Some future systems will be too complex to design and implement explicitly. Instead, we will have to learn to engineer the desired behaviours implicitly. We will do this through the discovery and programming of simple rules of behaviour, applied to a mass of dynamically configured and interacting components, from which desired complex behaviours emerge. Examples? Mechanisms design (game theory, micro-economics) - Rational actors have local, private information - Emergent: optimal allocation of scarce resources - Optimal decisions rely on truth revelation

Examples? - Swarming behaviour (flocks, wasp colony behavior) - Autonomous (non-rational) actors, local interactions only - Emergent: "swarm" behavior - UAV swarms and autonomous robots
Engineering Emergence

- Social communication (gossip, epidemic algorithms)
  - Large, ad hoc, dynamic networks
  - Emergent: minimum power to achieve eventual consistency
  - Low power, low reliability sensors and data propagation

Emergent behaviours: flocking, squabbling, migration waves, panic scattering, orbiting points of attraction (if only a small group), feeding frenzy (if a large enough flock), turbulence, maze solving, ...

Lightweight Communicating Processes

- Fine-grained
- Massively parallel (zillions)
- Process-oriented

- Processes, networks, networks-within-networks
  - Channel (reader-writer) synchronisation
  - Barrier (multiway synchronisation)

- Ever-changing network topologies
  - Dynamic birth, re-connections, death
  - Mobile channels and processes
  - Mobile process location and neighbour awareness

Case study:

- Boids: avoid collisions, match vector with those of birds in sight, take fright if a hool is spotted, be attracted by flock...

Engineering Emergence: an occam-π Adventure

A thesis, boids and a demo ...

- Process architecture and boids ...

- Observations of emergence ...
- Emergence engineering ...
- Summary and Conclusions ...

This is the way of the world ...

Processes, networks, networks-within-networks

- Channel (reader-writer) synchronisation
- Barrier (multiway synchronisation)

- Ever-changing network topologies
  - Dynamic birth, re-connections, death
  - Mobile channels and processes
  - Mobile process location and neighbour awareness

foo  bar  merge

server

(c) three processes sharing the client end of a channel bundle to a bank of servers sharing the other end.

(d) n processes enrolled on a shared barrier (any process synchronising must wait for all to synchronise).
The Matrix

Mobile Agents

Location (Neighbourhood) Awareness

Location (Neighbourhood) Awareness

Location (Neighbourhood) Awareness

Location (Neighbourhood) Awareness

Location (Neighbourhood) Awareness

Location (Neighbourhood) Awareness

Location (Neighbourhood) Awareness

occam-π Boids Model

- Each server is responsible for its own region of space …

- A region may hold many birds … or none …

- Each bird is in only one region at a time … but can consult with its immediately neighbouring regions …

So, not this …
Each bird registers its state (position, vector, alarm state, colour, etc.) to the server for its region.

Each bird knows its position relative to its current region of space – it doesn’t know which region that is.

Birds have a maximum range of vision (up to a radius of 1) – so may need to consult up to 4 servers.

Birds have a maximum range of vision (up to a radius of 1) – and also have a restricted angle of vision – in this case to 300° (i.e. missing 60° rear view).
A bird process follows a general pattern for mobile agents …

- It has a pilot sub-process, responsible for dealing with the servers in its immediate neighbourhood and, when necessary, moving between them. The pilot is the eyes and wings of the bird …

- It has brain sub-processes, receiving vision information from the pilot and computing wing muscle forces back to the pilot …

The birds are kept in step with each other (and with a visual renderer process) by barrier syncs … which also provides a model of time. The pilot process does this …

The occam-π BARRIER type corresponds to a multiway CSP event, though some higher level design patterns (such as resignation) have been built in.

Basic CSP semantics apply. When a process synchronises on a barrier, it blocks until all other processes enrolled on the barrier have also synchronised. Once the barrier has completed (i.e. all enrolled processes have synchronised), all blocked processes are rescheduled for execution.
Computing Laboratory

**occam-π Boids Model**

- A bird process follows a general pattern for mobile agents ...
- The birds are kept in step with each other (and with a visual renderer) process by *barrier sync* ... which also provides a model of time. The *pilot* process does this ...
- The birds are kept in step with each other (and with a visual renderer process) by *barrier sync* ... which also provides a model of time. The *pilot* process does this ...
- A regional server process holds a dynamic array of all visiting birds ...
- It supplies this information to all observers: the birds, the process doing the rendering ... and, in future, live hawks, food, etc.
- These server processes do not *sync* on the barrier ... they have no need keep note of time ... or keep in step with the birds.

**Engineering Emergence: an occam-π Adventure**

A thesis, *boids* and a demo ...
Process architecture and *boids* ...
Observations of emergence ...
Emergence engineering ...
Summary and Conclusions ...

**Observing Emergence**

**Case study - reminder**

*Birds*: avoid collisions, match vector with those of birds in sight, head for the centre of mass of birds in sight, take fright if a hawk is spotted, be attracted by food ... *Emergent behaviours*: flocking, squabbling, migration waves, panic scattering, orbiting points of attraction (if only a small group), feeding frenzy (if a large enough flock), turbulence, maze solving, ...

**occam-π Boids Model**
**Computing Laboratory**

**Title goes here**

---

**Observing Emergence**

**Case study – reminder**

There is nothing in the design or programming dealing with flocking, scattering, orbiting, feeding frenzies, migration waves, turbulent flow or solving mazes.

Emergent behaviours: flocking, squabbling, migration waves, panic scattering, orbiting points of attraction (if only a small group), feeding frenzy (if a large enough flock), turbulence, maze solving, ...

Nevertheless, the occam-pi kernel (CCSP) does a good job of very lightweight load balancing across all the cores (that we have right now) ...

**Engineering Emergence: an occam-π Adventure**

A thesis, boids and a demo ...  
Process architecture and boids ...  
Observations of emergence ...  
Emergence engineering ...  
Summary and Conclusions ...

**Scientific Instruments**

Scientist / Engineer  
Instrument  
Complex System

---

**To Boldly Go – Emergence Engineering**

It’s programming, Jim, but not as we know it ...

---

**Scientist / Engineer**  
Scientific Instruments
**Simulation as a Scientific Instrument**

*Simulation as an experimental design process for emergent systems*, Andrews-Stepney-Winfield

Research and discovery of low-level processes from which observed complex behaviours emerge.

**Discovering Unexpected Relations between Phenomena**

Research and discovery of low-level processes from which observed complex behaviours emerge.

**Discovering Unexpected Relations between Phenomena**

Research and discovery of low-level processes from which observed complex behaviours emerge.

**Discovering Unexpected Relations between Phenomena**

Research and discovery of low-level processes from which observed complex behaviours emerge.
Discovering Unexpected Relations between Phenomena

Research and discovery of low-level processes from which observed complex behaviours emerge.

Computer modelling and simulation can show unexpected relationships between apparently different complex phenomena, operating with different physics and at different scales.

Discovering Unexpected Relations between Phenomena

Computer modelling and simulation can show unexpected relationships between apparently different complex phenomena, operating with different physics and at different scales.

Discovering Unexpected Relations between Phenomena

Computer modelling and simulation can show unexpected relationships between apparently different complex phenomena, operating with different physics and at different scales.

Computer modelling and simulation can show unexpected relationships between apparently different complex phenomena, operating with different physics and at different scales.
Discovering Unexpected Relations between Phenomena

Computer modelling and simulation can show unexpected relationships between apparently different complex phenomena, operating with different physics and at different scales ... because their (differing) behaviours emerge from agents following identical low-level rules, just with slightly different key parameters ...

Through computer modelling and simulation, we can investigate the emergent properties of whole new worlds of materials, new states of physics, by experimenting with varieties of agent programmed with simple low-level rules. 

Motivated to find ways to build those agents for real! Some of these may turn out to be interesting and useful so that we might be ...
Engineering Emergence: an occam-π Adventure

A thesis, boids and a demo …
Process architecture and boids …
Observations of emergence …
Emergence engineering …
Summary and Conclusions …

To Boldly Go – Summary

We have described an architecture for the intentional emergence of complex systems behaviour.

Processes (mobile, communicating and lightweight) are good candidates for supporting such an architecture. occam-π provides this computational model and scales well across both shared and distributed memory.

Engineering the desired behaviour is indirect. We need to discover simple low-level rules for pieces that we can program and, then, run masses of them. For complex systems, there will be no high-level components that directly work the behaviour we want.

Future?

Drug design: try to build molecules with certain shapes (to match the geometry of suspected weak spots of rogue cells) …

Emergent behaviours: elimination (or inhibition) of tumours.

Autonomous driving: avoid collisions, head for the longest straight clear path (with speed in proportion), add bias in general favour of destination (if known) …

Emergent behaviours: safe driving, efficient use of the road, faster completion of journey.

To Boldly Go – Summary

Once more, and this time with feeling …

Any Questions?

Research projects:

- cosmos-research.org
- occam-pi.org
- concurrency.cc
- rmox.net
- moodle.kent.ac.uk/external/course/view.php?id=31

Any Questions?