SAPERE
Self-Aware Pervasive Service Ecosystems

Mirko Viroli
mirko.viroli@unibo.it

Alma Mater Studiorum—Università di Bologna, Cesena

Academic Year 2009/2010

What is SAPERE

A European Project

- European call for funding projects: FP7-ICT-2009.8.5
- Call name: Self-awareness in Autonomic Systems
- Period: 1/10/2010 - 30/9/2013
- Consortium: (i) Università di Modena e Reggio Emilia
  (ii) Università di Bologna
  (iii) University of London
  (iv) University of St Andrews, Dublin
  (v) Johannes Kepler Universitaet Linz

Impact on the Cesena site

- A "fulcrum" of research activities
- An opportunity for students (projects, thesis, post-laurea)

Outline/Goals of this seminar

- Deepen the "Future Pervasive Computing" scenario
- Describe the SAPERE Project
- Show what is the structure of an European Project
- Present early research results/ideas
- Propose thesis/projects

Outline

Future Pervasive Computing

SAPERE general aims

SAPERE structure

Early research ideas

Thesis

Bibliography

Challenges of Future Pervasive Computing

1 — Increasing introduction of pervasive devices

- The world will be more and more populated of computing devices
  - GPS, PDAs, smart phones, tags, cameras, displays, sensors, actuators
- Internet, Telecom, TV networks will become integrated
  - Think at Skype, Digital TV
- The network will become a true virtual counterpart of the entire world (events, sociality, business, physical structure)

Challenges of Future Pervasive Computing

2 — Prosumption of large masses of data and services

- Devices will produce large masses of data
- Users will themselves inject their data in the system [8]
  - Think at Facebook, Youtube
- We will expect that any environment properly react to our preferences/situation
- The distinction between data and services will fade
  - There is an increasing "long tail" in the market [1]

Challenges of Future Pervasive Computing

3 — Software will grow increasingly and be an "eternal beta"

- The standard notion of software is shifting
  - No longer a big monolithic engineering artifact
  - ..but rather a mash-up of services and data [5]
- similarly, the development cycle will become much different
  - No longer analysis-design-development-deployment
  - ..but rather it starts with initial services, and through years..
  - ..new simple services are injected
  - ..existing services are updated by small changes
  - ..even the availability of data and users will make the difference
- there is a feeling that a system is never in final state, but always in a "beta" state, that will improve next months
  - Open-source projects (even OS), social networks, ..

Challenges of Future Pervasive Computing

4 — Opennes, self-*, contex-awareness will be mandatory

- Opennes: we won't know which services, data, users, devices will be available soon, the infrastructure should work independently of this
- Self-* features should naturally emerge
  - Self-adaptiveness: tuning behaviour to ongoing changes
  - Self-organisation: find a better/new organisation (spatial/temporal)
  - Self-optimisation: be able to garbage services/data
  - Self-awareness: identifying situations
- Context-awareness: data and services will be relative to the position/location in which they reside
  - centralisation of data and software will soon be abandoned

The challenge of Pervasive Computing

A bad news for industry

- There is no clue on how (and whether) this can be addressed in its entirety
- Only specific solutions to specific problems so far (even by academia)

A good news for academia

A lot of work is going on in the following areas

- Research contexts: SOA, P2P, Grid, Cloud, Self-org, Coord
- There are *two next big things* in Information Technology
  - Pervasive Computing
  - Bio-ICT convergence (e.g. nature-inspired computing)

Outline

Future Pervasive Computing

SAPERE general aims

SAPERE structure

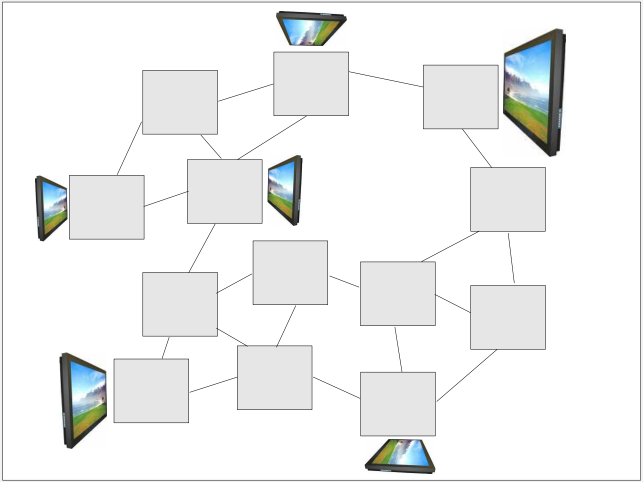
Early research ideas

Thesis

Bibliography

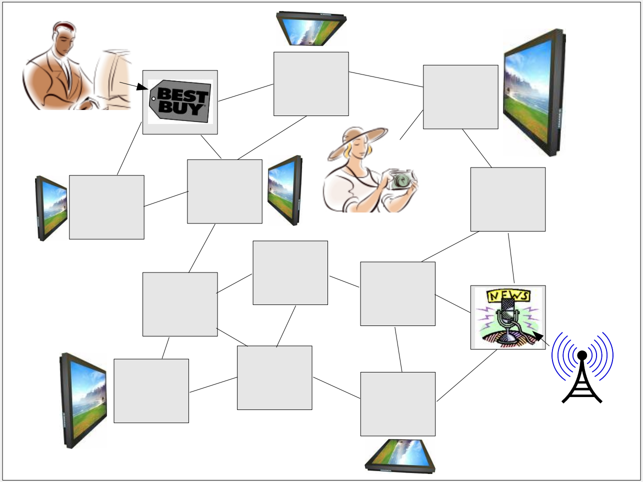
A scenario of pervasive displays

A pervasive network with displays spread around



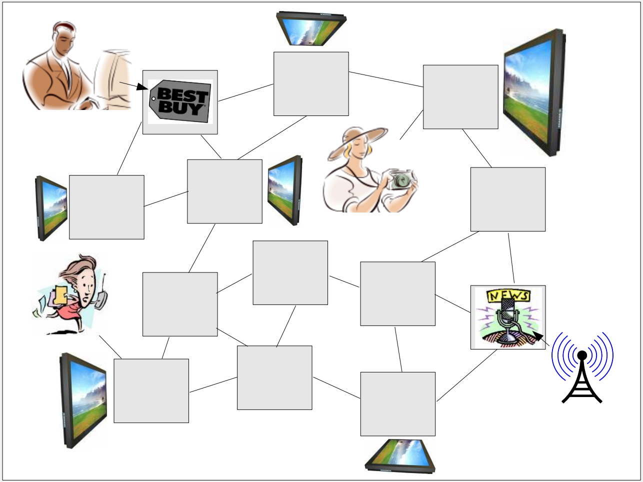
A scenario of pervasive displays

Visualisation services (news, ads, social data) get injected



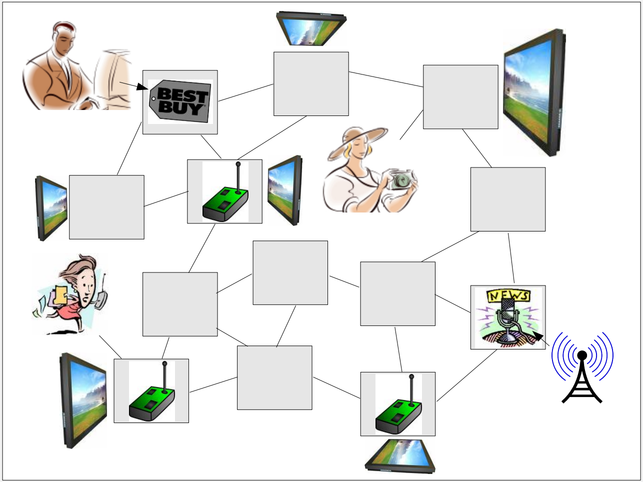
A scenario of pervasive displays

Displays should adapt visualisation to users nearby



A scenario of pervasive displays

Sensors provide contextual-information to improve adaptation



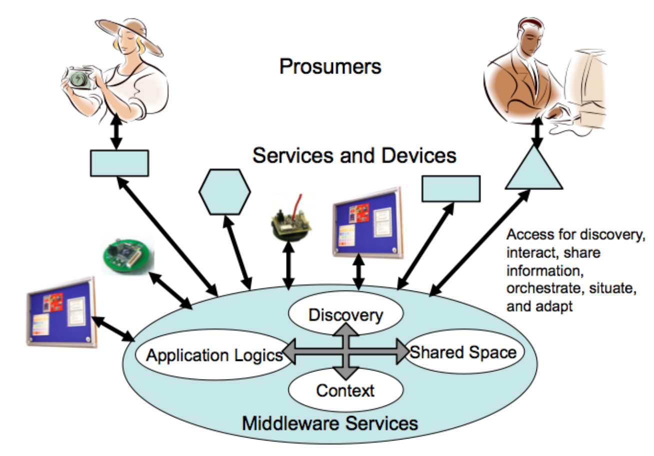
Some interesting services and features

- Displays show information based on majority of people around
- Alerts are shown as a given person passes nearby
- Displays coordinate to avoid irritating users
- Displays coordinate to provide visualisation streams
- Adjacent displays show a common, bigger content
- Injection of a new display cause redirection there
- A display used as a shared map
- Using eye-glasses with eye-pointers for immersed interaction

Abstract of SAPERE

The objective of SAPERE is the development of a highly-innovative theoretical and practical framework for the decentralized deployment and execution of self-aware and adaptive services for future and emerging pervasive network scenarios. The framework will be grounded on a foundational re-thinking of current service models and of associated infrastructures and algorithms. In particular, getting inspiration from natural ecosystems, the project will demonstrate and experiment the possibility of modelling and deploying services as autonomous individuals in an ecosystem of other services, data sources, and pervasive devices, and of enforcing self-awareness and autonomic behaviours as inherent properties of the ecosystem, rather than as peculiar characteristics of its individuals only.

A standard, centralised SOA solution



A standard, centralised SOA solution

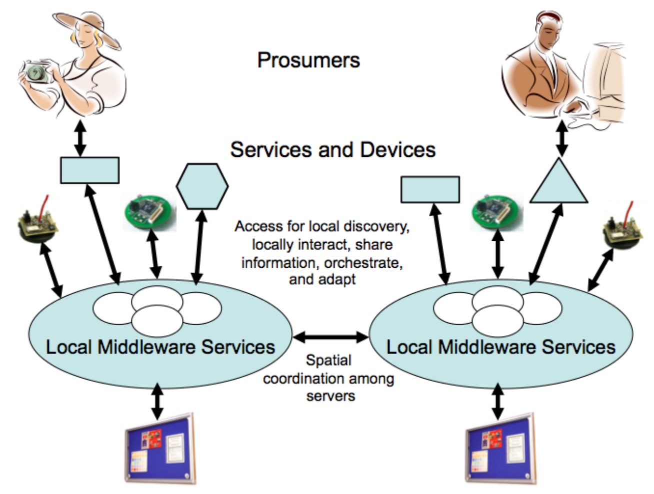
A centralised solution

One service for:

- Discovery: what components are available in the system?
- Context: where are components? (behaviour specialisation)
- Orchestration: coordinating components
- Shared space: depositing/retrieving local information
- Adaptation: reacting to contingencies

All components interact through such middleware services

De-centralising the SOA solution



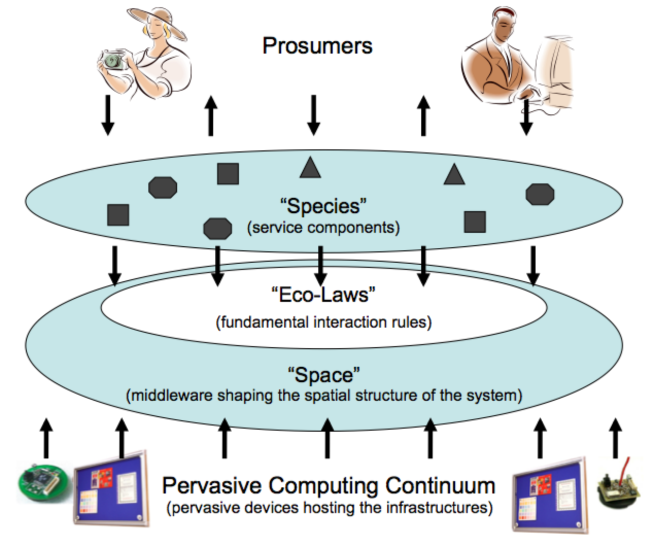
De-centralising the SOA solution

Duplicating middleware services in each location

Better deals with

- Better deals with situated components
- Simplifies contextualisation, discovery, and orchestration
- The role of shared spaces becomes more important
- Adaptation is still complex and crucial

Eco-inspired SOA solution



Eco-inspired SOA solution

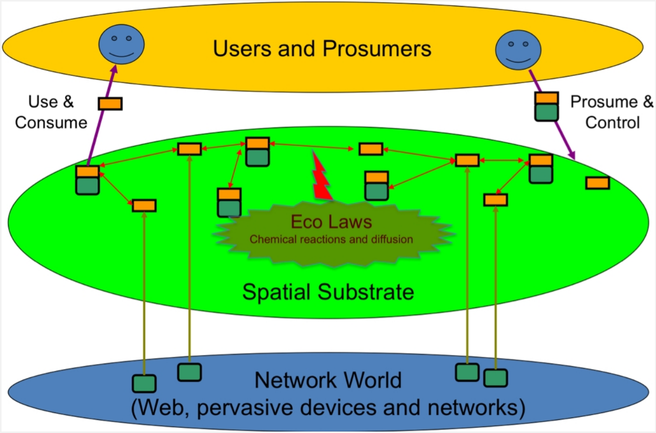
Fully decentralising middleware services

- Locations become very small and form a huge dynamic set
- Contextualisation, discovery, and orchestration almost vanish
- Midd. service just as a single space
- In overall we have a network of spaces with service “tags”
- Adaptation is achieved by simple rules combining tags

Drawing a bridge with natural ecosystems

We have a set of spatially situated entities interacting according to well-defined set of natural laws enforced by the spatial environment in which they situate, and adaptively self-organizing their interaction dynamics according to their shape and structure

Architecture



Pervasive Ecosystems

Example Patterns

Data/devices/services are added by injecting their tags, then:

- they could diminish until their population extinguish
- they compete with other "species", and may survive
- they compose with patches injected to improve them
- they diffuse around
- they move where the context is favourable
- they aggregate with other copies, forming an overlay

Eco-laws and Live Semantic Annotations

Live Semantic Annotations (LSA)

- A unified description for devices, data, services
- Is about interface, status, and behaviour of a component
- It provides semantic information, and it is dynamic

Eco-Laws

- They resemble chemical reactions
- They take some reagent LSA, and provide some product LSA
- They can diffuse an LSA in the neighborhood
- They can aggregate LSAs like in chemical bonding
- They form a small & fixed set of natural eco-laws

Outline

Future Pervasive Computing

SAPERE general aims

SAPERE structure

Early research ideas

Thesis

Bibliography

What is SAPERE

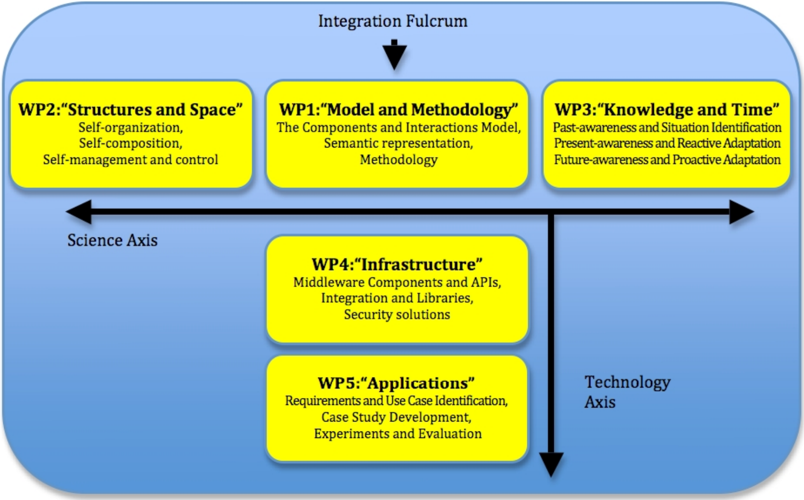
A European Project

- European call for funding projects: FP7-ICT-2009.8.5
- Call name: Self-awareness in Autonomic Systems
- Call type: Future and Emerging Technologies (FET)
- Funding Scheme: STREP (Specific Targeted Research Project)
- Period: 1/10/2010 - 30/9/2013
- Overall research grant: $\approx$ 2.5MEuro
- Consortium
    - Università di Modena e Reggio Emilia – Franco Zambonelli
    - Università di Bologna – Mirko Viroli
    - University of London – Giovanna di Marzo
    - University of St Andrews, Dublin – Simon Dobson
    - Johannes Kepler Universitaet Linz – Alois Ferscha

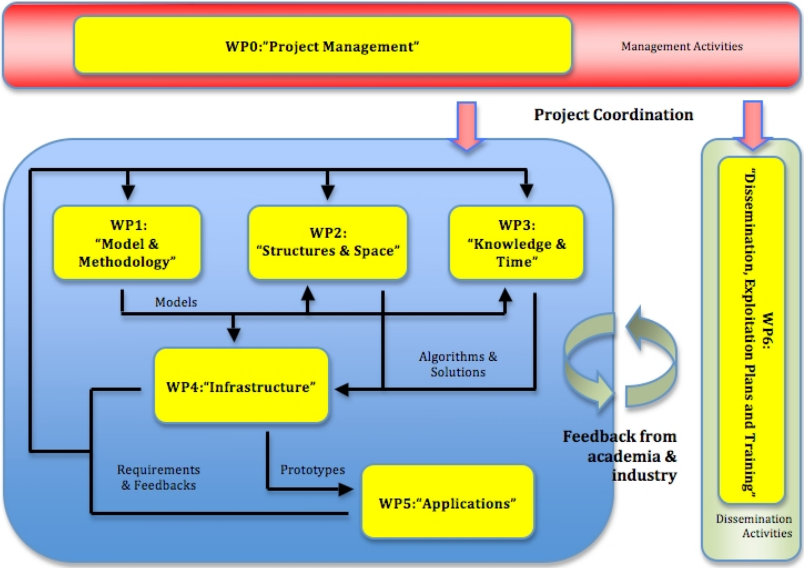
Elements of a Project Document (in general)

- Abstract
- Objectives
- Novelty and Contribution w.r.t. existing works
- Workplan (division in tasks, timing, efforts)
- Technical descriprions
- Deliverables
- Measure success indicators
- Dissemination activities
- Financial aspects

Strategy


Integration Fulcrum
WP2:"Structures and Space"
Self-organization,
Self-composition,
Self-management and control
WP1:"Model and Methodology"
The Components and Interactions Model,
Semantic representation,
Methodology
WP3:"Knowledge and Time"
Past-awareness and Situation Identification
Present-awareness and Reactive Adaptation
Future-awareness and Proactive Adaptation
Science Axis
WP4:"Infrastructure"
Middleware Components and APIs,
Integration and Libraries,
Security solutions
WP5:"Applications"
Requirements and Use Case Identification,
Case Study Development,
Experiments and Evaluation
Technology
Axis

Activities


WP0:"Project Management"
Management Activities
Project Coordination
WP1: "Model & Methodology"
WP2: "Structures & Space"
WP3: "Knowledge & Time"
Models
Algorithms & Solutions
WP4:"Infrastructure"
Requirements & Feedbacks
Prototypes
WP5:"Applications"
Feedback from academia & industry
WP6: "Dissemination, Exploitation Plans and Training"
Dissemination Activities

WP1: Model & Methodology – Leader UNIBO

Task T1.1 – The Components and Interaction Models

- Abstract (representation independent) model of eco-laws
- Services structure and interactions
- Studying analysis tools for behaviour verification

Task T1.2 – Semantic representation

- Shape of LSAs
- Studying analysis tools for logic reasoning

Task T1.3 – Methodology

- Finding a SE methodology
- Conceiving tools for development/analysis

WP1: Deliverables

D1.1 – Early Operational Model (M12) – Editor: UNIBO

- First version of abstract computational model
- First version of live semantic annotation framework
- Early demonstration of the operational model at work

D1.2 – Complete Operational Model (M20) – Editor: UNIBO

- ...

D1.3 – Final Operational and Semantic Model (M24) – Editor: UNIBO

- ...

D1.4 – Early Report on Methodology (M24) – Editor: STA

- ...

D1.5 – Methodology and Analysis Suite (M32) – Editor: UNIBO

- ...

D1.6 – Final Report on Methodology and Suite (M36) – Editor: UNIBO

- Complete and refined engineering methodology documentation
- Assessment of methodology and of associated tools.

Efforts

Partic. no.	Partic. short name	WP0	WP1	WP2	WP3	WP4	WP5	WP6	Total person months
1	UNIMORE	**15**	12	0	22	**34**	16	2	101
2	BIRKBECK	1	15	**36**	0	21	0	1	74
3	STA	1	13	8	**36**	0	15	1	74
4	UNIBO	1	**36**	20	0	20	0	4	81
5	JKU	1	0	16	16	0	**40**	2	75
Total		19	76	80	74	75	71	10	405

UNIBO details

- Leader of WP1: "Model and Methodology"
- Leader of WP6: "Dissemination, Exploitation, .."
- Involved in WP2: "Structures & Space"
- Involved in WP4: "Infrastructure"

Some details on the Ecosystem Model

Syntax / semantics of eco-laws

How is an eco-law specified?

1. Chemical template: $\texttt{X} + \texttt{Y} \xrightarrow{r} \texttt{Z}$
2. Matching functions: how LSAs (l_x and l_y) match X and Y?
3. Bond functions: are l_x and l_y compatible/complementary?
4. Generation function: how is l_z produced from l_x and l_y?
5. Rate: how l_x and l_y affect r, i.e., velocity/probability?

Bio-ICT convergence

This can be viewed as either:

- a semantic-oriented chemical model
- a semantic-oriented population dynamics model

Some details on Infrastructure

Seen as a tuple-space infrastructure

- Agents (devices/services/data) inject their LSA as a tuple
- Agents monitor changes in their LSA, which become actions
- Users perceive the ecosystem by oberving population of LSA
- The infrastructure provide a networked set of LSA-spaces
- LSAs evolve/diffuse by eco-inspired coordination laws

Possible implementation framework

- TuCSoN as a basic middleware
- ReSpecT (or equivalent) as language to express eco-laws
- Need a semantic module, and a chemical module

Outline

Future Pervasive Computing

SAPERE general aims

SAPERE structure

Early research ideas

Thesis

Bibliography

Chemical Tuple Spaces

Main idea

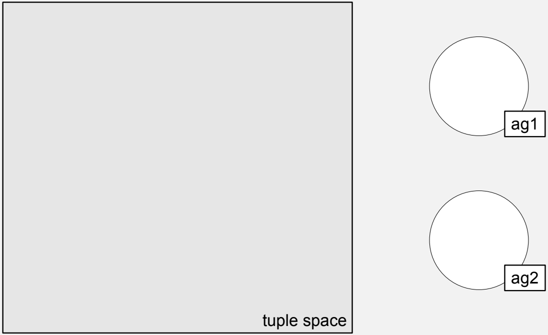
- Tuple spaces + chemical reactions as coordination laws
- Tuples have a concentration (a.k.a. weight, or activity value)
- Concentration is evolved "exactly" as in chemistry [4]
- Some reactions can even fire a tuple from one space to another

Why design coordination with biochemical metaphor?

- Chemistry fits coordination (Gamma)
- Can get inspiration from natural/artificial biochemistry
- Can model population evolution (prey-predator, [2])

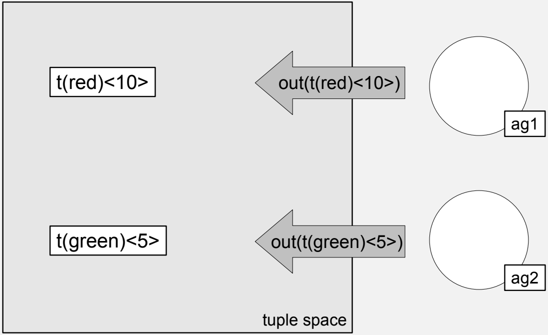
First settings

One tuple space, two agents



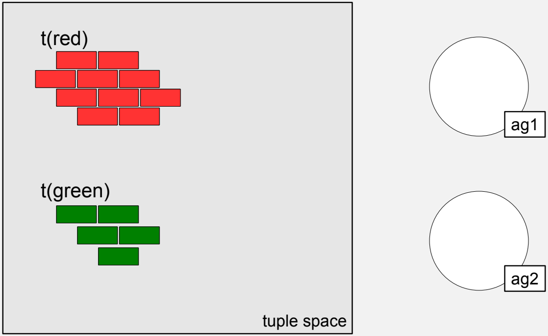
Inserting tuples

Primitive out: default concentration is 1



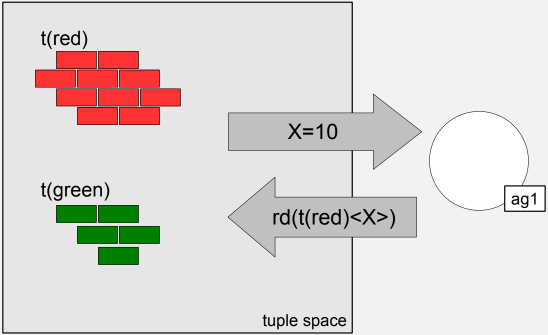
A pictorial representation

A tuple as substance of uniform molecules – but still a single tuple



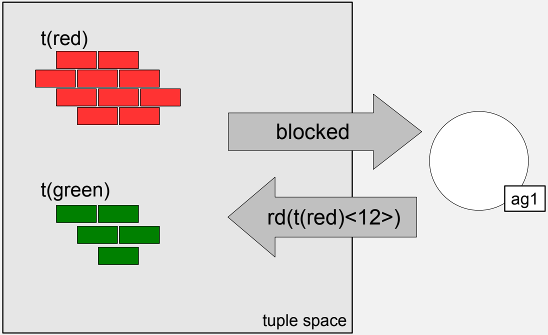
Reading Tuples

Primitive rd: reading current concentration



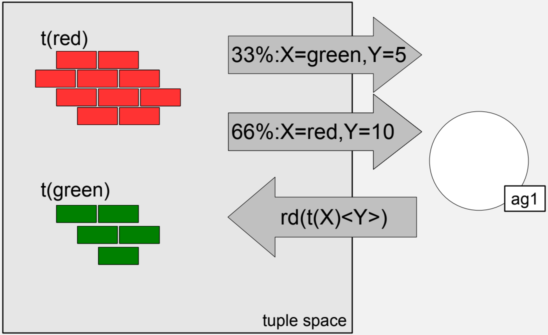
Reading Tuples

Primitive `rd`: reading a given amount – possibly blocking



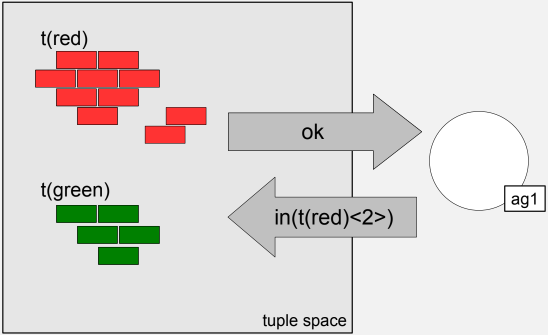
Reading Tuples

Primitive `rd`: concentration as probability, i.e., relevance



Removing Tuples

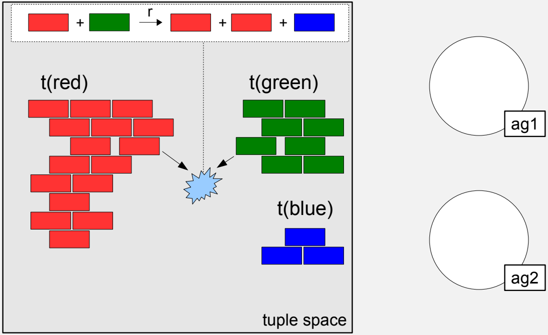
Primitive `in`: removing entirely or partially a tuple



Installing Chemical Reactions

A chemical reaction, with tuples in place of molecules

$$\mathtt{t(red)} + \mathtt{t(green)} \xrightarrow{r} \mathtt{t(red)} + \mathtt{t(red)} + \mathtt{t(blue)}$$

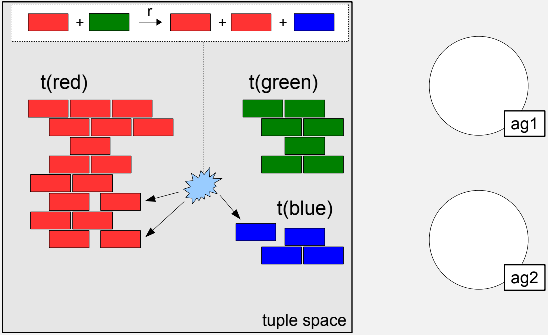


Firing Chemical Reactions

Reactions are executed over time according to [4]

$$\mathtt{t(red)} + \mathtt{t(green)} \xrightarrow{r} \mathtt{t(red)} + \mathtt{t(red)} + \mathtt{t(blue)}$$

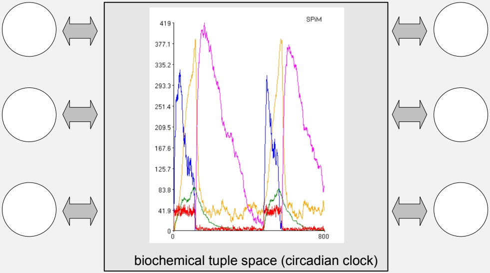
Transition (Markovian) rate: $r * \#\mathtt{t(red)} * \#\mathtt{t(green)}$



A tuple space as a chemical solution

Coordination through an exact chemical solution of tuples

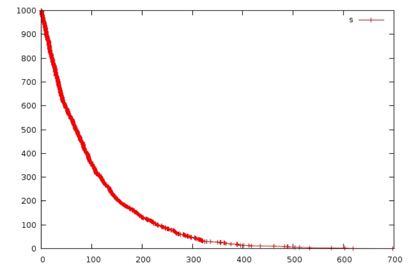
- The tuple space resembles a chemical solution in a glass
- Each tuple resembles a chemical substance
- Agents observe, insert and remove substances
- Tuple concentration drives the selection of chemical reactions



Decay example

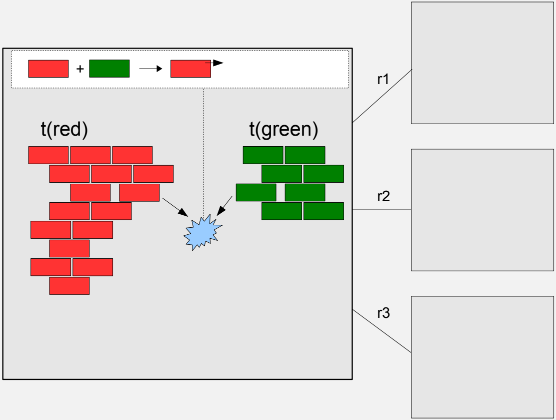
After installing reaction $\mathtt{t(X)} \xrightarrow{0.01} 0$

- We let tuples decade (evaporate like pheromones)
- This is useful to enact time-pertinency
- An agent perceives that the tuple is fading until disappearing
- E.g. `t(s)` represents the temporaneous publication of a service



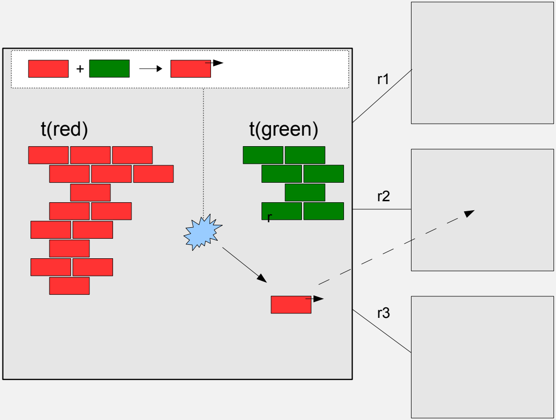
Tuple Transfer

Right-hand side of a reaction can have a firing tuple



From one node to a full biochemical network

Firing tuples are sent to any neighbour, probabilistically a là $S\pi$



On matching and rates

Overcoming discrete matching

- Matching is by substitution of variables, but it is ranked
- We use an application-dependent match function $\mu(t, t')$
  - $0 =$ no match, $1 =$ perfect match, $]0, 1[$ partial match
  - Chemical reactions are applied "modulo match ranking"
  - E.g. with $\mu = 0.5$, actual chemical rate is divided by 2
- A typical scenario of match-making with preferences

Example of general decay rule: $\mathtt{DECAY} \xrightarrow{r\_dec} 0$

- Tuple t decays with chemical rate $\mu(\mathtt{DECAY}, t) * r\_dec$
- E.g., t is service publication, granted after paying money
- μ inspects how much it was payed, tuning service life-time

Some implementation fact

Gillespie "direct" simulation algorithm [4]

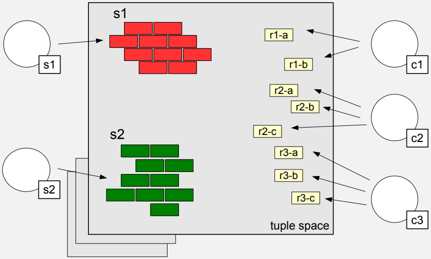
1. Compute the markovian rate $r_1, \ldots, r_n$ of reactions, let R be the sum
2. Choose one of them probabilistically, and execute its transition
3. Proceed again with (1) after $\frac{1}{R} * ln\frac{1}{\tau}$ seconds, with $\tau = random(0, 1)$

Tuple Space implementation

- Prototyped on top of TuCSoN
- Tuple centres programmed with the above algorithm

The scenario of service ecosystems

Services and requests as tuples



Clients and services as "individuals of an ecology"

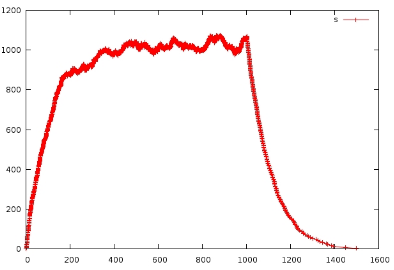
- Unused services fade until completely disappearing
- Concentration of a service increases upon usage
- Similar services compete for survival

Positive-Negative feedback

Service tuples decay, but can be sustained by a feedback token

- Decay rule: $\texttt{DECAY} \xrightarrow{r\_dec} 0$
- Feed rule: $\texttt{publish(SER)} \xrightarrow{r\_feed} \texttt{publish(SER)} + \texttt{SER}$

Example simulation: $r\_dec = 0.01, r\_feed = 10$



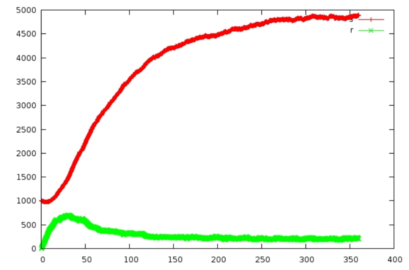
- time 0: Catalyst Token `publish(S)` is inserted
- time 400: Service `S` reaches an equilibrium
- time 1000: The token is removed (or decays)
- time 1600: Service `S` vanishes

Feedback by using (a.k.a. prey-predator)

Idea: Matching Service-Request sustains the service

- Use rule: $\mathtt{SER} + \mathtt{REQ} \xrightarrow{r\_use} \mathtt{SER} + \mathtt{SER} + \mathtt{toserve}(\mathtt{SER}, \mathtt{REQ})$

Sim: $r\_dec = 0.01$, $r\_use = 0.00005$, $request\_arrival\_rate = 50$



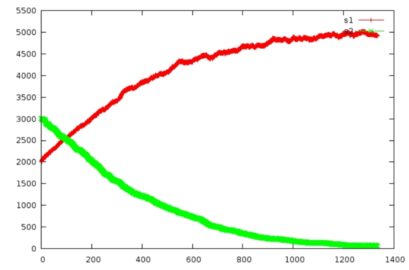
- time 0: Injection of requests raises service level
- time 30: Requests are tamed
- time 350: Unserved requests and service stabilise

Competition

What if more services can handle the same requests?

- higher concentration means higher match frequency
- some service may match better the request, being more proper

Sim: $r\_use_1 = 0.06, r\_use_2 = 0.04$



- time 0: The two services are in competition for the same requests
- time 100: The one with better use rate (better match) is prevailing
- time 1300: Service $s2$ lost competition and fades

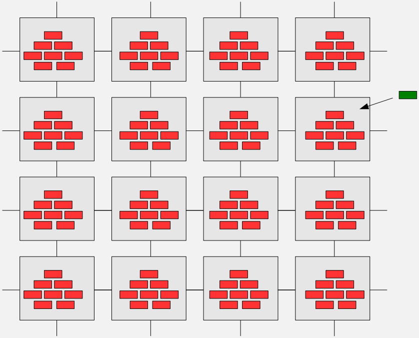
Spatial Diffusion and Competition

One service monopolises a network and its requests

Services continuously diffuse around, by rule:

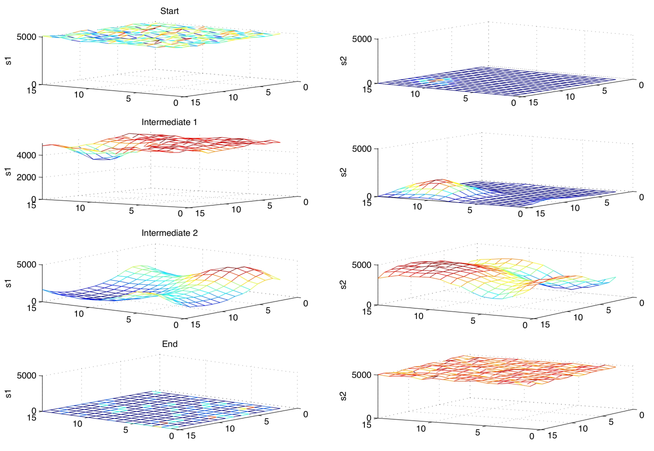
- Diffuse rule: $\mathtt{SER} \xrightarrow{r\_diff} \mathtt{SER}^{\rightsquigarrow}$

Scenario: a better service is injected in a node



Resembling a biological tissue scenario

Example Simulation: $r\_use_1 = 0.05, r\_use_2 = 0.1$



Discussion

Properties

The coordination space achieves the following:

- self-adaptation: the best service is selected over time
- self-optimisation: unused services get disposed
- opennes: can deal with incoming new services and requests

Outline

Future Pervasive Computing

SAPERE general aims

SAPERE structure

Early research ideas

Thesis

Bibliography

A list of available thesis/projects

Chemical behaviour

- Implementing a chemical-oriented tuple space, on top of TuCSoN or CArtAgO

Eco-laws

- Designing a language for expressing eco-laws
- Semantic reasoning (as in Description Logic) into tuple spaces

Simulation

- Working on existing simulation tools towards approximate model checking

Outline

Future Pervasive Computing

SAPERE general aims

SAPERE structure

Early research ideas

Thesis

Bibliography

Chris Anderson.
The Long Tail: Why the Future of Business is Selling Less of More.
Hyperion, 2006.

Alan A. Berryman.
The origins and evolution of predator-prey theory.
Ecology, 73(5):1530–1535, October 1992.

Gabriella Castelli, Alberto Rosi, Marco Mamei, and Franco Zambonelli.
A simple model and infrastructure for context-aware browsing of the world.
Pervasive Computing and Communications, 2007., pages 229–238, 19-23 March 2007.

Daniel T. Gillespie.
Exact stochastic simulation of coupled chemical reactions.
The Journal of Physical Chemistry, 81(25):2340–2361, 1977.

Mehdi Jazayeri.
Species evolve, individuals age.
In *IWPSE '05: Proceedings of the Eighth International Workshop on Principles of Software Evolution*, pages 3–12, Washington, DC, USA, 2005. IEEE Computer Society.

Gheorghe Paun.
Membrane Computing: An Introduction.
Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.

Andrew Phillips.
The Stochastic Pi Machine (SPiM), 2006.
Version 0.042 available online at http://www.doc.ic.ac.uk/~anp/spim/.

Raghu Ramakrishnan and Andrew Tomkins.
Toward a peopleweb.
Computer, 40(8):63–72, 2007.

University of Birmingham.
The PRISM probabilistic model checker.
Version 3.1.1 and documentation available online at
http://www.prismmodelchecker.org, September 2007.