



# MACC Using Stigmergy

---

Paul Valckenaers  
K.U.Leuven – PMA  
Belgium



# Overview

---

- Emergence in the real world
  - PROSA
- Suitability for integration, evolution...
  - Stable constraints, ironware
- Concerns in control systems
  - Stigmergy and ants
  - Emergent load forecasting
  - Process plans or recipes
- Other issues and conclusions



# Real World Emergence

---

*Talk is about making desirable overall behaviour emerge, without taking every possible situation into account!*

- How did life emerge?
  - Pool of basic material + energy + chance
  - Autocatalytic sets
  - Consequences: being first, lock-in...
  - Other mechanisms & insights: hierarchical structure...
- Complex artefacts
  - Symbiosis (domestic animals, fruit, vegetables...)
  - Artefact (i.e. software) cost structure
  - Autocatalytic set: critical mass versus artefact complexity



# PROSA & Critical Mass

---

- **P**roduct agents
- **R**esource agents
- **O**rders agents
- **S**taff agents
- **A**rchitecture
  
- Basic: P – R – O
  
- Basic agents:
  - Reflect what is/exists
  - Responsible
- Staff agents:
  - Know-how, legacy...
  - Support, advise
- OO:
  - Specialisation
  - Aggregation



# Integration, evolution...

---

- Lock-in is a problem:
  - Spanish & French railways, driving on left or right
  - Many useful things fail to materialise, much is wasted, crashes...
  - Hard to show how much it costs, but
  - It affects critical mass and slows down development speed: exponential effect
- How to?
  - Avoid/minimise future integration problems
  - Design an ability to evolve, adapt...



# integration, evolution...

---

- Infinite state space of the universe: **U**
- A problem is a constraint on **U**, which defines a subset of **U**: **P**
- A solution keeps the state of universe within the subset **P**
- Any solution  $X$  corresponds to a constraint on **U**, which defines a subset of **P**: **S<sub>x</sub>**

*Consider only solvable problems*



# Integration, evolution...

---

- Solution  $X$  corresponds to a subset of  $\mathbf{P}$
- Solution  $X$  is embedded in the state of the universe:
  - $\mathbf{S}_x$  only contains one state for every time coordinate in the past and present
  - $\mathbf{S}_x$  can contain multiple states with the same time coordinate if they are (far enough) in the future  
*! Autonomous systems !*
- We cannot preserve  $\mathbf{P}$



# Integration, evolution...

---

- Subsolutions need to be integrated, need to accommodate uncertainty and changes...
- $\mathbf{S}_{xp}$  needs to be reconciled with  $\mathbf{S}_{vq}$ ...
- $\mathbf{P}$  has no conflict with  $\mathbf{Q}$
- Ideal solutions avoid the introduction of constraints that can cause future conflicts
- Or at least make them easy to undo by avoiding the build-up of inertia



# Integration, evolution...

---

- Known mechanisms:
  - Low and late commitment
  - Low inertia implementation: CAD, CAE...
- Design principles:
  - Stable constraints (= no new constraints)
  - Earlier design choices are **not** stable constraints, only constraints

*Fall-back solutions ( + Herbert Simon)*



# Integration, evolution...

---

- Stable constraints
  - Scope of the solution
  - Physical laws
  - Mathematical laws
  - Invariant/hard technological limitations
  - Invariant application domain properties
- Example: motion control



# Stable Constraints

---

- Coordination and control of ironware
  - Physics, mathematics, technology
  - Speed difference: computers - ironware
  - No competition for ITC resources
  - Cost-benefit leverage
  - Communication bandwidth and processing power
  - Response times and transmission delays



# Stable Constraints

---

- Reality is always consistent
  - Ironware is part of the reality for MACC
  - Essential models in OO-design
  - Reflecting reality - full life cycle
- Xmas lists often are inconsistent
- Xmas lists change frequently
- Functional decomposition is fragile



# Stable Constraints

---

- **P**roduct agents
- **R**esource agents
- **O**rder agents
- **S**taff agents
- **A**rchitecture
  
- Basic: P – R – O
  
- Basic agents:
  - Reflect what is/exists
  - Responsible
- Staff agents:
  - Know-how, legacy...
  - Support, advise
- OO: (Herbert Simon)
  - Specialisation
  - Aggregation



# Concerns in Control Systems

---

- Feasibility
  - Recipes, deadlock...
  - Thrashing, starvation...
- Operations
  - Load balancing, lead times, batching...
- Staff
  - Initial solutions, guidance...



# Concerns in Control Systems

---

- Providing information (reflection)
  - Feasibility, operational, staff
- Decision taking (fragile)
  - Uses (and respects) the information
  - Adaptation without software maintenance cascading to the information providing parts
- How to design *stable agents*?
  - Challenge: global coordination (= system-wide)
  - Stigmergy: ants provide the inspiration



# Stigmergy and Ants

---

- Stigmergy
  - Indirect interaction
  - Signs in the environment
  - Lightweight compared to direct negotiation
- Ants foraging for food
  - Simple rules, emergent coordinated behaviour
  - Global information is locally available
  - Environment is part of the solution



# Emergent Load Forecasting

---

- Resource agents reflect resources
  - Environment is part of the solution
  - Full life cycle, topology, ...
  - Locations have attached information spaces
  - What-if mode (control of ironware)
- Ex. Conveyor belt agent
  - Attributes, observers...
  - Modifiers/actions (autonomous)
  - What-if services



# Emergent Load Forecasting

---

- Attributes/observers
  - Information spaces attached to ...
- Modifiers/actions
  - Life cycle
    - Create, delete, connect, disconnect...
  - Usage (includes notifications)
    - Start, stop...
  - Synchronisation with reality



# Emergent Load Forecasting

---

- What-if services
  - PropagateUpstream
  - PropagateDownstream
    - Adapt time information
    - Register intentions
  - GiveLoadForecast
    - Based on intentions communicated by ...



# Emergent Load Forecasting

---

- Order agents create mobile agents that virtually move across the resources
  - Mobile agents behave as if the present time is their *estimated arrival time*
  - Mobile agents have the resource agents forecast their travel times
  - Mobile agents use the content of the info spaces
  - Mobile agents reflect decision mechanism of the order agent (without making assumptions about it)



# Emergent Load Forecasting

---

- Mobile agent inform resource agents about the order agent's intentions
- Resource agents combine intentions into a load forecast

*When visiting a processing unit, the product agent and resource agent predict processing time and ...*

*Mobile agents may send back an estimated arrival time for the next decoupling point (provided uncertainty remains low)*

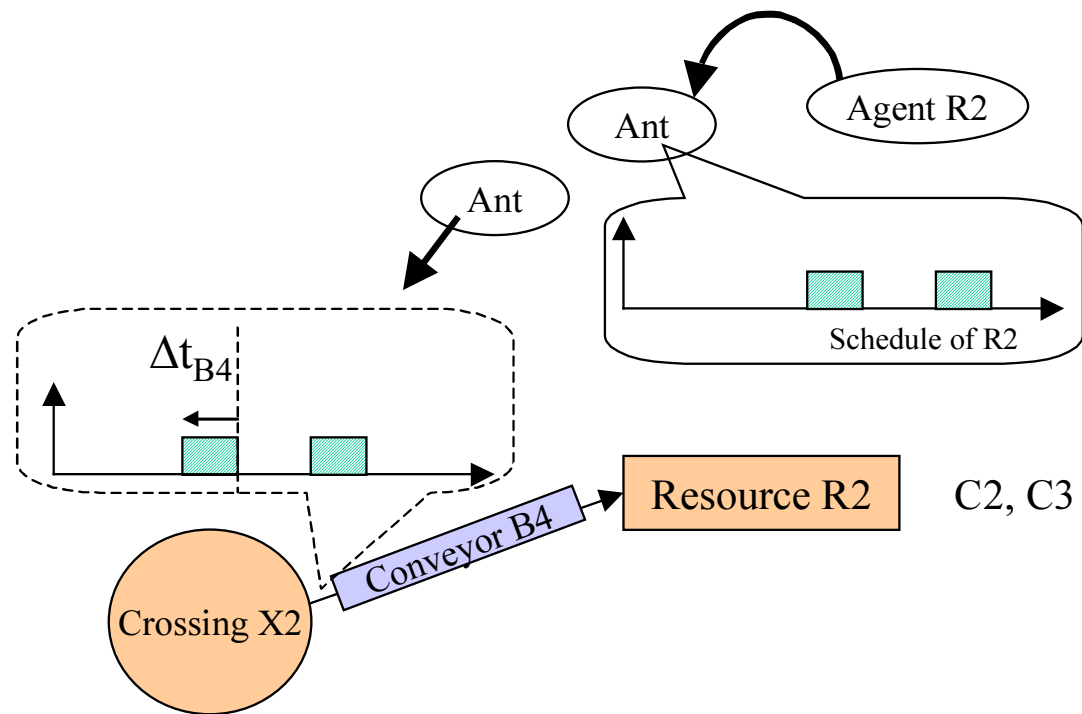


# Emergent Load Forecasting

---

- Critical resource agents (processing units) propagate load forecasts upstream
  - This information is copied into the information spaces attached to the resource agents (typically attached to exits, entries...)
- Upstream propagation ***adapts*** the forecasts to account for transportation time, ...
- Evaporation & refresh

# Emergent Load Forecasting





# Emergent Load Forecasting

---

## *Damping & nervousness*

- Order agents and their mobile agents have a tendency to stick to declared intentions (otherwise, forecasts will be invalid)
  - Perceived improvement thresholds
  - Frequency containment
  - Proximity in time to commitment



# Process Plans or Recipes

---

- Factory exit agents create mobile agents that propagate themselves upstream
- The mobile agents collect resource capabilities on their journey through the system:
  - The possible sequences of processing steps
- The mobile agents drop this information in the relevant information spaces attached to resources



# Process Plans or Recipes

---

- When resources have multiple entries, the mobile agents clone themselves and continue upstream along all paths
- When mobile agents discover that they or one of its clones has already visited a resource:
  - New loop detected
    - > adapt information and propagate
  - No new loop > terminate



# Process Plans or Recipes

---

- Order agents (and their mobile agents) read the information about the availability of processing steps along routes among which they have to select one
- Only feasible routes can be selected
- Refresh and evaporation
  - *Sponge* is only an optimisation



# Other Issues

---

- Concerns
  - Batch building and *unbuilding*
  - Pushing out-of-the-way (and avoid being-in-the-way)
  - Sequencing
  - ...
- Information types
  - Probabilistic/possibilistic intentions
  - Info adaptation for propagation purposes
    - Cycles, buffers, testing, propagation length...



# Concluding Remarks

---

- Emergence
  - Important properties/mechanisms
  - Design principles, stable constraints
  - MACC of ironware
- System engineering properties
  - Limited exposure, emergence...
  - Global information locally available
  - Evolving systems
- Actual control decisions come last