Engineering Self-Organization and Emergence: issues and directions

Franco Zambonelli
franco.zambonelli@unimore.it

Agents and Pervasive Computing Group
Università di Modena e Reggio Emilia

Outline

- Why Engineering Emergence and Self-Organization?
  - The Micro Scale
  - The Macro Scale
  - The Meso Scale
- Engineering Emergence as a Form of Control
  - Why is it Important?
  - Control by Perturbation
  - Control by Injection
- Engineering the Environment
  - Stigmergy and Emergence
  - Our Current Approach: Co-Fields and TOTA
  - Our Future Approach: the CASCADAS Project
- Conclusions and Open Issues
**The “Micro” Perspective**

- Traditional Mainstream (Software) Engineering adopts a "Micro" approach
  - Focus on individual components and their interactions
  - Full predictability at each level
  - Controlled non-determinism
  - Formal methods
- And this is here to stay for a multiplicity of applications
  - B2B, workflow systems, safety-critical systems
- Though getting more and more “autonomic”
  - Delegation of responsibility, automated negotiation, environmental dynamics, internal control loops, etc.
  - But this is far from emergence and self-organization...

**The “Macro” Perspective**

- Dealing with large-scale distributed systems
  - Dynamic P2P networks, Wireless sensor networks, multiagent systems ecologies, self-assembly
- Global and emergent behavior
  - As a way to obtain functionalities and self-* features
- Focus on “macro” aspects
  - No control over single components
  - Non-determinism
  - Global goal achieved via self-organization \(\rightarrow\) from local to global
- This is where current research (and most of us) are
Engineering in the Macro

- Two key issues
- Reverse engineering of natural systems
  - Study a natural phenomena and understand it
  - Can it apply to something?
  - Reproduce it in software
  - Several patterns available but...
  - Still “solutions in search of problems”
- Direct Engineering
  - Starting from a problem
  - There exists a set of local rules and local interactions that solves the problem at the macro-level?
  - Enforcing self-* features such as self-organization, self-adaptation, self-healing
  - Can we define methodologies and tools to do that?

- But there is more to self-organization and emergence...

But “Micro” and “Macro” are not Independent worlds...

- In most of real-world situations
  - Micro systems are developed
  - And situated in an operational environment on which we have no full control
  - And which cannot be “stopped”
- In other words:
  - Micro-scale system are immersed into macro- scale one
- And this is indeed always the case for
  - Networking
  - Service-oriented computing
  - P2P Computing
  - Pervasive Computing
The “Meso” Scale

- The micro and the macro scales co-exist and influence each other
  - How the local behavior affects the global one
  - And vice versa
- We must take both into account in system design and management
  - How can we predict at both levels?
  - How can we enforce properties at both levels?

More General “Meso” Scale Scenarios

- We can also consider “macro” into “macro”
  - A “self-*” system which we know well
  - Interact with another one
  - E.g., Gnutella into the Internet
- Problems
  - How do we know the two systems preserve their own properties?
  - How can we re-tune them to ensure properties at both levels?
Examples (non computational)

○ Traffic Management
  • We know well how a roundabout (a sharp example of self-organizing system) works per se
  • But what about its impact in a complex network of streets?

○ Ecology
  • We may know a lot about a specific ecosystems and about specific species
  • But what about the introduction of a new species into an ecosystem?
  • The Internet

Examples (computational)

○ Cellular Automata Networks
  • Upon small perturbations on the lattice (e.g., re-wiring)
  • Global changes in the CA dynamics

○ Routers instability
  • Upon topology changes or relevant traffic changes
  • Some router may fail to sustain updates
  • With waterfall congestion effects at the global level

○ Computational markets
  • Upon the insertion of agents with differentiated strategies in markets
  • Emergence of war-prices, cyclic phenomena, inflationary processes
Engineering vs. Emergence

- If we work at the micro scale only
  - With traditional “mainstream” engineering techniques
  - With a “design” approach
  - We miss the global perspective

- If we work at the “macro” scale only
  - We can achieve global properties by emergence
  - But may miss local goals
  - And may miss control

But why this is important?

- It is not only the recognition of a basic need
  - Most real cases face such issues

- More important, knowing what happens when we have a new system into an existing one
  - Can form the basis for new forms of decentralized control
  - And a new approach to engineer emergent systems

- Have a complex self-org macro system and
  - Know what happens when acting on it
  - Knowing how to enforce a specific behavior on it

- Shifting from
  - Local Rules → Hopefully Useful Global Behavior (pure macro-scale approach) TO
  - Local Rules + Decentralized Control → Engineered Purposeful Emergence
Decentralized Control via External Perturbation

- Try to
  - Disturb the system where and how you can
  - Attempting to identify perturbation patterns that
  - Affect as needed the emergent system behavior
- Example with Cellular Automata
  - Introduce a moderate degree of stochastic behavior in cells
  - And have global (otherwise non emerging) patterns emerge

Decentralized Control via Components Injection

- Try to
  - Inject new components/subsystem
  - That interact with the systems so as to
  - Affect as needed the emergent system behavior
- Example with Cellular Automata
  - Inject a few percentage of cells with modified rules
  - And have peculiar needed patterns (not any ones) emerge
Engineering Emergence at the Meso Scale

- What is needed to advance knowledge in decentralized meso-scale control?
  - So as to produce a set of practical tools
  - Enabling the engineering and control of complex self-org systems
  - In a methodical and repeatable way?
- Conceptual advances in modeling
  - Discrete vs. Continuum Computing
  - Logics vs. Physics
  - Genotypes vs. Phenotypes
  - Design vs. Intention
  - Topology vs. Dynamics
- Or maybe the adoption of more usable abstractions other than those of local rules and local interactions?

The Role of the Environment

- Let us assume the system is (or can be abstracted as) immersed in an environment
  - The physical environment (or a pervasive network infrastructure)
  - A manageable representation of a network environment (e.g., a structured overlay)
- And that
  - Interactions occur via the environment (**stigmergy**)
  - The environment reifies in the form of specific properties of it (artifacts or distributed states) the actual state of the system
- Then
  - Observing the environment means observing the system
  - Controlling the environment (i.e., controlling its properties) implies controlling the systems
  - Conceptual shift from controlling the system to controlling the environment
From Engineering Systems to Engineering Environments

- Once an environment abstraction is properly enforced
  - We may know how the system structure/dynamics reflect in the environment
  - We may know how to inject properties in the environment or how to perturb the properties of the environment
- Then we can study how
  - Given a complex systems of interacting components
  - A specific global behavior of the systems can be affected/influenced by the environment
  - A specific behavior can be enforced by acting on the environment

Spatial Environments

- The environment abstraction must be simple and usable
  - Enable an easy understanding of its structure
  - Enable an easy modeling of its properties and dynamics
- Environments as metric spaces
  - To apply concepts of coordinates and distances
  - To apply standard dynamical systems modeling
- Examples
  - Mobile Ad-Hoc Networks and Geographical Routing
  - Self-Assembly and Modular Robots
  - Pervasive Computing and Logical Spaces
  - P2P Structured Overlays (e.g., CAN, Chord)
- Open Question: can other types of systems tolerate a suitable mapping in metric spaces? (e.g., complex social networks)
Cognitive Stigmergy

- Most phenomena and systems relying on spatial stigmergic interactions
  - E.g., ant colonies and hormones in self-assembly
  - Assumes that components/agents simply reacts to properties in the environment
- However
  - It is possible to make the properties of the environment more “semantic”
  - E.g., not simply pheronomes but more complex artifacts and data structures
- And have components/agents “reason” about what they perceive
  - So that decentralized forms of control can be enforced directly into the system
  - “Cognitive self-management” achieved through controlled self-organization

The TOTA Approach at UNIMORE

- Attempting to exploit Stigmergic + Spatial + Cognitive Self-Organization In pervasive computing scenarios
  - A simple mechanisms for field-based stigmergy
  - Supported by a simple and highly usable API
- Spatial self-organization
  - The components of the systems (robots, mobile users)
  - Live and execute in space (or in a network lattice)
- Stigmergic self-organization
  - All interactions occurs via computational fields that diffuse
  - And that are locally sensed by agents
- Cognitive self-organization
  - Agents can perceive properties associated with fields
  - They are not necessarily merely “reactive”
The Model: Co-Fields

- Agents (or the environment itself) spread fields across the environment
  - Field-specific (app. specific) propagation rule
  - Local sensing of fields
- Global behavior and self-organization
  - Perception by agents of "coordination fields" as combinations of individual fields
  - Action driven by local shape of coordination fields
  - Preserve the possibility of field-awareness

Example: flocking

The Infrastructure: TOTA

- "Tuples On The Air" relies on a distributed tuple-based coordination model
  - Distributed tuple structures propagated across a network environment (or parasitically in an RFID-enriched env)
  - Locally read by agents
  - Providing context-awareness and field-based structures
- Each tuple characterized by $T = (C, P, M)$
  - $C =$ content, associated with it, possibly changing during propagating
  - $P =$ propagation rule, how the tuple propagates and how it changes $C$ while propagating
  - $M =$ maintenance rule, how the tuple structure re-shape upon network changes
- Application agents can
  - Inject any application-specific tuple structure (field)
  - Read locally read available tuples
  - React, if they think so, and be affected by these tuples
A Simple Example

Injection of a simple tuple

\[ C = (\text{int } v = 0) \]

A Simple Example

Propagation of the tuple

\[ P = (\text{spread everywhere, inc } v \text{ at each hop}) \]
A Simple Example
other agents can locally sense the tuple
i.e., the local value deriving from propagation
A Simple Example
what if the network topology changes?

*e.g., due to mobility of source node*

A Simple Example

a maintenance rule can specify how to act

*e.g., $M = (\text{react to changes, restructure tuple})$*
Engineering Emergence in TOTA

- **Micro-scale**
  - Exploit fields as a sort of distributed shared memory for contextual interactions
  - Tolerating network and environmental dynamics

- **Macro-scale**
  - Exploit fields to re-produce known phenomena of self-organization
  - Or to invent new (we can invent our own laws for field propagation)

- **Meso-scale**
  - When a specific (micro) field-based application is immersed in a macro-scale scenario
  - Proper combination of fields can accommodate both
    - The needs of the micro-scale
    - The needs of the macro-scale
  - In any case, new fields can be injected for the goal of enforcing the needed controls

Engineering Emergence in TOTA: An Example (1)

- **Micro-scale**
  - Users visiting a museum
    - Where are you? → PRESENCE fields
  - Meeting by tourists
    - Tourists following each others’ PRESENCE fields
  - Flocking by museum guards
    - As in the flocking example
  - All of these realized by application-specific and independent fields
Engineering Emergence in TOTA: An Example (2)

- **Macro-scale**
  - Diffusive Load Balancing of Crowd
    - Global diffusion of Presence fields
    - Weighted with data expressing room capacity
  - Users behavior
    - Can follow suggestions
    - Can analyze what’s happening (cognition!)
  - Overall good load balancing even if a limited percentage of users follows the fields suggestions

- **Meso-scale**
  - Flocking + Load Balancing
    - How can we conciliate?
  - Approach:
    - Have flocking agent perceive (and react upon) the following field:
      \[
      \text{Coord} \_ \text{Field}(x, y, t) = \text{Flock} \_ \text{Field}(x, y, t) + \mu \cdot \text{LB} \_ \text{Field}(x, y, t)
      \]
    - And tune \(\mu\) (shape perceived fields) as needed
  - \(\mu = 0\)
    - Ignore load balancing, and do pure flocking
    - Small decrease in load balancing quality
  - \(\mu > 0\)
    - Flock accounting for crowd
    - Decrease in the accuracy of the flock formation
  - In any case, each single agent can “see” the individual fields and take actions accordingly
The CASCADAS Project

- Integrated Project to be funded by EU under FET “Situated and Autonomic Communication” Initiative
  - “Component-ware for Autonomic and Situation-Aware Communication Services”
  - 13 Partners
  - Starting January 2006
- General objective
  - Identify models and tools for a new generation of adaptable, self-organizing, context-aware communication services
  - Based on the central abstraction of ACE “autonomic communication element”, as the basic building block for complex service networks
  - Exploiting biologically and socially-inspired self-organization and self-management phenomena

The CASCADAS Approach

- Goes in the already stated directions
  - Level of services
  - Level of control
  - Exploiting sorts of shared knowledge networks for stigmergic cognitive interaction
- But the project hasn’t started yet...
Conclusion and Open Issues

- Engineering emergence is definitely a challenge
  - Producing self-organizing systems in a repeatable and measurable way
  - Controlling the continuous evolution and increase of complexity of existing systems

- Possible promising approaches include
  - Focusing at the “meso” scale
  - Promoting stigmergy and environment engineering (as in Co-Fields and TOTA, and as in the “knowledge networks” of CASCADAS)
  - Controlling the environment to control emergence