From Chemical Reactions to Ecosystems of Services
The SAPERE Approach

Giovanna Di Marzo Serugendo
University of Geneva, Switzerland
Giovanna.Dimarzo@unige.ch

University of Modena e Reggio Emilia, Italy
University of Bologna, Italy University of Geneva, Switzerland
University of St Andrews, UK Johannes Kepler Universitaet Linz, Austria
FET-Proactive Self-Awareness in Autonomic Systems

STREP – 3 years
www.sapere-project.eu

Outline

- SAPERE Project and Approach
  - LSAs and Chemical Reactions
- Case Studies
  - Crowd Steering through Public/Private Displays
  - Content Sharing
- Current Status
- From Bio-Inspired Design Patterns to Chemical reactions to Services
- Self-* Algorithms Examples with Eco-Laws

SAPERE Project

Goal

"Development of a highly-innovative theoretical and practical framework for the **decentralized** deployment, execution, and **management**, of **self-aware** and **adaptive** pervasive **services**"

Addresses

- Situation-awareness, adaptivity, autonomic behaviour - inherent properties of the service ecosystem,
- Management of complex pervasive service systems

Framework: SAPERE project

Ecosystem of Services

- Chemical Interactions among Services
  - Smooth data/service distinction
  - Spontaneous interactions of available services
    - **Bio-chemical reactions**
  - Middleware for Android phones / tablets
- **Context-awareness** (user, situation recognition)
- Case Study
  - Focus on public/private displays for crowd steering
- Domains
  - Context-Aware Advertisement, Crowd Steering, User guidance
- **EU Funded Project (SAPERE: http://www.sapere-project.eu)**
  - Collaboration: U Geneva, U Bologna, U Modena, U Linz, U St-Andrews
  - 2010-2013

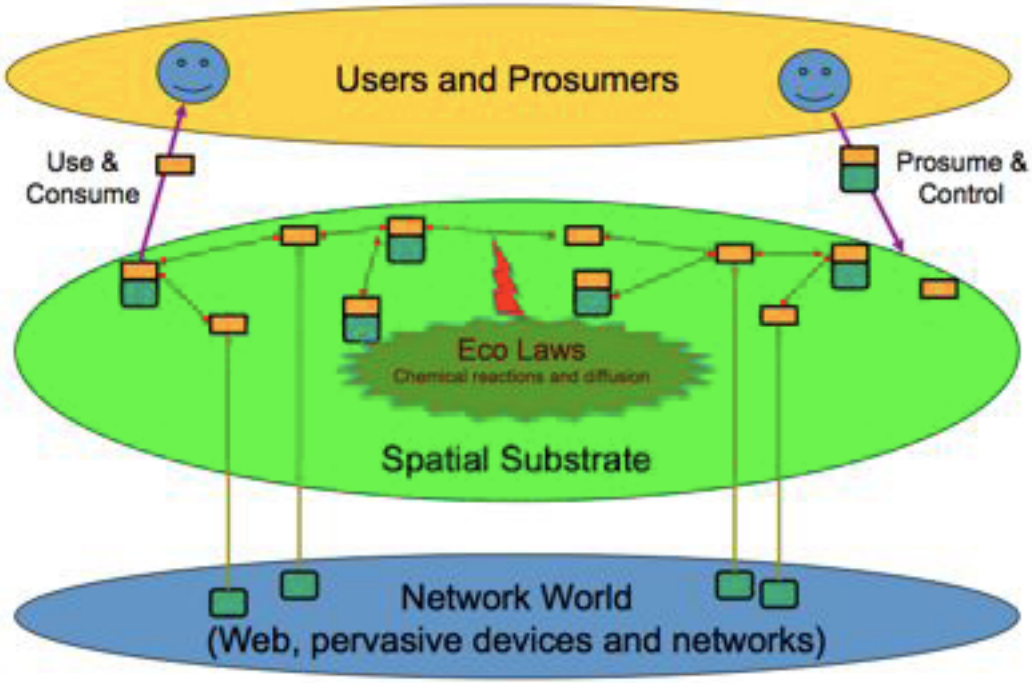
SAPERE Approach

SAPERE Concepts

- Ecosystem of services is ruled by "**eco-laws**"
  - Drive the dynamics of the system
  - Act like Chemical Reactions
  - Trigger when enabled (pattern-matching, rewriting rules)

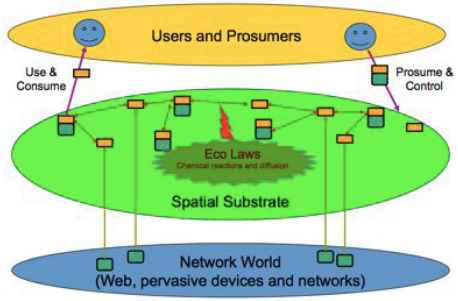
- Information about services, data, devices: "**Live Semantic Annotations (LSAs)"**
  - Reflect changes in services, data, devices

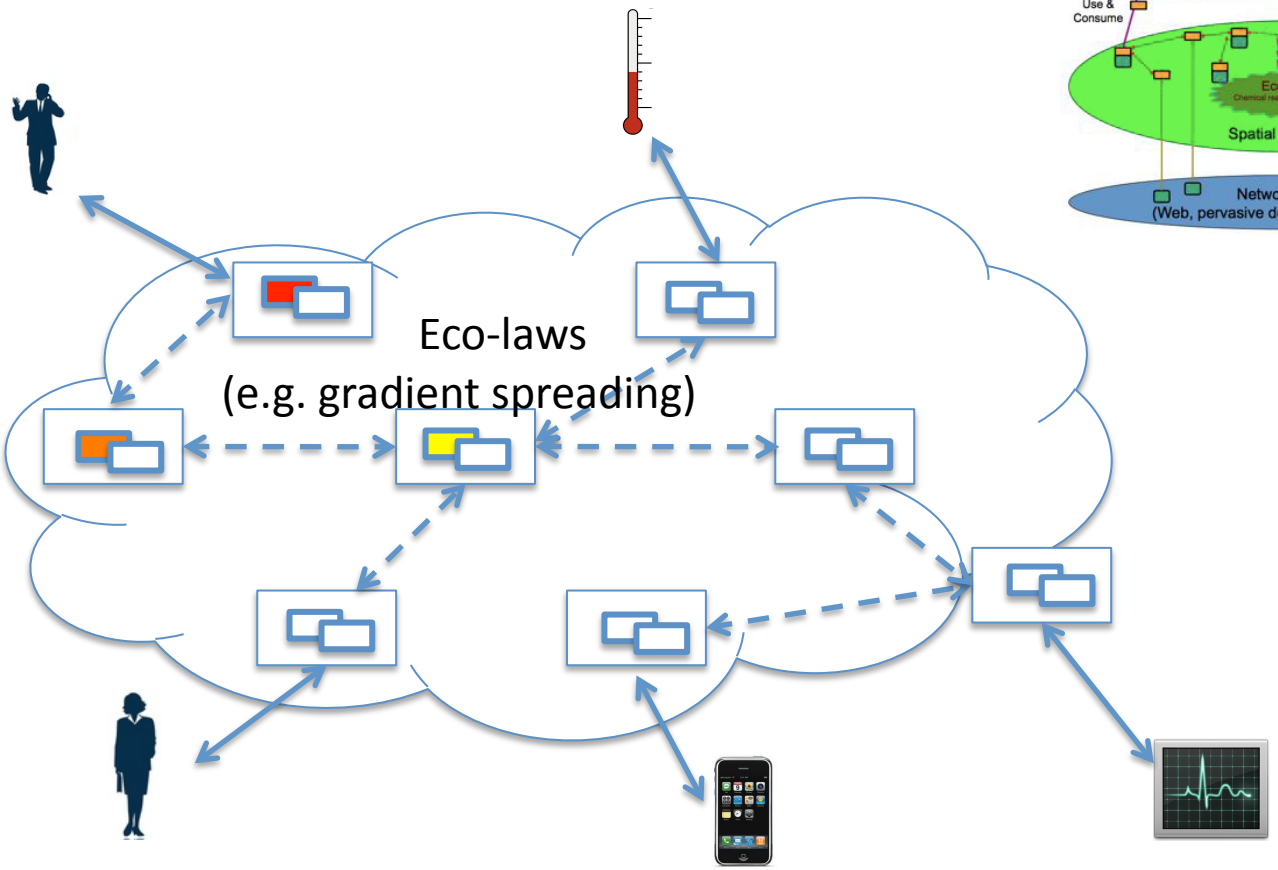
SAPERE Approach



SAPERE Approach

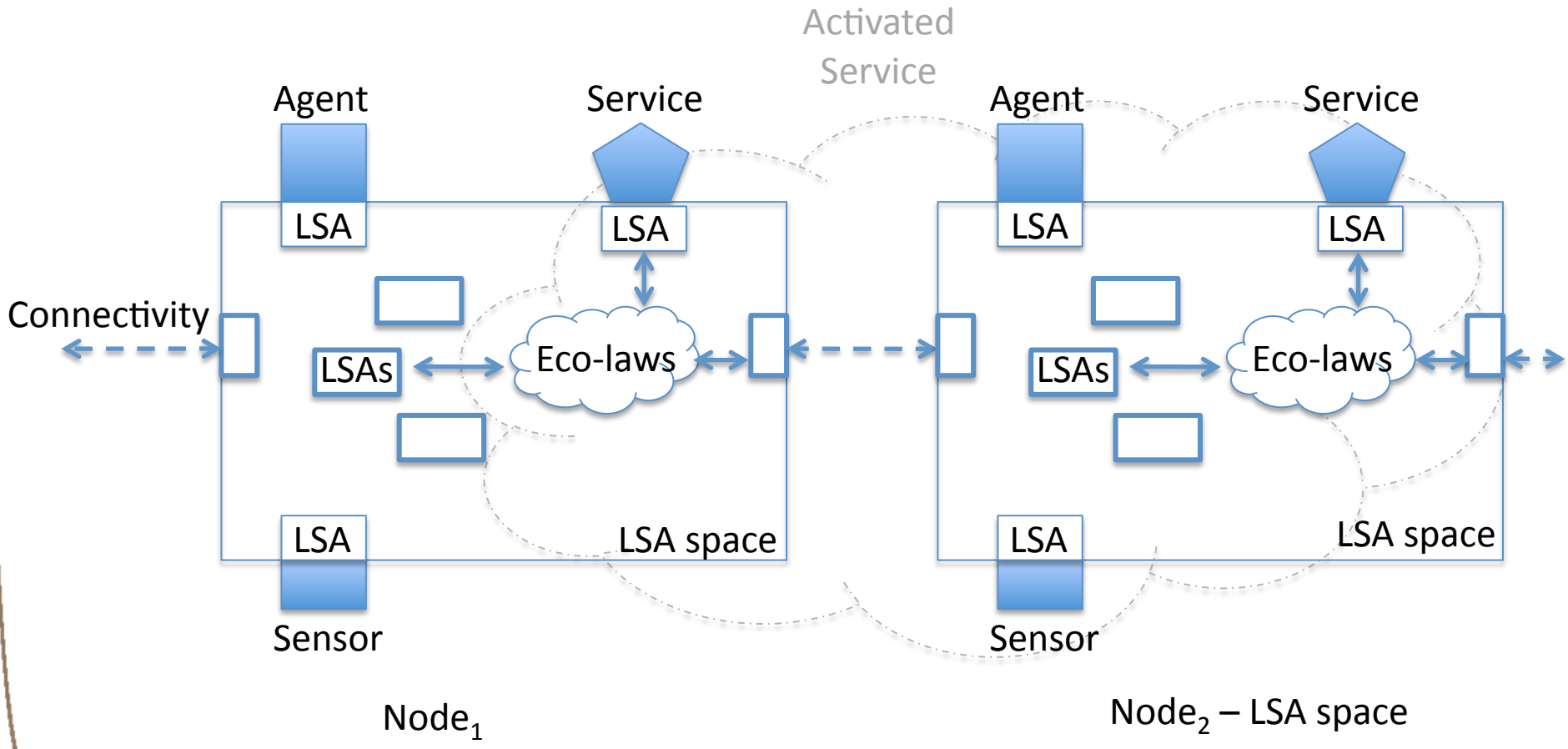
Node$_n$ - LSA space





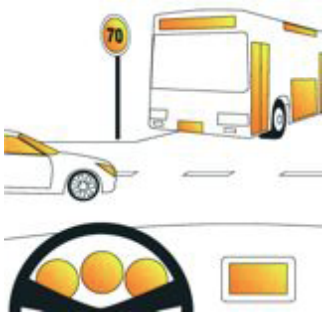
Network of Nodes
--
Relates to
Users
Sensors
Displays
Web

SAPERE Approach



Case Studies

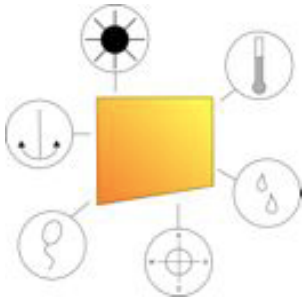
Public/Private Displays

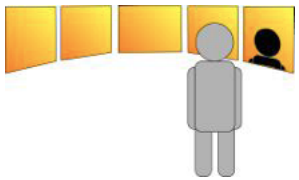


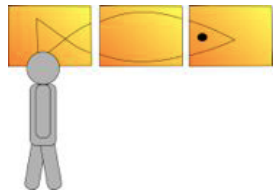
Case Studies

Self-Organising Context-Aware Public/Private Displays

- Collaborative Displays
- Context/Situation-Aware



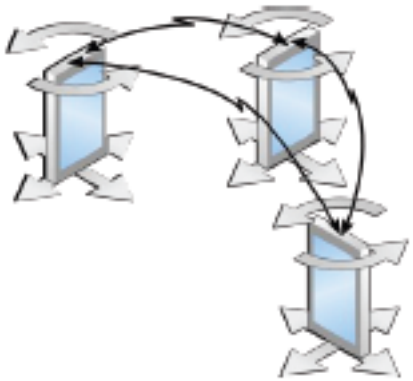


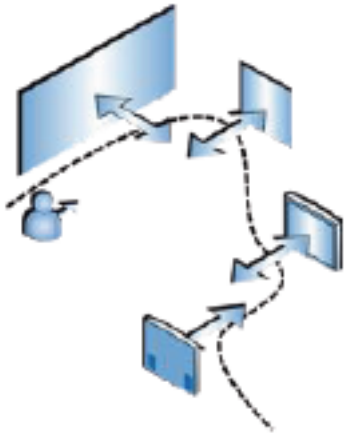


Case Studies

Crowd Steering through Self-Organising Public/Private Displays

- Collaborative displays
- Self-organising spontaneous interactions
  - Bio-inspired (gradients, gossip, stigmergy, flocking)

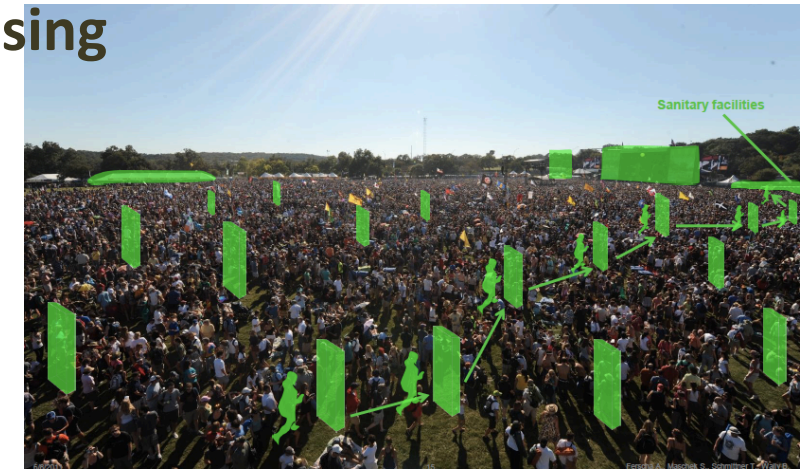


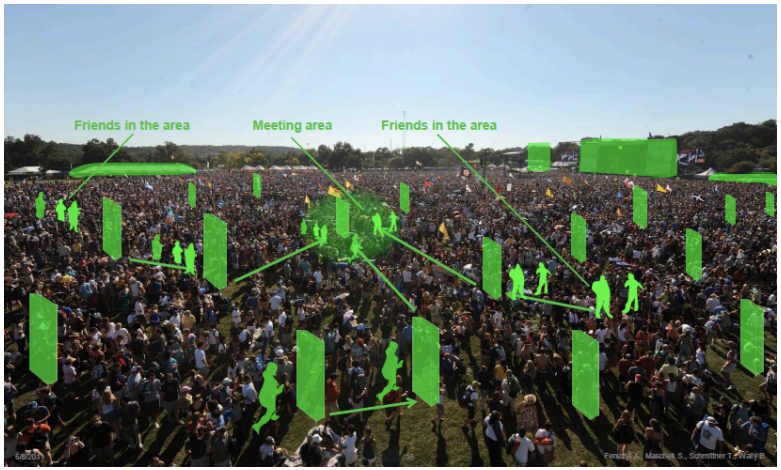


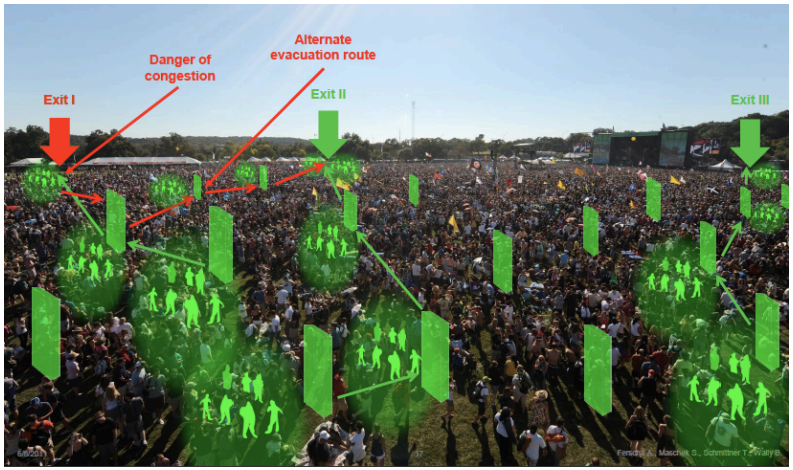
Case Studies

Crowd Steering through Self-Organising Public/Private Displays

- Single user
- Multiple user
- Emergency



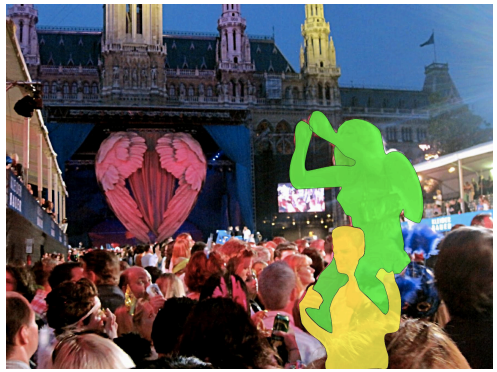


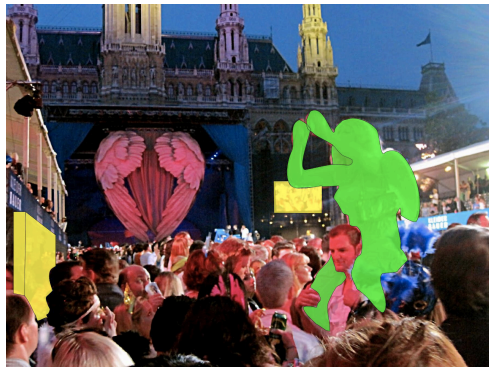


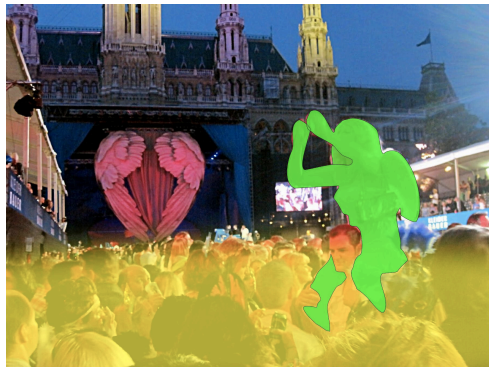
Case Studies

Content Sharing

- With Closest friends
- With Displays
- With Crowd







Current Status

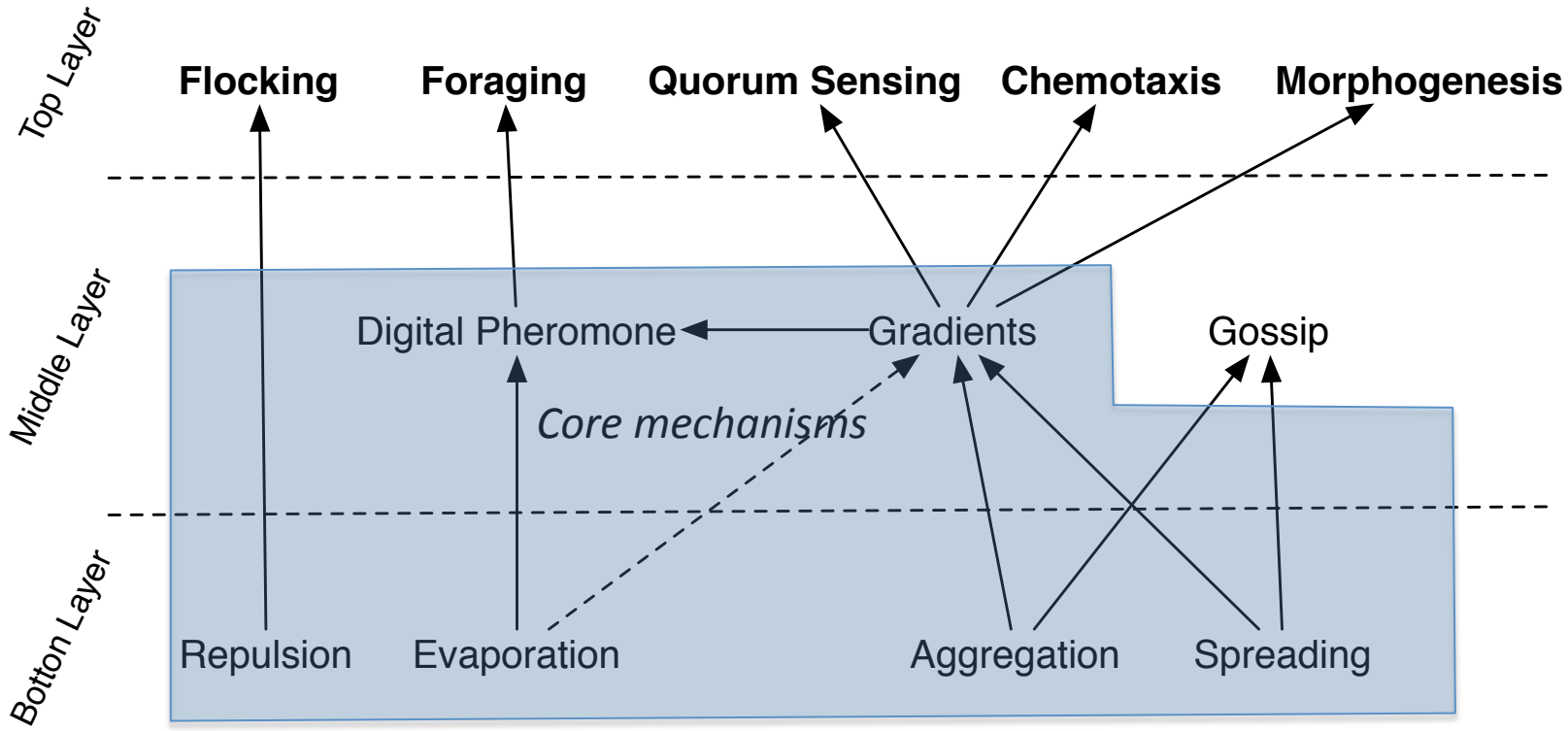
- **List of bio-inspired design patterns**
  - **Self-* algorithms as set of eco-laws**
- **From Eco-laws to Services**

- Operational language for eco-laws
  - Link with reasoning in SPARQL
- Middleware in preparation

Outline

- SAPERE Project and Approach
  - LSAs and Chemical Reactions
- Case Studies
  - Crowd Steering through Public/Private Displays
  - Content Sharing
- Current Status
- From Bio-Inspired Design Patterns to Chemical reactions to Services
- Self-* Algorithms Examples with Eco-Laws

Bio-Inspired Patterns



[Fernandez11a] [Fernandez11b]

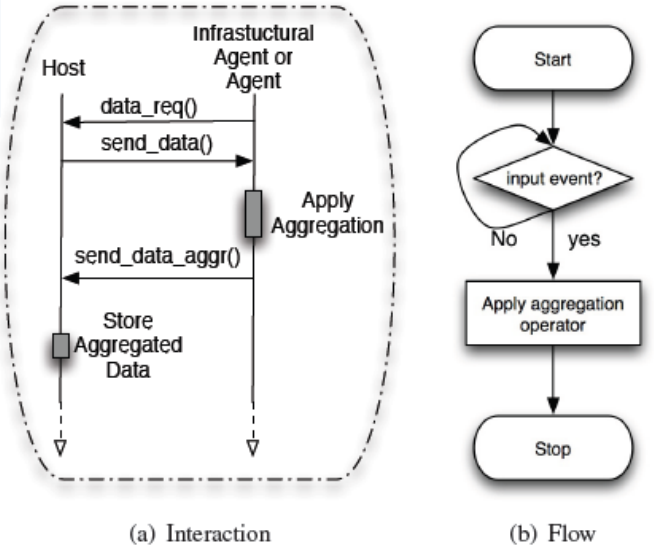
Design Patterns

- Design Patterns description
  - Abstract rules, sequence diagrams, explanations, implementation details

$$name :: data \xrightarrow{rate} action \quad if \quad condition$$

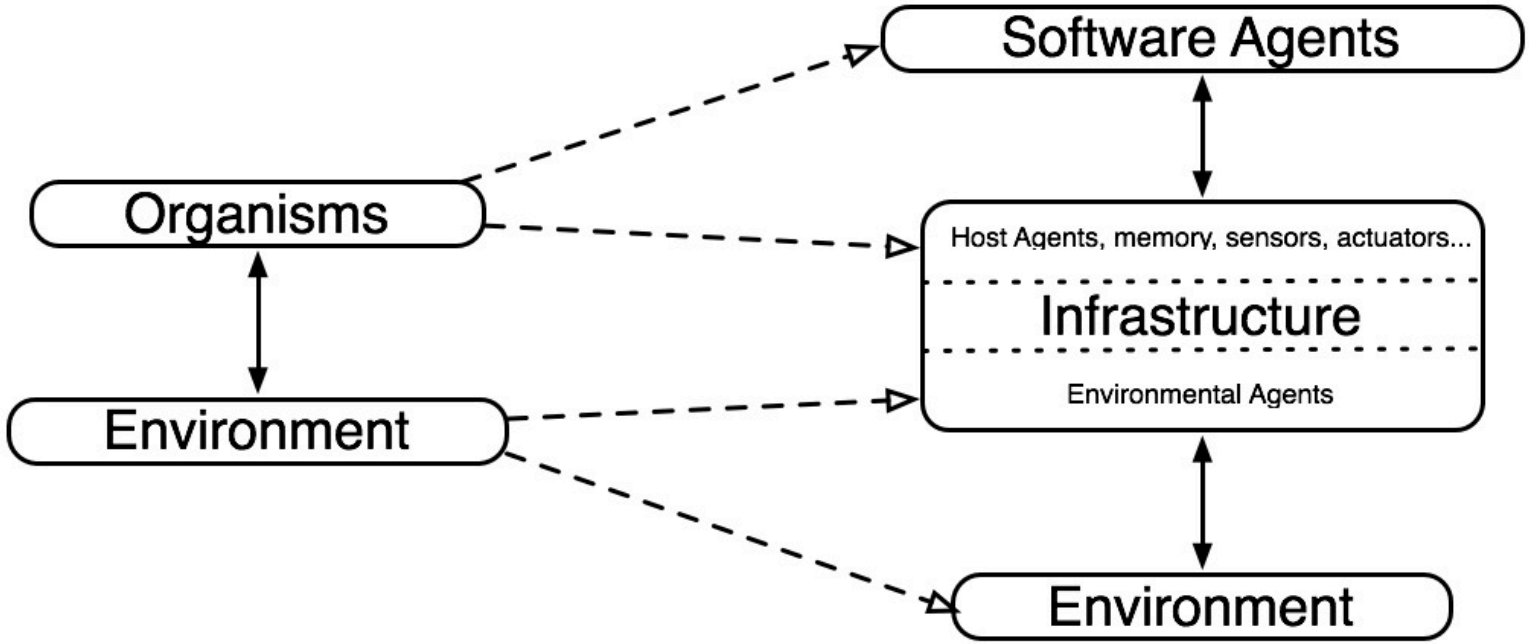
$$aggregation :: I \rightarrow op(I)$$

$$spreading :: inf \rightarrow send(inf, neighbors)$$



(a) Interaction (b) Flow

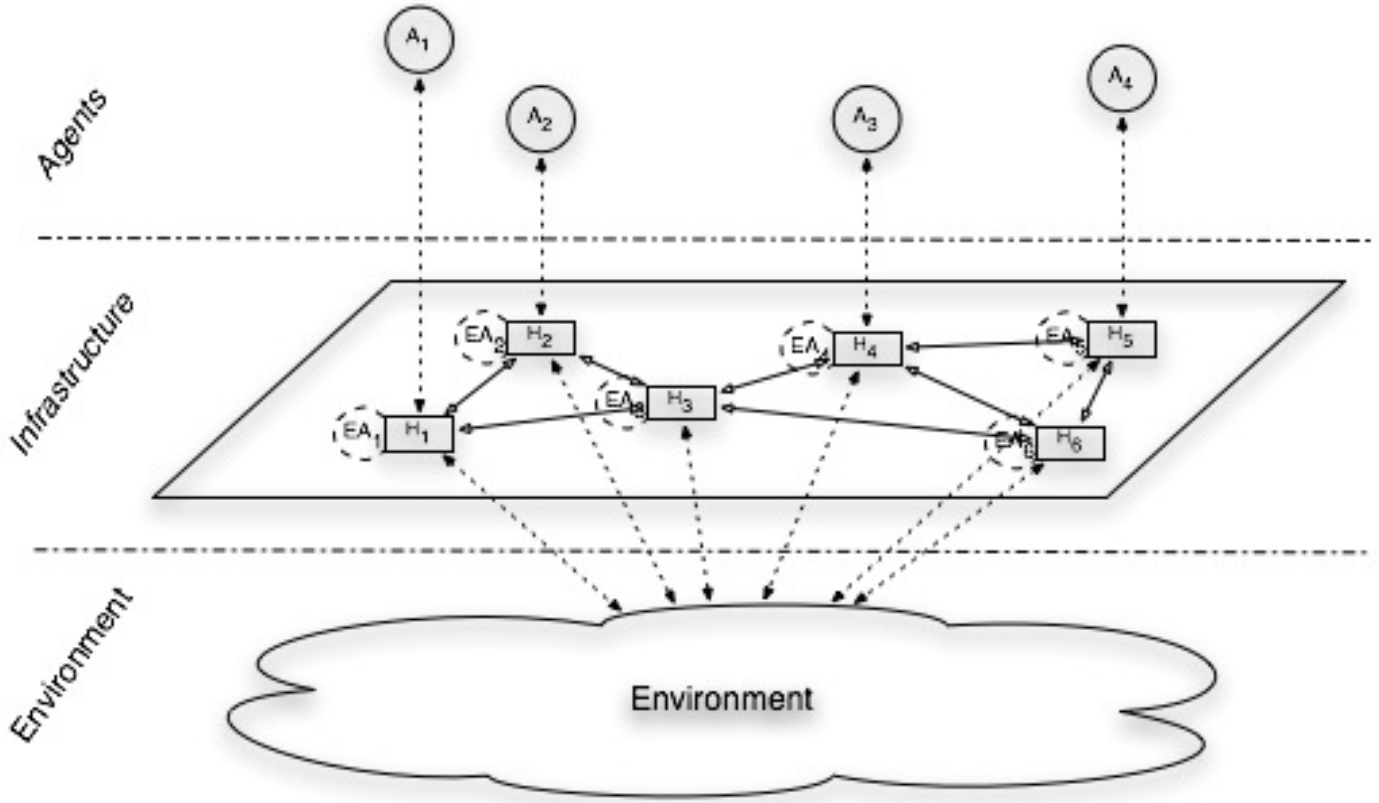
Computational Model (1)



(a) Biological Model

(b) Computational Model

Computational Model (2)



Computational Model (3)

- Software agents
  - Active autonomous entities (services, information, etc.)
- Host
  - Computing device hosting agent execution
    - E.g. sensors, PDAs, computers, …
- Environmental agents
  - Autonomous entity working on behalf of infrastructure
    - E.g. evaporating pheromone, updating gradients
- Environment
  - Anything else on which agents have no control
    - E.g. user switching device off, temperature, humidity, etc.

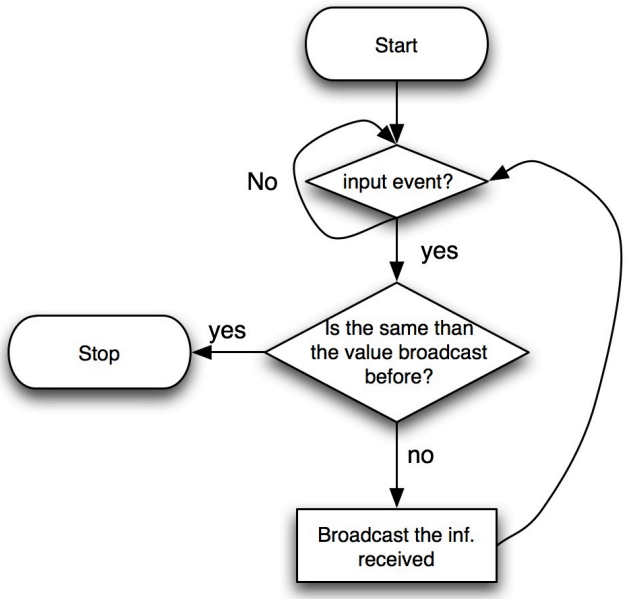
Spreading Pattern

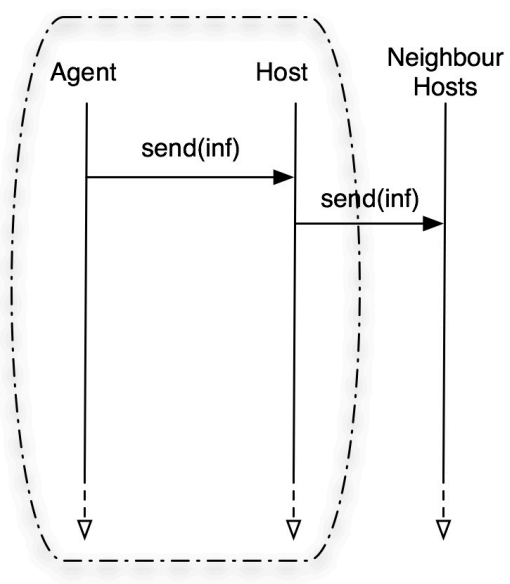
- **Problem:** in systems, where agents perform only local interactions, agents' reasoning suffers from the lack of knowledge about the global system.
- **Solution:** a copy of the information (received or held by an agent) is sent to neighbors and propagated over the network from one node to another. Information spreads progressively over the system and reduces the lack of knowledge of the agents while keeping the constraint of the local interaction.
- **Abstract Transition Rule:**

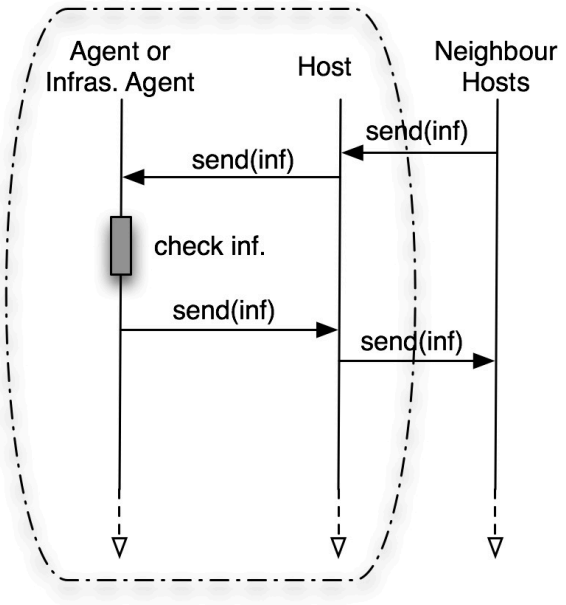
$$spreading :: inf \rightarrow send(inf, neighbors)$$

Spreading Pattern

- **Dynamics:**







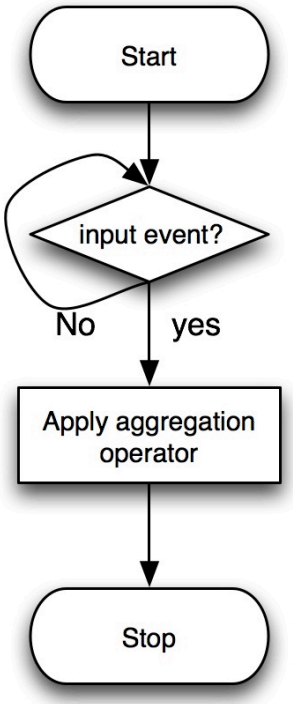
Aggregation Pattern

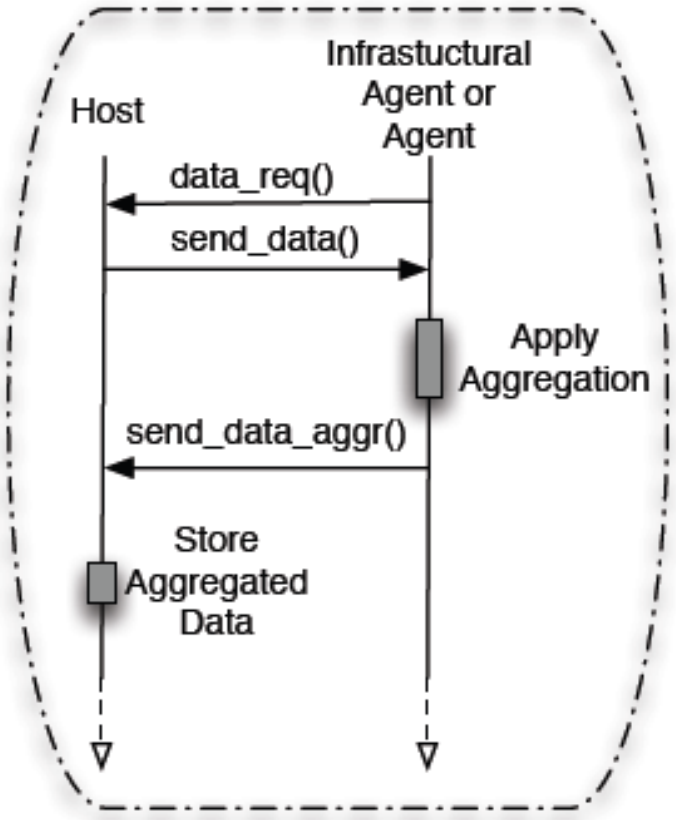
- **Problem:** in large systems, excess of information produced by the agents involves network and memory overloads. Information must be distributively processed in order to reduce the amount of information and to assess meaningful information.
- **Solution**: aggregation consists in locally applying an aggregation operator to process the information and synthesize macro information. This operator can take many forms, such as filtering, merging, aggregating or transforming.
- **Abstract Transition Rule:**

$$aggregation :: I \rightarrow op(I)$$

Aggregation Pattern

- **Dynamics:**





From Design Patterns to Chemical Reactions

- Bio-Inspired design patterns
  - Expressed with LSAs and Eco-Laws
- Eco-laws / LSA expressed with eco-law language
  - Generic form of an eco-law $P_1, \ldots, P_n \overset{r}{\mapsto} P'_1, \ldots, P'_m$
  - Matching LSAs on left-hand side and trigger law:
    - remove left-hand side LSAs,
    - add right-hand side LSA inside same node
  - Probabilistic rate r of firing eco-law
  - Adding LSA L in one neighbour at random $+\langle L \rangle$
  - Adding LSA L into specified neighbour N $N\langle L \rangle$
  - Adding LSA L into all neighbousr $bcast\langle L \rangle$
  - Assumptions
    - Access to neighbours nodes LSAs
    - Possibility to insert LSAs into neighbours nodes

From Design Patterns to Chemical Reactions

$$L, L' \xmapsto{R_{agg}} L$$

$$\langle chem, q, ts \rangle, \langle chem, q2, ts2 \rangle \xmapsto{R_{agg}} \langle chem, (q + q2), max(ts, ts2) \rangle$$

Aggregation

$$L \xmapsto{R_{evp}} L'$$

$$\langle chem, q, ts \rangle \xmapsto{R_{evp}} \langle chem, q * EvRate, ts \rangle$$

Evaporation

From Design Patterns to Chemical Reactions

$$G_i \xmapsto{R_{grSpr}} G_i, +G_j$$

$$G_i, G_j \xmapsto{R_{grAgg}} G_i$$

(Active) Gradient

$$\langle \mathbf{source}, g, max, A \rangle \xmapsto{R_{grInit}} \langle \mathbf{source}, g, max, A \rangle, \langle \mathbf{grad}, g, 0, max, A \rangle$$

Initialisation

$$\langle \mathbf{grad}, g, val, max, A \rangle \xmapsto{R_{grSpr}} \langle \mathbf{grad}, g, val, max, A \rangle, +\langle \mathbf{grad}, g, val{+}1, max, A \rangle$$

Spreading

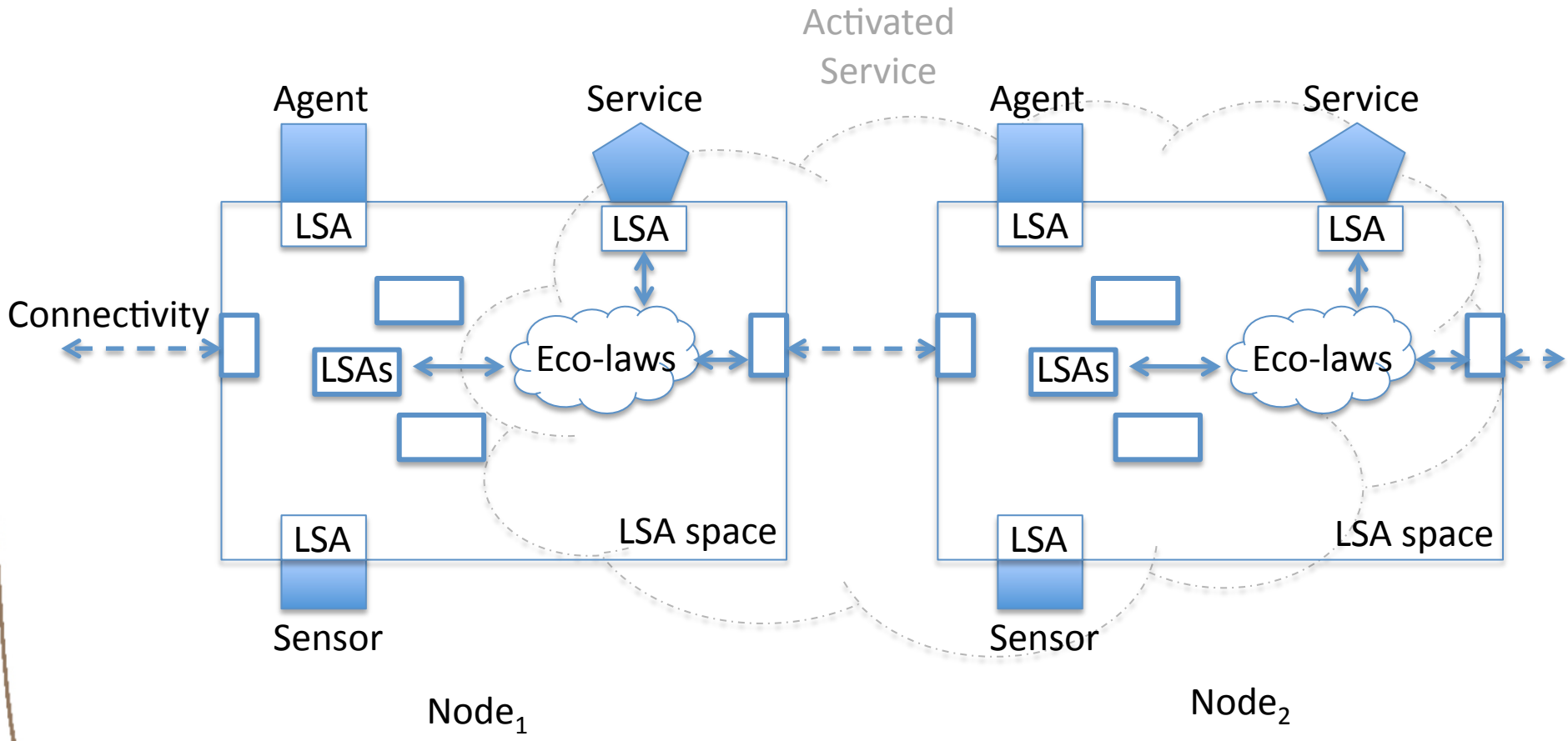
$$\langle \mathbf{grad}, g, val, max, A \rangle, \langle \mathbf{grad}, g, val{+}i, max, A \rangle \xmapsto{R_{grAgg}} \langle \mathbf{grad}, g, val, max, A \rangle$$

Aggregation

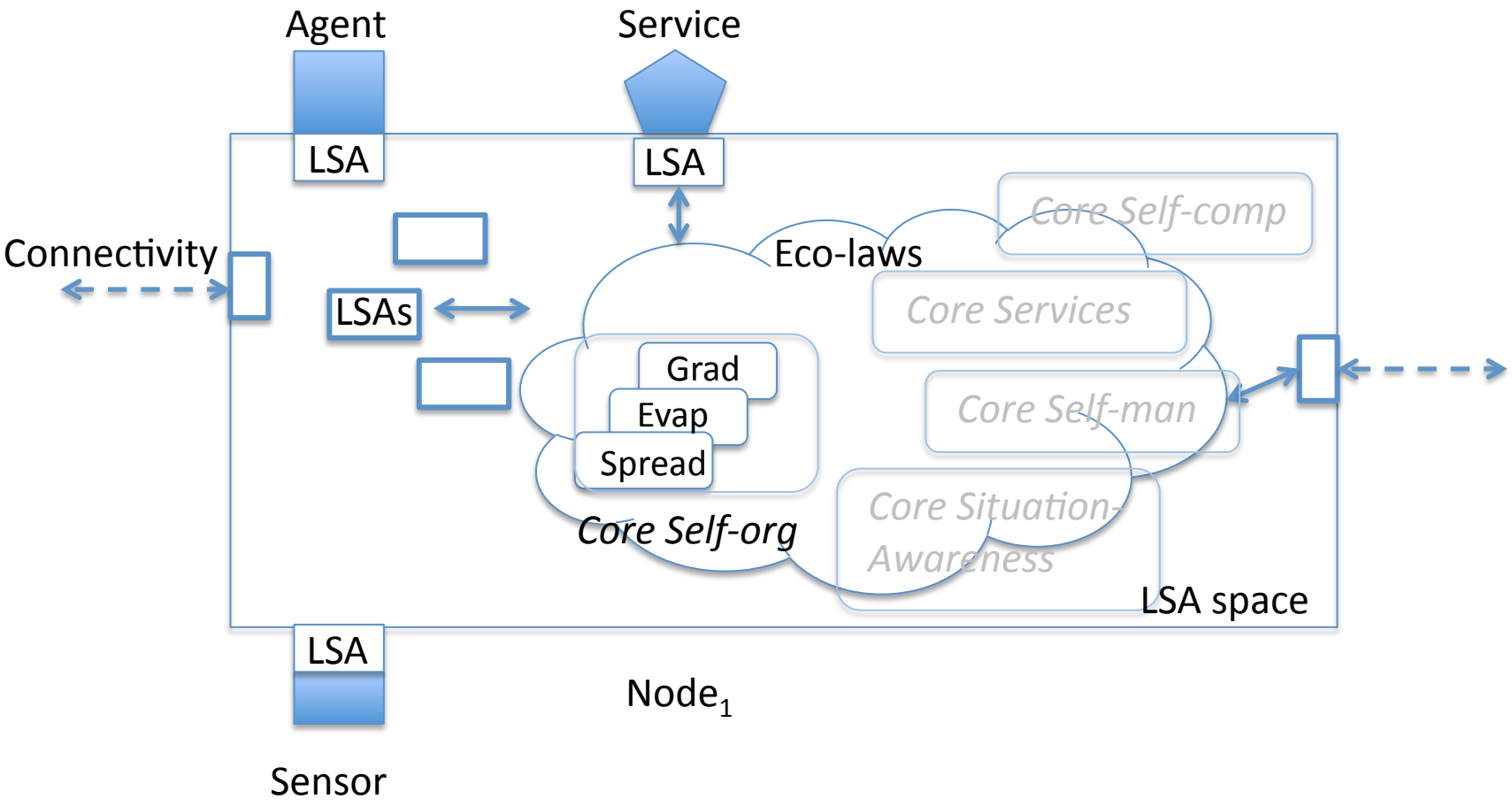
From Chemical Reactions to Services

- **First Step**: Core mechanisms expressed as generic eco-laws
  - Set of generic eco-laws triggered by appropriate LSAs
    - E.g generic Spreading Service, Gradient Service, etc.
- **Second Step**: Other mechanisms (or agents) use core laws
  - Two different mechanisms use both the same core mechanism
    - E.g. two mechanisms uses evaporation or spreading laws
  - A service submitted to the laws of two different core mechanisms
    - E.g. an agent relies on both Spreading and Evaporation
- **Third Step**: dynamic composition of services using self-compositions laws

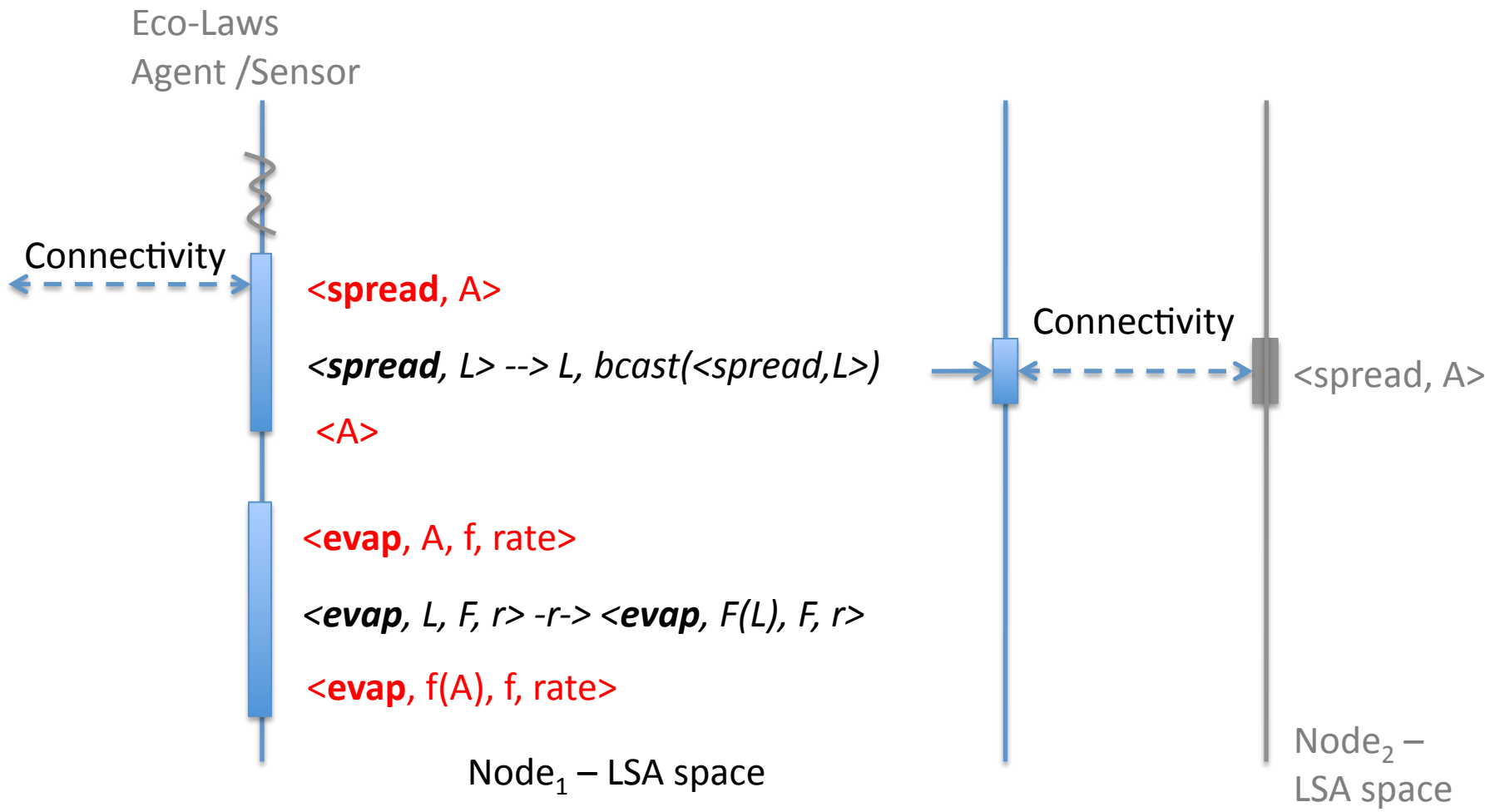
SAPERE Approach



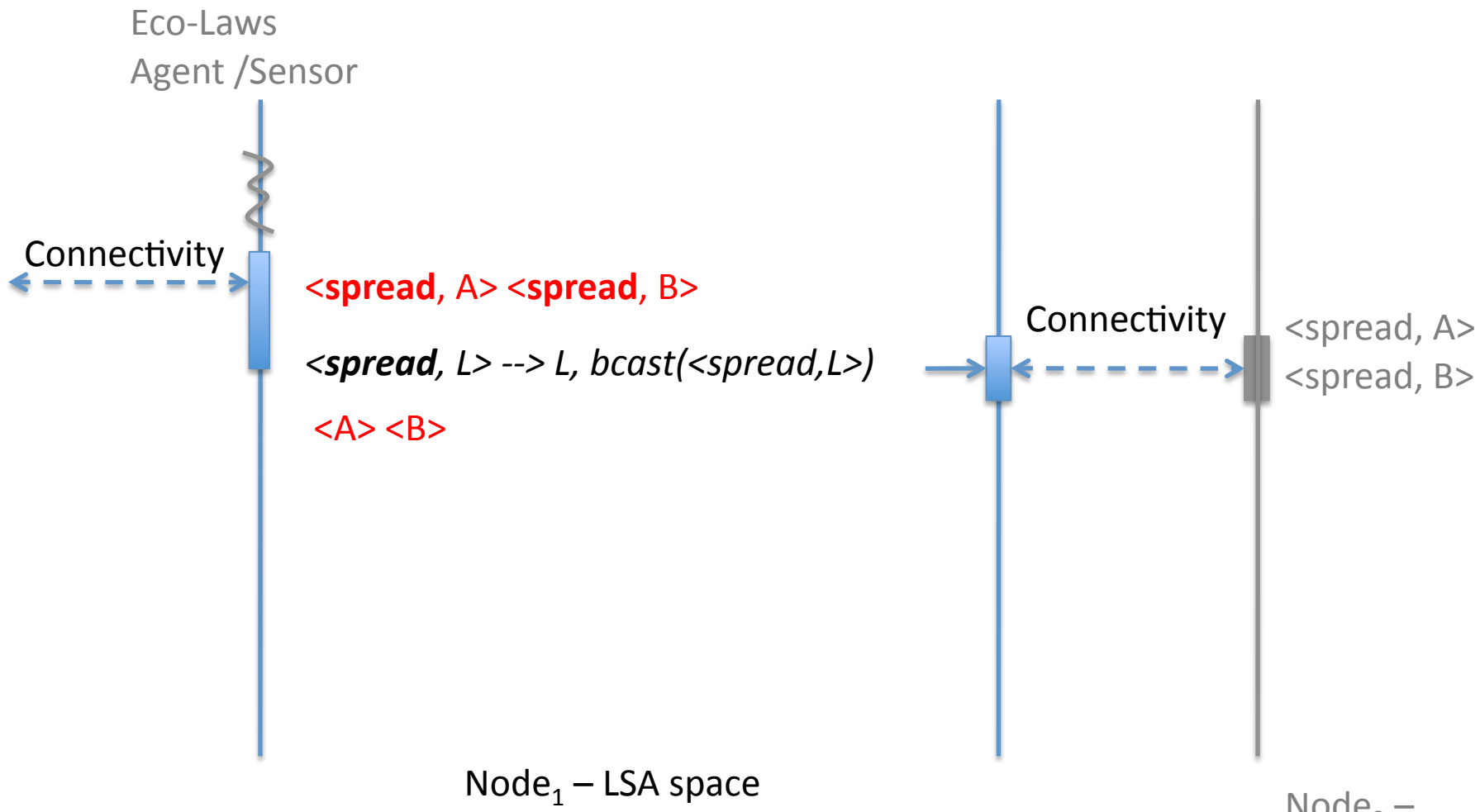
Core Mechanisms as Eco-laws



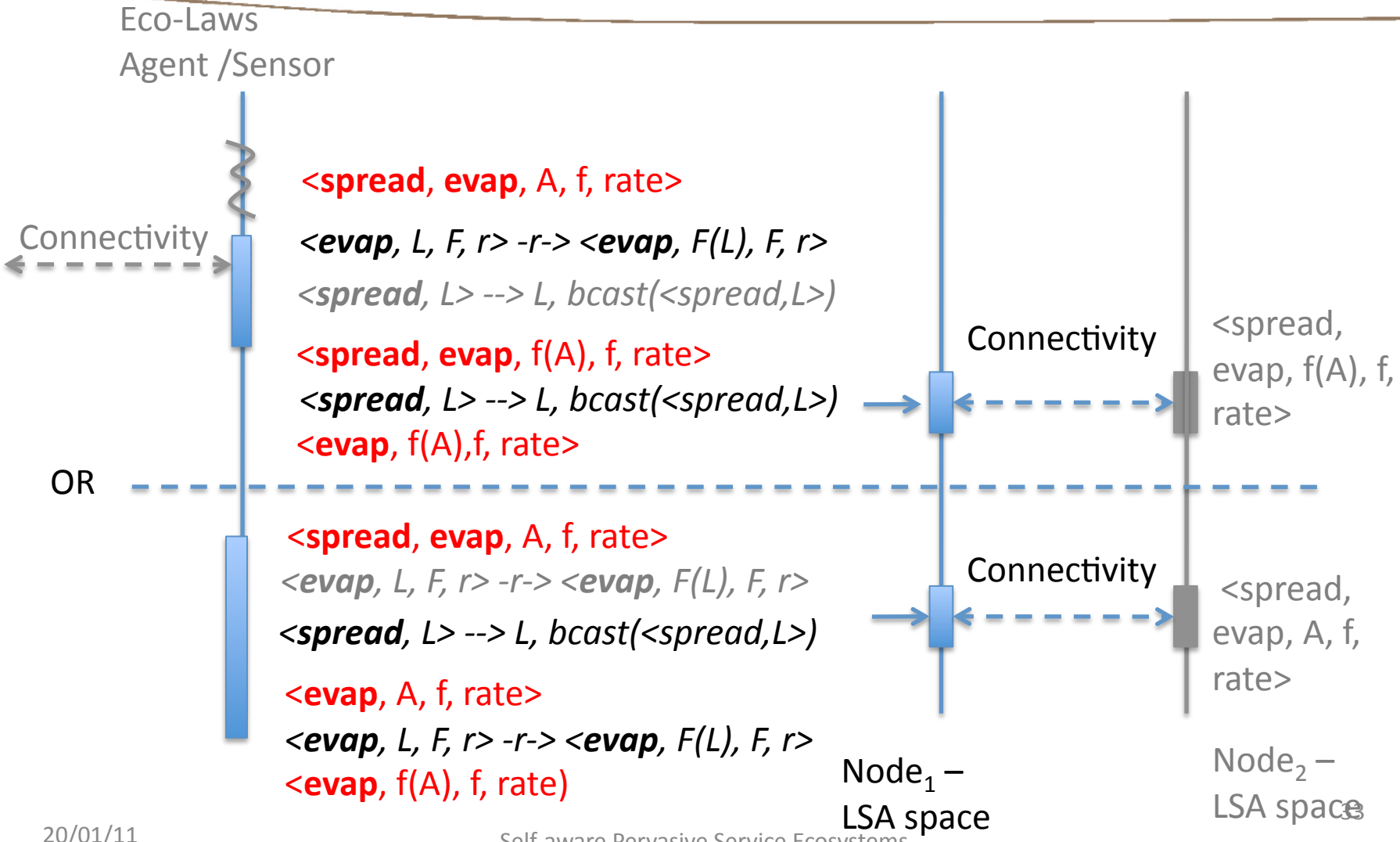
Mechanisms / Agents use Core Laws



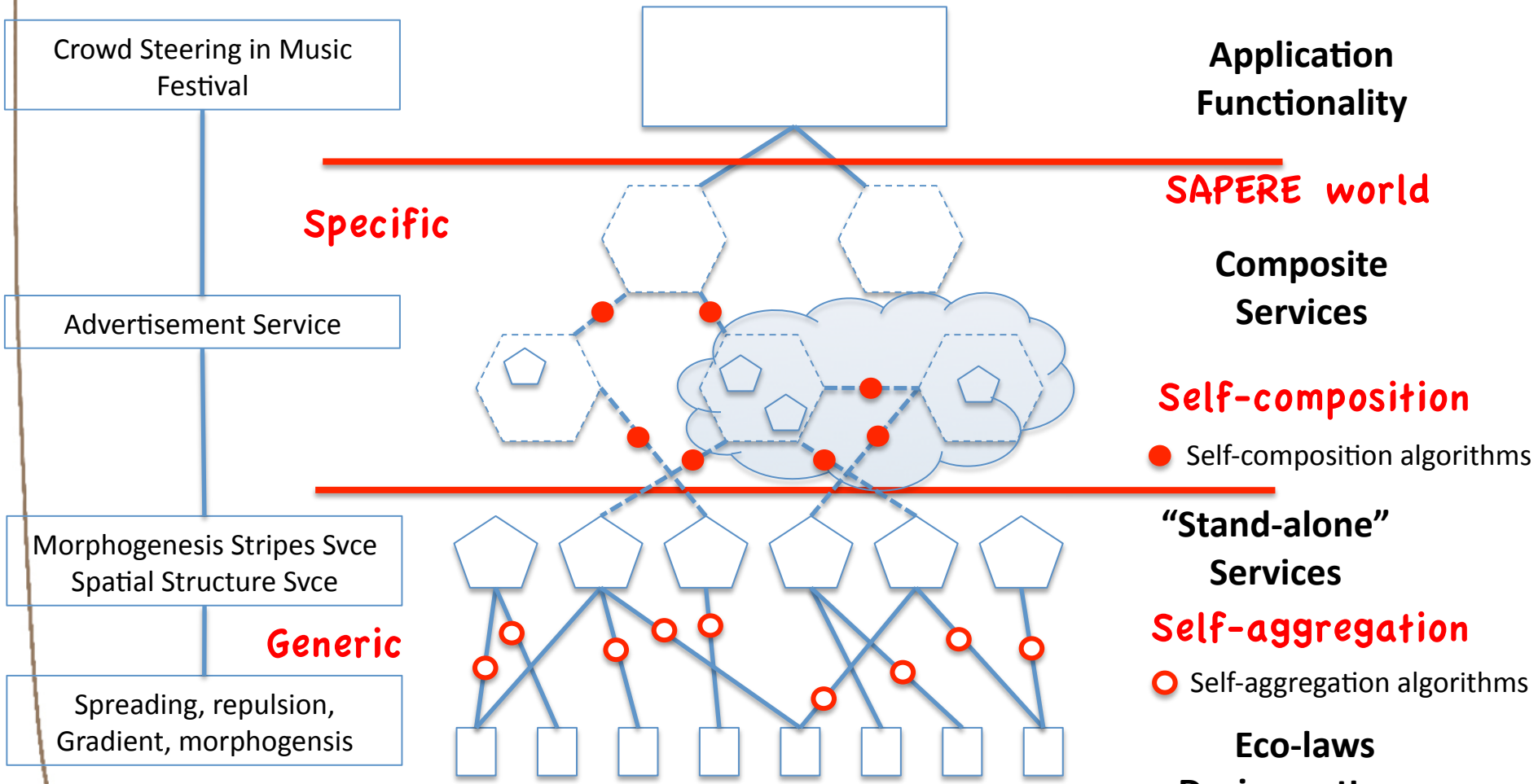
Mechanisms / Agents use Core Laws



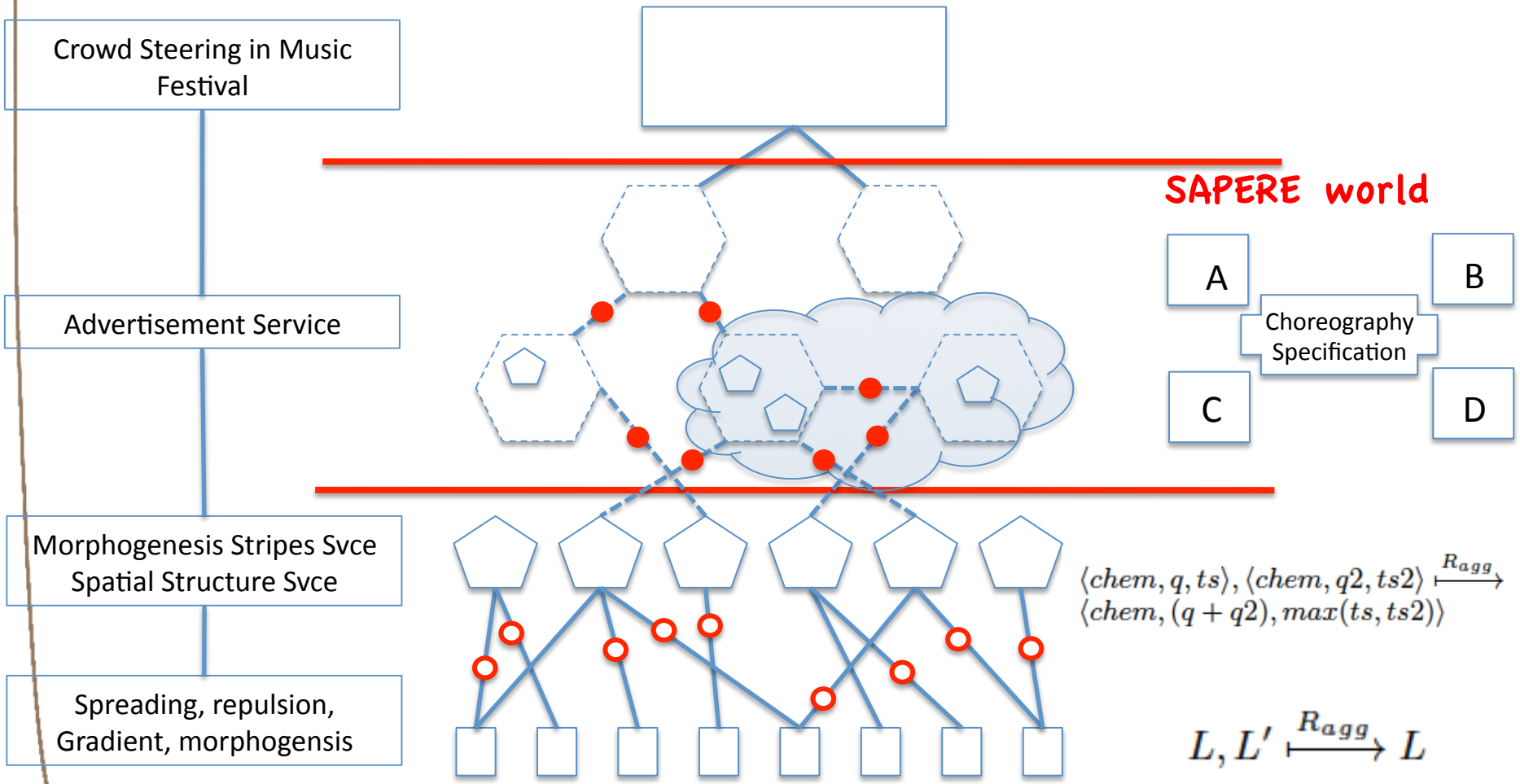
Mechanisms / Agents use Core Laws



From patterns to services

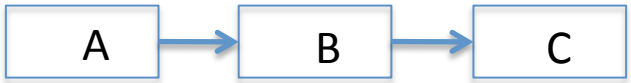


From design patterns to services

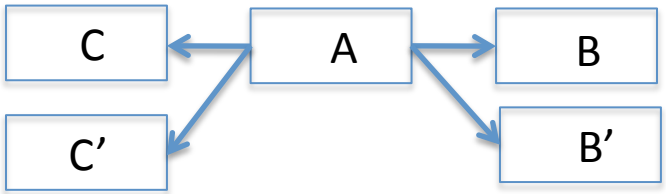


Services Composition

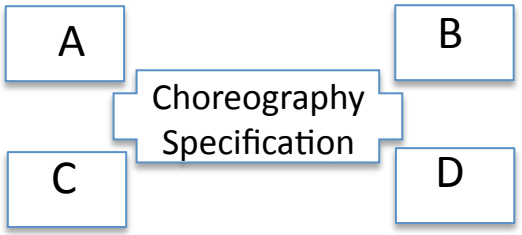
- Chains:
  - Service A requires functionality F provided by Service B
  - Service B requires functionality G provided by Service C



- Groups:
  - Service A requires both functionality F and G provided by Service B and C



  - Choreography:
    - specification of roles, services attach themselves to the roles
  - Chemical reactions [Frei2010]
    - A expresses needed functionality F
      B reacts to A through F if it can provide F
    - A is either a service or a "task" in a flow

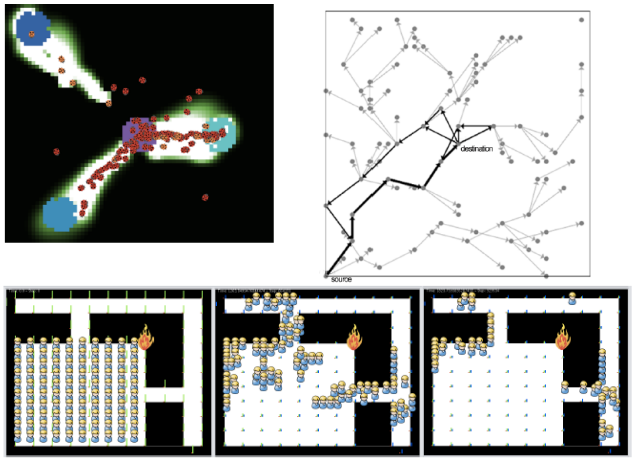


Outline

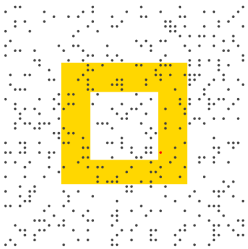
- SAPERE Project and Approach
  - LSAs and Chemical Reactions
- Case Studies
  - Crowd Steering through Public/Private Displays
  - Content Sharing
- Current Status
- From Bio-Inspired Design Patterns to Chemical reactions to Services
- Self-* Algorithms Examples with Eco-Laws

Self-* Algorithms

- Available [Montagna11, Tchao11]
  - AODV – Manet routing protocol (spreading)
  - Foraging (Digital pheromone, gradient)
  - Firefly (Aggregation, Spreading, Gossip)
  - Evacuation scenario (Gradient)
  - Hoverinfo service (spreading, repulsion)



- In preparation
  - Morphogenesis stripes service (gradient, morphoge
  - Prokariotis (chemotaxis)
  - Presence detection service



- Evaluation (preliminary):
  - convergence, scalability, speed of convergence, stability

Simulators

- Alchemist
  - Alchemist – inspired from chemistry, follows continuous time Markov chains
  - Firing of eco-laws follows a probability, which depends on "concentration" of LSAs enabling eco-laws (and follows CTMC model)
- Repast-based
  - Java
  - "Repast" – inspired by distributed / concurrent systems, follows discrete time, evaluation of eco-laws at simulation "ticks", firing of all enabled eco-laws provided concurrency is ensured (rate of eco-law is also taken into account)

Examples

Examples of Self-organising systems / algorithms with eco-laws and LSAs

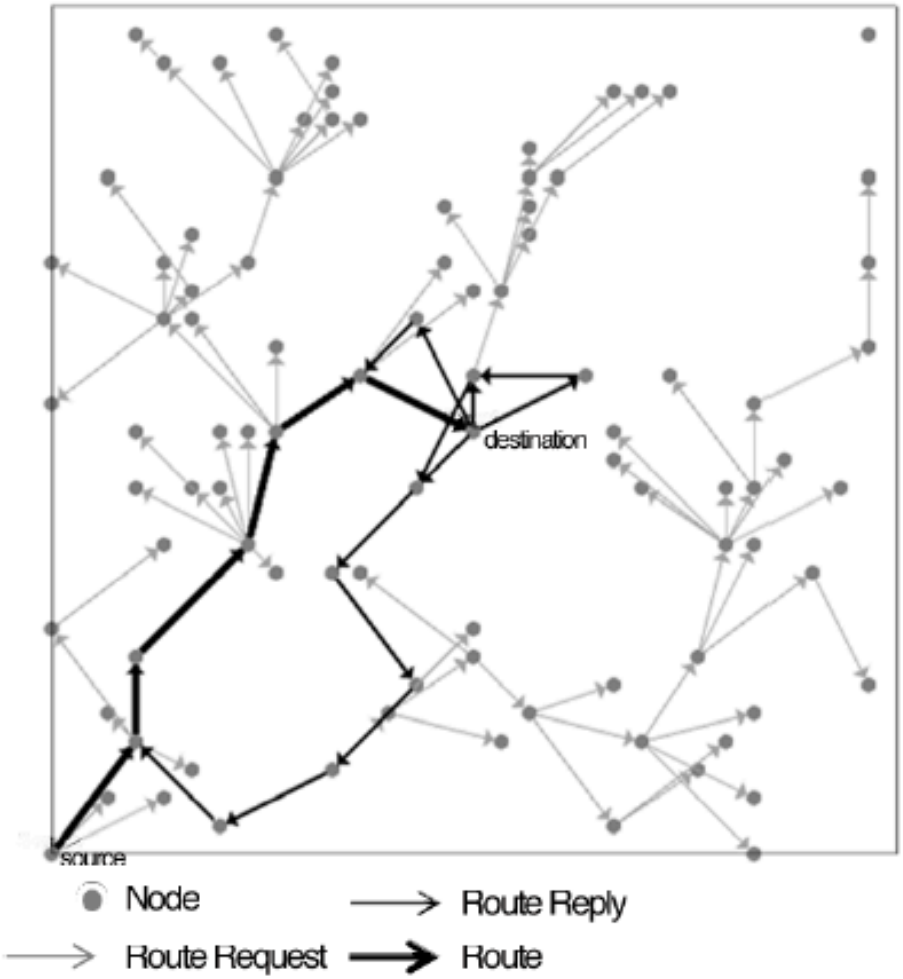
- AODV
- Ant foraging
- Firefly synchronisation
- Hovering Information

AODV

Goal: sending MSG from source node to a dest node

- Discovering a route (series of broadcasts from source to dest): RREQ
- Intermediary nodes keep track of the previous node in the chain who sent them the RREQ
- The destination node, receiving RREQ, replies with RREP
- An intermediary node, knowing the route, replies RREP
- Establishing the “shortest” and “freshest” route: RREP following the way back along the RREQ
- (Route updated while nodes are moving)

AODV



AODV

LSAs:

- Initial LSA triggering the service:

$$\langle ID, dest, \texttt{MSG} \rangle$$

- LSA for RREQ

$$\langle reqID, \texttt{RREQ}, source, dest, senderID, processed \rangle$$

- LSA for RREP

$$\langle \texttt{RREP}, source, dest, senderID, DSN, hopcount, expiry \rangle$$

Functions:

```
nxtReqID()
nId()
```

AODV

Eco-Laws:

- Create RREQ

$$\langle id, t, \texttt{MSG} \rangle \mapsto \\ \langle id, t, \texttt{MSG} \rangle, \langle \texttt{nxtReqID}(), \texttt{RREQ}, \texttt{nId}(), t, \texttt{nId}(), 0 \rangle$$

- Broadcast and tag RREQ

$$\langle reqID, \texttt{RREQ}, s, t, sen, 0 \rangle \mapsto \\ \langle reqID, \texttt{RREQ}, s, t, sen, 1 \rangle, bcast \langle reqID, \texttt{RREQ}, s, t, \texttt{nId}(), 0 \rangle$$

- Aggregate RREQ

$$\langle reqID, \texttt{RREQ}, s, t, sen, 0 \rangle, \langle reqID, \texttt{RREQ}, s, t, sen, 1 \rangle \mapsto \\ \langle reqID, \texttt{RREQ}, s, t, sen, 1 \rangle$$

- Send RREP

$$\langle reqID, \texttt{RREQ}, s, \texttt{nId}(), sen, 0 \rangle \mapsto \\ sen \langle \texttt{RREP}, s, \texttt{nId}(), \texttt{nId}(), \texttt{nxtDsn}(), 1, \texttt{time}() + \#LIFE \rangle$$

AODV

Eco-Laws:

- Unicast RREP

$$\langle \texttt{RREP}, s, t, sen, dsn, hop, exp\rangle,$$
$$\langle reqID, \texttt{RREQ}, s, t, sen2, 1\rangle \mapsto$$
$$\langle \texttt{RREP}, s, t, sen, dsn, hop, exp\rangle,$$
$$\langle reqID, \texttt{RREQ}, s, t, sen2, 1\rangle,$$
$$sen2\langle \texttt{RREP}, s, t, \texttt{nId}(), dsn, hop{+}\texttt{1}, exp\rangle$$

- Aggregate RREP: “Freshest” route

$$\langle \texttt{RREP}, s, t, sen, dsn, hop, ts\rangle,$$
$$\langle \texttt{RREP}, s, t, sen2, (dsn2 : dsn2{>}dsn), hop2, ts2\rangle \mapsto$$
$$\langle \texttt{RREP}, s, t, sen2, dsn2, hop2, ts2\rangle$$

AODV

Eco-Laws:

- Aggregate RREP : “Shortest” route

$$\langle \texttt{RREP}, s, t, sen, dsn, hop, ts\rangle,$$
$$\langle \texttt{RREP}, s, t, sen2, dsn, hop{+}n, ts2\rangle \mapsto$$
$$\langle \texttt{RREP}, s, t, sen, dsn, hop, ts\rangle$$

- Unicast the Message

$$\langle id, t, \texttt{MSG}\rangle, \langle \texttt{RREP}, s, t, sen, dsn, hop, exp\rangle \mapsto$$
$$sen\langle id, t, \texttt{MSG}\rangle, \langle \texttt{RREP}, s, t, sen, dsn, hop, exp\rangle$$

- Evaporation

$$\langle \texttt{RREP}, s, t, sen, dsn, hop, (exp : exp \leq \texttt{time}())\rangle \mapsto$$

Ant Foraging

LSA:

- Food

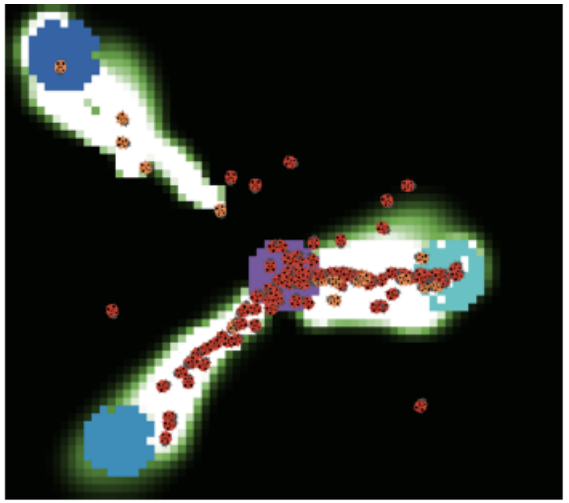
  $\langle \texttt{info}, \texttt{food}, \texttt{value}, \texttt{timestamp} \rangle$

- Nest and Pheromone Gradient

  $\langle \texttt{grad}, \texttt{nest}, \texttt{value}, \texttt{max}, \texttt{annealing} \rangle$
  $\langle \texttt{grad}, \texttt{chemical}, \texttt{value}, \texttt{max}, \texttt{annealing} \rangle$
  $\langle \texttt{info}, \texttt{chemical}, \texttt{value}, \texttt{timestamp} \rangle$

- Ant Carrying Food

  $\langle \texttt{info}, \texttt{ant}, \texttt{value}, \texttt{timestamp} \rangle$



Ant Foraging

Eco-Laws:

- Establish nest gradient

$$\langle \texttt{source}, \texttt{nest}, M, A \rangle \mapsto \langle \texttt{source}, \texttt{nest}, M, A \rangle, \langle \texttt{grad}, \texttt{nest}, 0, M, A \rangle$$

$$\langle \texttt{grad}, \texttt{nest}, V, M, A \rangle \mapsto \langle \texttt{grad}, \texttt{nest}, V, M, A \rangle, +\langle \texttt{grad}, \texttt{nest}, V+1, M, A \rangle$$

$$\langle \texttt{grad}, \texttt{nest}, V, M, A \rangle, \langle \texttt{grad}, \texttt{nest}, V+N, M, A \rangle \mapsto \langle \texttt{grad}, \texttt{nest}, V, M, A \rangle$$

Ant Foraging

Eco-Laws:

- Diffuse chemical gradient

$$\langle \texttt{grad}, \texttt{chem}, V, M, A\rangle \mapsto bcast\langle \texttt{grad}, \texttt{chem}, V, M, A\rangle$$
$$\langle \texttt{grad}, \texttt{chem}, V, M, A\rangle, \langle \texttt{grad}, \texttt{chem}, V+N, M, A\rangle \mapsto \langle \texttt{grad}, \texttt{chem}, V+N, M, A\rangle$$

- Evaporate chemical

$$\langle \texttt{grad}, \texttt{chem}, V, M, A\rangle \mapsto \langle \texttt{grad}, \texttt{chem}, V * \#ERATE), M, A\rangle$$

- Follow chemical gradient

$$\langle \texttt{info}, \texttt{ant}, \texttt{false}, TS\rangle,$$
$$\langle \texttt{grad}, \texttt{chem}, (V{:}V \ in \ [0.5{-}2.0]), M, A\rangle,$$
$$+\langle \texttt{grad}, \texttt{chem}, V{+}N, M, A\rangle \mapsto$$
$$+\langle \texttt{info}, \texttt{ant}, \texttt{false}, TS\rangle, \langle \texttt{grad}, \texttt{chem}, V, M, A\rangle,$$
$$+\langle \texttt{grad}, \texttt{chem}, V{+}N, M, A\rangle$$

Ant Foraging

Eco-Laws:

- Wiggle at random

$$\langle \texttt{info}, \texttt{ant}, \texttt{false}, TS \rangle \mapsto +\langle \texttt{info}, \texttt{ant}, \texttt{false}, TS \rangle$$

- Pick up food

$$\langle \texttt{info}, \texttt{ant}, \texttt{false}, TS \rangle, \langle \texttt{info}, \texttt{food}, V{+}1, TS2 \rangle \mapsto \\ \langle \texttt{info}, \texttt{ant}, \texttt{true}, TS \rangle, \langle \texttt{info}, \texttt{food}, V, TS2 \rangle$$

- Return to nest, follow nest gradient

$$\langle \texttt{info}, \texttt{ant}, \texttt{true}, TS \rangle, \langle \texttt{grad}, \texttt{nest}, V{+}1, M, A \rangle, \\ +\langle \texttt{grad}, \texttt{nest}, V, M, A \rangle \mapsto \\ +\langle \texttt{info}, \texttt{ant}, \texttt{true}, TS \rangle, \langle \texttt{grad}, \texttt{nest}, V{+}1, M, A \rangle, \\ +\langle \texttt{grad}, \texttt{nest}, V, M, A \rangle, \langle \texttt{info}, \texttt{chem}, 60, TS \rangle$$

Ant Foraging

Eco-Laws:

- Aggregate chemical

$$\langle \texttt{info}, \texttt{chem}, Q, TS\rangle, \langle \texttt{grad}, \texttt{chem}, C, M, A\rangle \mapsto \\ \langle \texttt{grad}, \texttt{chem}, \texttt{min}(C{+}Q, M), M, A\rangle$$

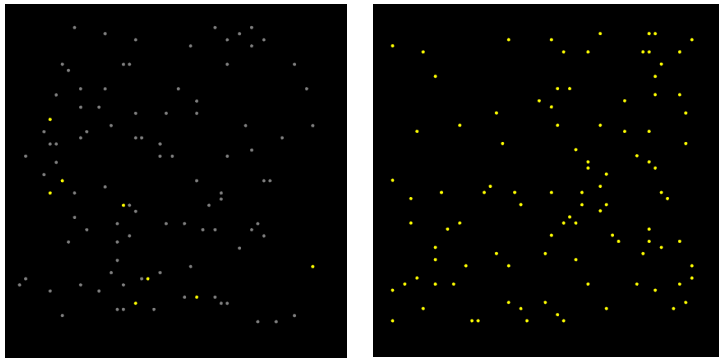
- Returning to the nest

$$\langle \texttt{info}, \texttt{ant}, \texttt{true}, TS\rangle \mapsto \\ +\langle \texttt{info}, \texttt{ant}, \texttt{true}, TS\rangle, \langle \texttt{info}, \texttt{chem}, 60, TS\rangle$$

- Leave food in the nest

$$\langle \texttt{info}, \texttt{ant}, \texttt{true}, TS\rangle, \langle \texttt{grad}, \texttt{nest}, 0, M, A\rangle \mapsto \\ \langle \texttt{info}, \texttt{ant}, \texttt{false}, TS\rangle, \langle \texttt{grad}, \texttt{nest}, 0, M, A\rangle$$

Fireflies flashes synchronization



LSA:

$$\langle \mathtt{ID}, \mathtt{phase}, \mathtt{cycle} \rangle$$

Eco-Laws:

- Phase advance + share

$$\langle ID, p, c \rangle \mapsto +\langle ID, (p + 1 \; mod \; c), c \rangle$$

- Synchronization

$$\langle \mathtt{ID}, \mathtt{p} + 3, \mathtt{c} \rangle, \langle \mathtt{ID2}, 0, \mathtt{c2} \rangle \mapsto \langle \mathtt{ID}, 0, \mathtt{c} \rangle, \langle \mathtt{ID2}, 0, \mathtt{c2} \rangle$$

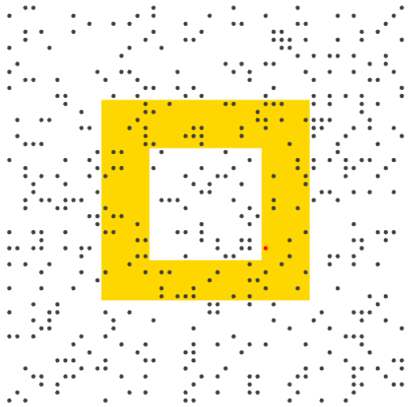
Hovering information

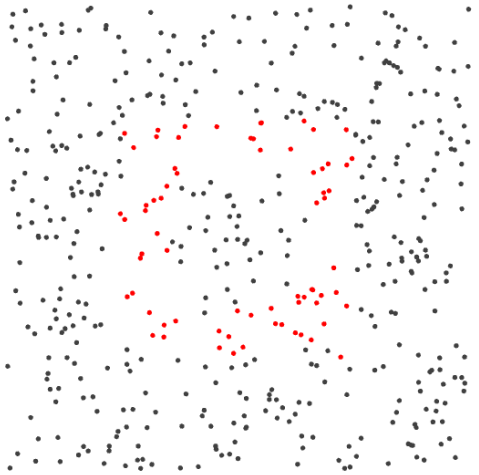
LSA:

$$\langle \texttt{ID}, \texttt{data} \rangle$$

Function:

$$\text{isInShape}(\text{position: p})$$





Hovering information

Eco-Laws:

- Broadcast

$$\langle \mathtt{pos} : \mathtt{isInShape}(\mathtt{pos}) \rangle, \langle \mathtt{ID}, \mathtt{data} \rangle \mapsto \langle \mathtt{pos} \rangle, \langle \mathtt{ID}, \mathtt{data} \rangle, bcast \langle \mathtt{ID}, \mathtt{data} \rangle$$

- Aggregate information

$$\langle \mathtt{ID}, \mathtt{data} \rangle, \langle \mathtt{ID}, \mathtt{data} \rangle \mapsto \langle \mathtt{ID}, \mathtt{data} \rangle$$

- Remove information

$$\langle \mathtt{pos} : \mathtt{not}(\mathtt{isInShape}(\mathtt{pos})) \rangle, \langle \mathtt{ID}, \mathtt{data} \rangle \mapsto \langle \mathtt{pos} \rangle$$

Summary

- SAPERE approach (2010-2013)
  - Apply self-* principles and context-awareness to pervasive scenarios
    - Crowd steering through public displays / Content sharing
  - Chemical reactions guide the behaviour of services
    - LSAs and Eco-laws
- Core set of bio-inspired design patterns
  - Expressed as generic eco-laws, triggered by matching LSAs
  - Supporting different other mechanisms and services at the same time
- Self-* algorithms developed with eco-laws

References

[Tchao11] Modeling Self-* Systems using Chemically-Inspired Composable Patterns. A. Tchao, M. Risoldi, G. Di Marzo Serugendo. SASO'11 Conference, October 2011

[Fernandez11a] **Description and Composition of Bio-Inspired Design Patterns: the Gossip Case**. JL Fernandez, J. Arcos, G. Di Marzo Serugendo, M. Casadei. EASE'11 Conference, June 2011

[Fernandez11b] **Description and Composition of Bio-Inspired Design Patterns: the Gradient Case.** JL Fernandez, J. Arcos, G. Di Marzo Serugendo, M. Viroli, S. Montagna. BADS'11 Workshop, June 2011

[Montagna11] **Self-organising Pervasive Ecosystems: A Crowd Evacuation Example**. S. Montagna, M. Viroli, M. Risoldi, D. Pianini, G. Di Marzo Serugendo. SERENE'11 Workshop, September 2011