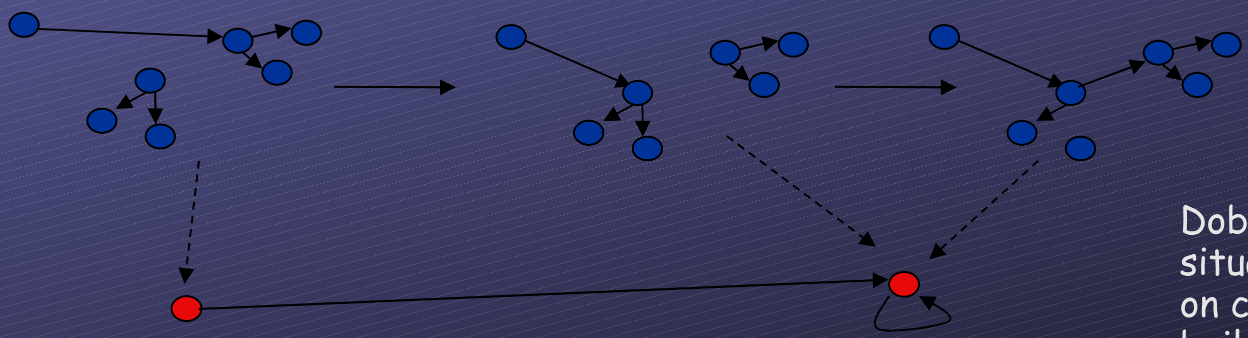
- Noise makes exact determination problematic
  - Maintain a dynamic view of *possible* and *most probable* situations
  - Refine as observations come in
  - Leverage the structure of behaviour

Dobson and Nixon. More principled design of pervasive computing systems. LNCS 3425. 2004.

Map context to situation

- Context (as RDF) fibres over situations
  - Each context identifies a situation, which in turn selects some appropriate behaviour

Context graph defines current and past observations of the world



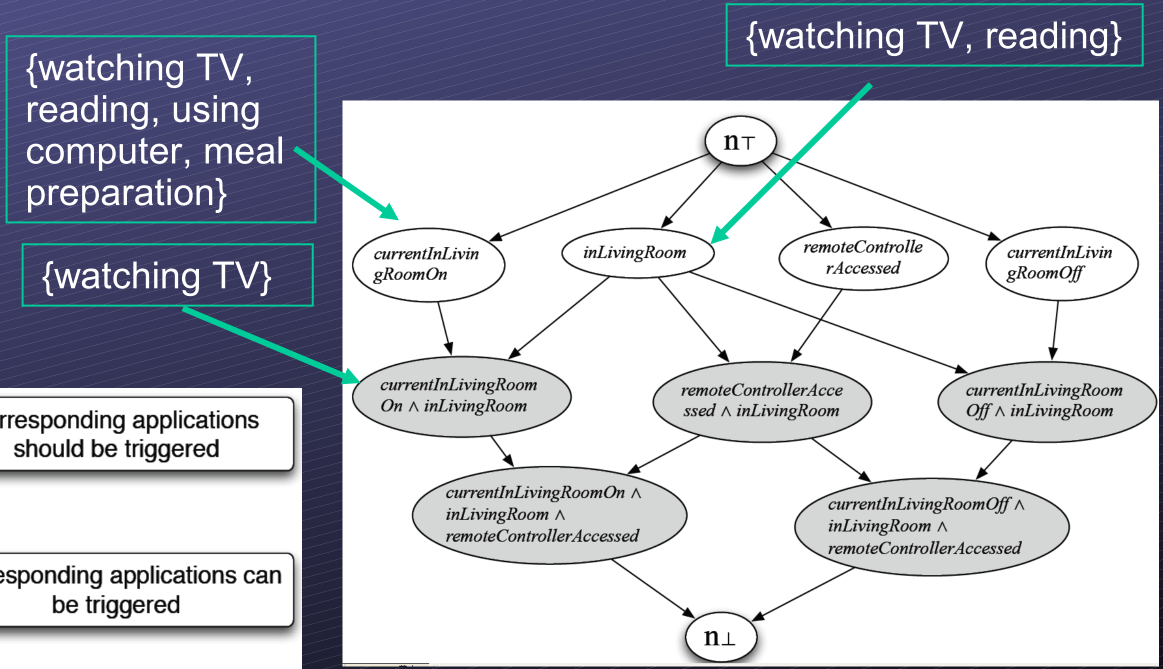
Dobson and Ye. Using fibrations for situation identification. Proc. Workshop on combining theory and system-building at Pervasive'06.

Situation transitions provide a workflow for how the user's situation is expected to evolve

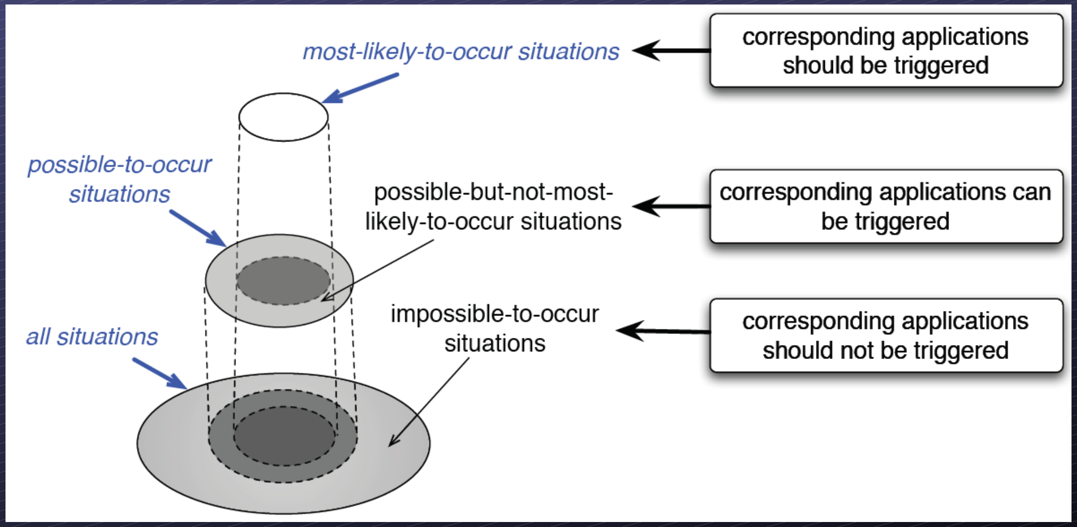
  - Simplifies management, reasoning
  - Doesn't handle under-identification of situations

Capture under-identification

- Lattice structure represents mapping from context to *sets* of situations
  - Validated against PlaceLab data set
  - Use structure to aid inference



Ye, Coyle, Dobson and Nixon. Using Situation lattices in sensor analysis. Proc. Percom'09.



Component recomposition

- Use a model of functionality to drive (re)composition of web service components
  - Interface specs alongside “normal” signatures
- If a component fails, apply tactics to generate a new composition that'll work
  - If the database falls over, substitute a log file and a replayer that'll replay the transactions once the database is back
  - Prove that the tactic meets (fully or completely) the functionalities it replaces

Martinez and Dobson. Functionality recomposition for sellf-healing. Proc. ICSDT. To appear

From reasoning to networks

- Very mission-driven
  - Must manage provenance of collected data
- Mission trade-offs can’t be made *a priori*
  - Fixed sensing and comms periods (*duty cycle*) makes for predictable battery usage
  - Too long a sensing period risks missing phenomena
  - ...too short burns power sensing the uninteresting
  - Too long a communications period risks losing data through failures, either local or remote
  - ...too short runs down everyone's batteries

Adaptive sensing

- We therefore want to *entangle* the management of a node with its sensing functions
  - Make duty cycle etc a function of what’s being sensed
  - Increase frequencies when there's “something interesting going on”; reduce them otherwise
- Makes things *much* more interesting
  - Hard to model power lifetime etc
  - Additional, uncertain factors to consider in terms of system’s adaptive (*process*) correctness

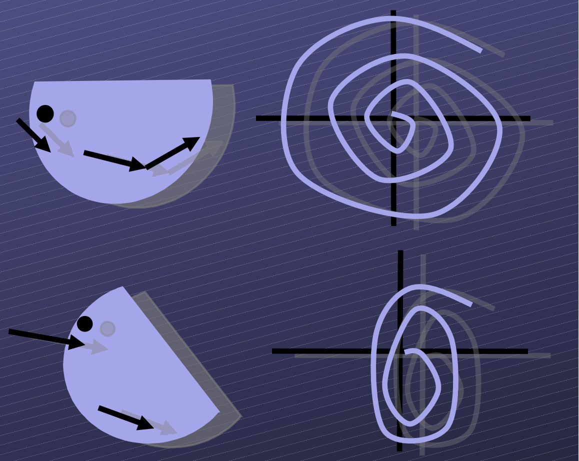
The uncertainty principle

- We don't want sensing to alter what we're sensing
  - The Heisenberg uncertainty principle applied to sensing, perhaps?
- This places limits on many things
  - The size and intrusiveness of sensors – must be small enough not to interfere
  - Their number – can't flood an area to the detriment of other uses


STOCKS POSTED GAINS TODAY ON NEWS THE "HEISENBERG UNCERTAINTY PRINCIPLE" WAS FOUND TO BE A CERTAINTY...
SCHWADRON

A framework for adaptive behaviour

- Capture the space of possible behaviours
  - Power consumption, bandwidth, frame rate, resolution, jitter, …
  - Define a dynamics moving between valid states
- Model evolution through changing adaptive space and/or dynamics
  - Whole-system descriptions amenable to analysis
  - Extending mainstream software correctness



Concept mission: marine sensing

- Network of static sensors
  - Position in “interesting” places (or at random)
  - In reality, constrained to stay away from fisheries, scenic spots, ...
- Mobile sensors
  - Move around, purposefully (or at random)
  - Detect and respond to “interesting” events
  - Provide “good” data

What constitutes “interesting”?

When is data “good”? How can we guarantee that it matches the phenomenon we’re tasked to sense?

Options

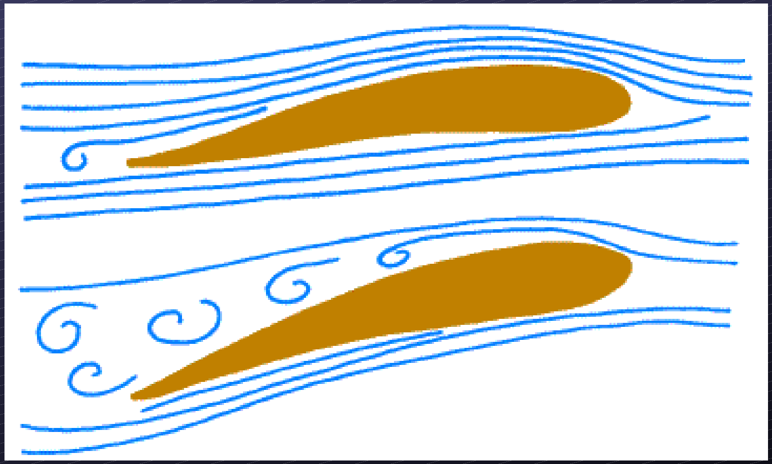
- Network of static sensors
  - Position in “interesting” places (or at random)
  - In reality, constrained to stay out of the shipping lanes, scenic areas, fisheries, ...
- Mobile sensors
  - Move around, purposefully or at random
  - *Try* to stay out of everyone's way, or be small enough to be run down without a problem
  - Much harder control problem

Challenges

- Too many to mention...
  1. How can we move sensors under computer control so it goes where we want it to go?
  2. How to we *decide* where we want to go?
  3. How do we *express* this goal in a way we can analyse?
  4. What is the best programming approach and/or language for highly sensorised adaptive systems?
- For this talk we'll focus on the second and third

Where to go?

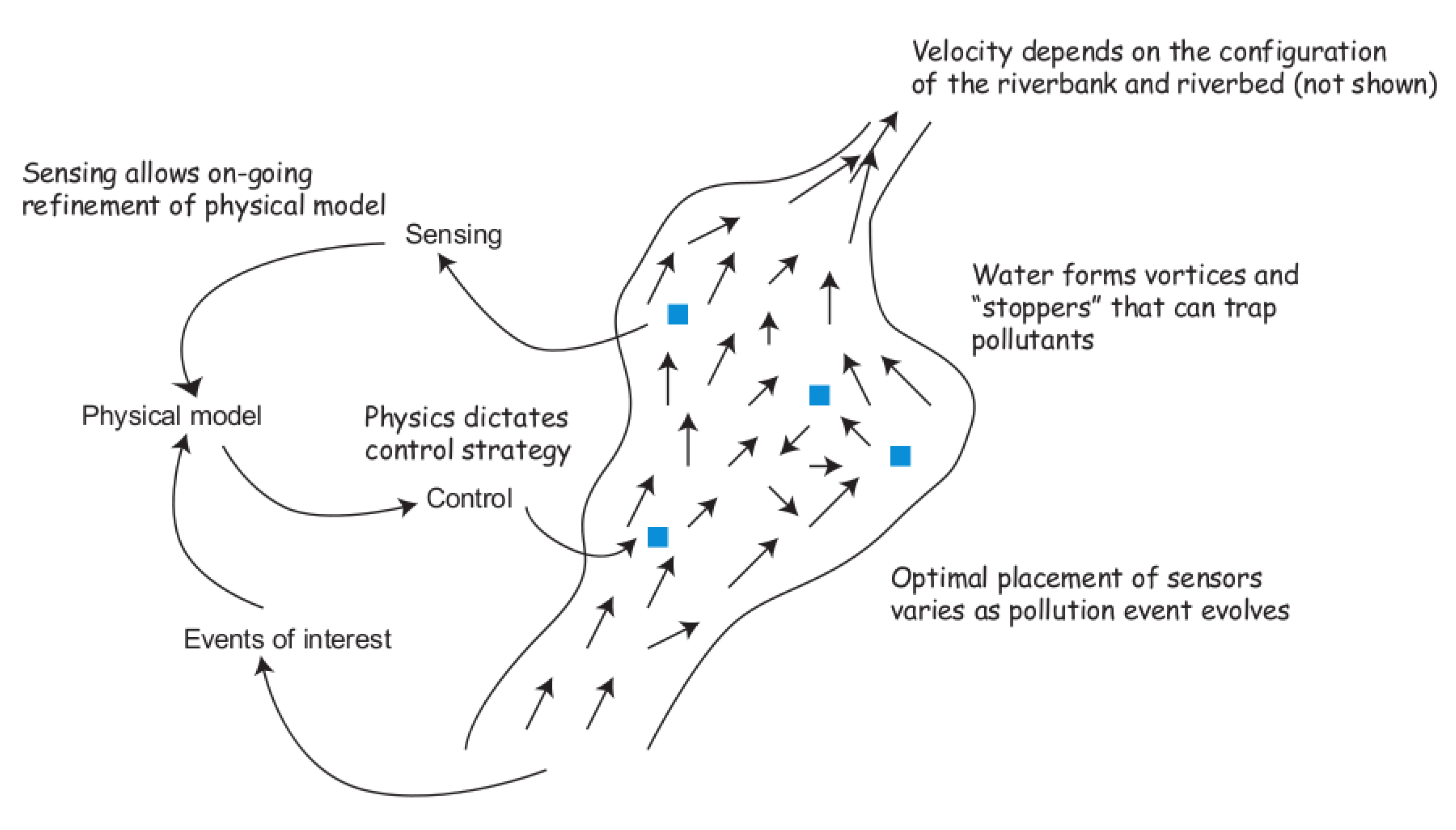
- Where would we want to move to?
  - Random direction – might find something interesting
  - Static search pattern – can be tailored
  - Dynamic pattern – need to know how to plan the pattern
- Analogy: if you randomly sample an airflow over a wing, you'll get mostly laminar flow



Knowing the physics

- In order properly to plan a search pattern, we need to understand the physics of what we're searching for
  - What constitutes an “interesting” place?
  - How do these places evolve?
- Although the detailed understanding of water flows is extremely complex, a naïve understanding will (to some degree) suffice for our purposes

A naïve understanding



Dobson, Coyle, O'Hare and Hinchey. From physical models to well-founded control. Proc. IEEE EASe. 2009.

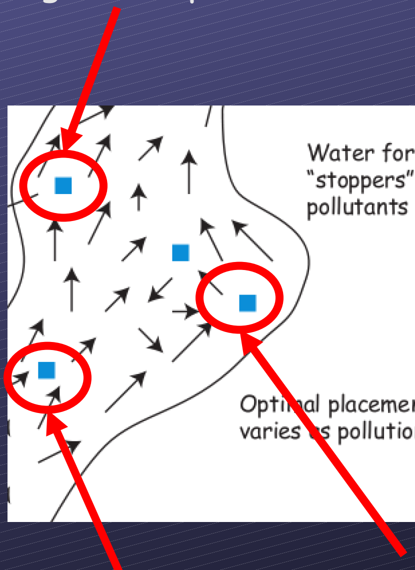
Controlling the swarm – 1

- Define a value function over space
  - Wind (vector)
  - Flow field (vector), pollutant level (scalar)
  - Location of nodes (GPS, inertial tracking)
- Balance issues
  - Maximise coverage of interesting things, but not at the expense of global coverage
  - Don't yet have a really good definition
  - All seem to need local *and* global information
  - Implementation-neutral

This essentially encodes the mission goals

Controlling the swarm – 2

- Tactics
  - Change the constellation of sensor nodes so as to improve the value function of the system
  - Piecewise dynamics
  - Need to maintain “inertia” of individual nodes’ behaviours
  - Can we define envelopes of stability for the system?



Outward-facing management

- The point here is that it's the real world that defines how the system behaves
  - A physical, scientific model, used to evaluate tactics
- This *dynamic evaluation* is really important
  - Not a policy set decided *a priori*
  - Dynamic change and re-purposing
  - The management functions are *driven by a model* of the environment, maintained on an on-going basis

What this gives us

- In environmental sensing, one always has the question of whether the data collected really matches the world it purports to model
- Model-driven management gives confidence that this is the case
  - The network collected according to the physics
  - ...so express the goals in scientific terms
  - ... and we *know* that (to some degree) we follow them

Three things to take away

- Ubiquitous systems must face outwards, and embrace the world in which they're embedded
- Model-driven managemant – bringing an understanding of the world into the management system – gives leverage
- The science can be used to improve practice