Nature-inspired Coordination
Current Status and Research Trends

Andrea Omicini [•]
andrea.omicini@unibo.it

Stefano Mariani [•]
stefano.mariani@unimore.it

[•] Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

[•] Dipartimento di Scienze e Metodi dell'Ingegneria (DISMI)
Università di Modena e Reggio Emilia

Leipzig, Germany
23 August 2017

Abstract

Originating from closed parallel systems, coordination models and technologies gained in expressive power so as to deal with complex distributed systems. In particular, nature-inspired models of coordination emerged in the last decade as the most effective approaches to tackle the complexity of pervasive, intelligent, and self-* systems.
In the first part of the tutorial we introduce the basic notions of coordination and coordination model, and relate them to the notions of interaction and complexity. Then, the most relevant nature-inspired coordination (NIC) models are discussed, along with their relationship with the many facets of tuple-based models. In the third part we discuss the main open issues and explore the trends for future development of NIC. Finally, as a case study, we focus on MoK (Molecules of Knowledge), a NIC model for knowledge self-organisation, where data and information autonomously aggregate and spread toward knowledge prosumers.

Part I: Interaction, Complexity, Coordination

Part II: Nature-inspired Models of Coordination

Part III: Research Trends in NIC Models

Part IV: Molecules of Knowledge

Part I

Interaction, Complexity, Coordination

Next in Line...

Scenarios for Nowadays Computational Systems

Issues

concurrency / parallelism
- *multiple* independent activities / loci of control
- active *simultaneously*
- processes, threads, actors, active objects, agents...

distribution
- activities running on different and heterogeneous execution *contexts* (machines, devices, ...)

social interaction
- *dependencies* among activities
- collective *goals* involving activities coordination / cooperation

environmental interaction
- interaction with external *resources*
- interaction within the *time-space* fabric

Complexity & Interaction I

An essential source of complexity for computational systems is

interaction

[Goldin et al., 2006]

The power of interaction [Wegner, 1997]

Interaction is a more powerful paradigm than rule-based algorithms for computer-based solving, overtiring the prevailing view that all computing is expressible as algorithms.

Complexity & Interaction II

Interactive computing [Wegner and Goldin, 1999]

- finite computing agents that interact with an environment are shown to be more expressive than Turing machines according to a notion of expressiveness that measures problem-solving ability and is specified by observational equivalence
- sequential interactive models of objects, agents, and embedded systems are shown to be more expressive than algorithms
- *multi-agent* (distributed) models of coordination, collaboration, and true concurrency are shown to be more expressive than sequential models

Which Sorts of Components?

Open systems

- no hypothesis on the component's life & behaviour

Distributed systems

- no hypothesis on the component's location & motion

Heterogeneous systems

- no hypothesis on the component's nature & structure

(Non) Algorithmic Computation I

Elaboration / computation

- Turing Machine (TM) [Turing, 1937, Wegner and Goldin, 2003]
    - gets an input, elaborates it, throws an output
    - no interaction during computation
- black-box algorithms
- Church's Thesis and computable functions
    - in short, a function is *algorithmically computable iff* can be computed by a TM
    - so, all computable functions are computable by a TM

(Non) Algorithmic Computation II

The power of interaction [Wegner and Goldin, 2003]

real computational systems are not rational agents that take inputs, compute logically, and produce outputs... It is hard to draw the line at what is intelligence and what is environmental interaction. In a sense, it does not really matter which is which, as all intelligent systems must be situated in some world or other if they are to be useful entities.

[Brooks, 1991]

... a theory of concurrency and interaction requires a new conceptual framework, not just a refinement of what we find natural for sequential [algorithmic] computing.

[Milner, 1993]

(Non) Algorithmic Computation III

Beyond Turing Machines

- Turing's *choice machines* and *unorganised machines* [Wegner and Goldin, 2003]
- Wegner's Interaction Machines [Goldin et al., 2006]
- examples: AGV, Chess oracle [Wegner, 1997]

Basics of Interaction I

Component model

computation inner behaviour of a component

interaction observable behaviour of a component as *input* and *output*

Crossing component boundaries

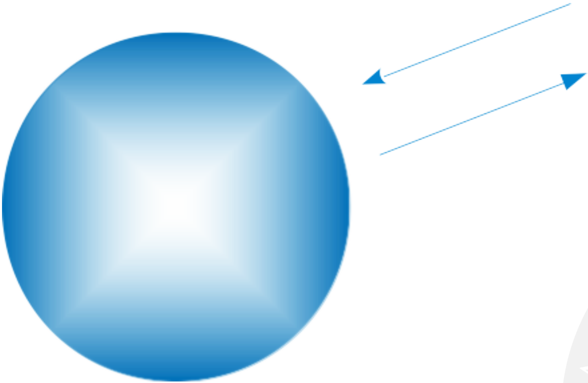
- control?
- information
- time & space—internal / computational vs. external / physical

Component interaction

output shows part of its state outside

input bounds a portion of its own state to the outside

Basics of Interaction II



Component of an interactive system with *input* and *output*

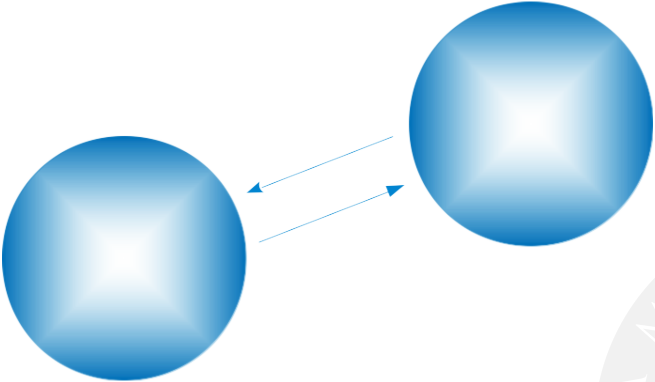
(Interacting) Computational System [Goldin et al., 2006] I

Computational system

In a computational system, two or more components

- *behave* (by computing), and
- *work together* (by interacting)

(Interacting) Computational System [Goldin et al., 2006] II



Basic interacting computational system

Basic Engineering Principles for Interactive Systems I

Principles

abstraction
- problems should be faced / represented at the most suitable *level of abstraction*
- resulting abstractions should be *expressive* enough to capture the most relevant problems
- *conceptual integrity*

locality & encapsulation
- design abstractions should embody the solutions corresponding to the domain entities they represent

run-time vs. design-time abstractions
- incremental change / evolution
- on-line engineering [Fredriksson and Gustavsson, 2004]
- (cognitive) self-organising systems [Omicini, 2012]

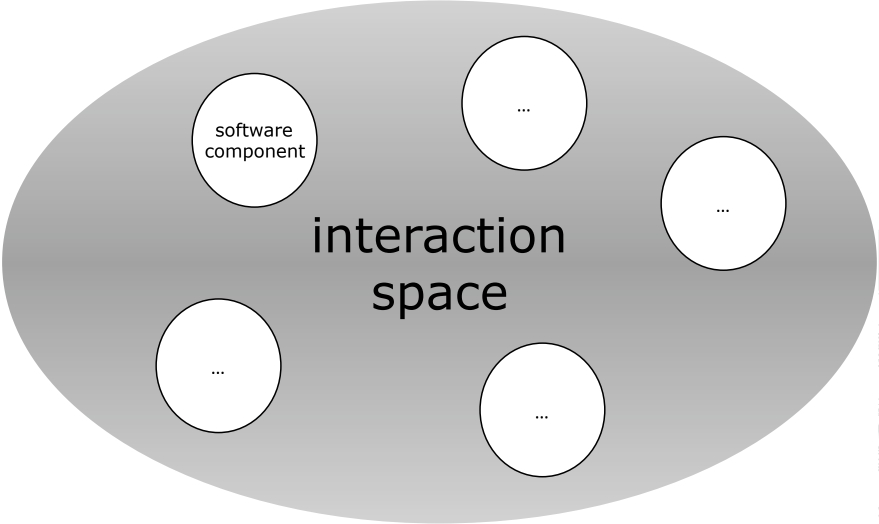
Basic Engineering Principles for Interactive Systems II

Issues

- what is the most suitable abstraction level to deal with interaction?
- which sort of dedicated SE abstractions could encapsulate the logic of the management of interaction?
- which sort of general conceptual framework could generally deal with all interaction-related issues?
- is there any general-purpose middleware technology that could reify such a framework?

Next in Line. . .

Interacting System



Coordination in Distributed Programming I

Coordination model as a glue

A coordination model is the glue that binds separate activities into an ensemble

[Gelernter and Carriero, 1992]

Coordination model as an agent interaction framework

A coordination model provides a framework in which the interaction of active and independent entities called agents can be expressed

[Ciancarini, 1996]

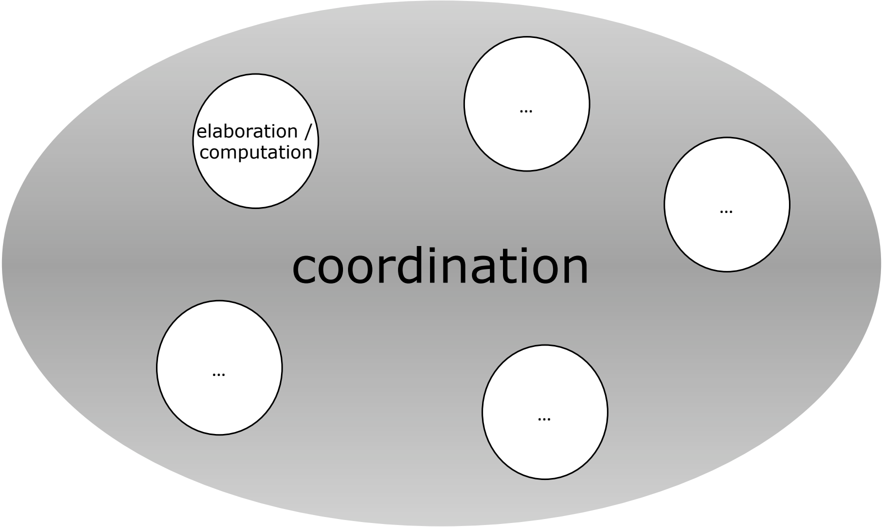
Coordination in Distributed Programming II

Issues for a coordination model

A coordination model should cover the issues of creation and destruction of agents, communication among agents, and spatial distribution of agents, as well as synchronization and distribution of their actions over time

[Ciancarini, 1996]

What is Coordination?



A New Perspective over Computational Systems

Programming languages

- interaction as an *orthogonal* dimension [Wegner, 1997]
- *languages* for interaction / *coordination* [Gelernter and Carriero, 1992]

Software engineering

- interaction as an *independent design* dimension [Ciancarini et al., 2000]
- coordination *patterns* [Deugo et al., 2001]

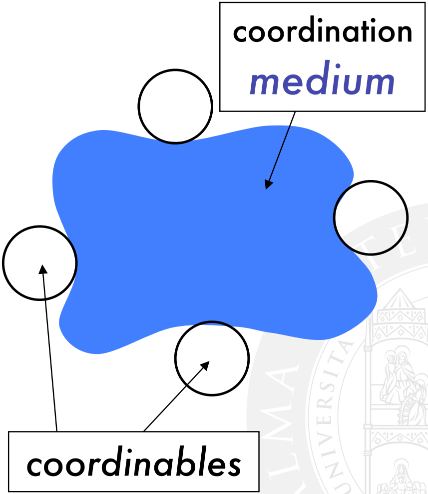
Artificial intelligence

- interaction as a new *source for intelligence* [Omicini and Papadopoulos, 2001]
- *social intelligence*—e.g., BIC [Castelfranchi et al., 2010]

Coordination: Sketching a Meta-model

The *medium of coordination*

- "fills" the interaction space
- enables / promotes / governs the admissible / desirable / required interactions among the interacting entities
- according to some *coordination laws*
    - enacted by the behaviour of the medium
    - defining the semantics of coordination



Coordination: A Meta-model [Ciancarini, 1996]

A constructive approach

Which are the components of a coordination system?

coordination entities entities whose mutual interaction is ruled by the model, also called the *coordinables*

coordination media abstractions enabling and ruling interaction among coordinables

coordination laws laws ruling the observable behaviour of coordination media and coordinables, and their interaction as well

Coordinables

Original definition [Ciancarini, 1996]

These are the entity types that are coordinated. These could be Unix-like processes, threads, concurrent objects and the like, and even users.

examples processes, threads, objects, human users, agents, . . .

focus observable behaviour of the coordinables

question are we anyhow concerned here with the internal machinery / functioning of the coordinable, in principle?

Coordination Media

Original definition [Ciancarini, 1996]

These are the media making communication among the agents possible. Moreover, a coordination medium can serve to aggregate agents that should be manipulated as a whole. Examples are classic media such as semaphores, monitors, or channels, or more complex media such as tuple spaces, blackboards, pipelines, and the like.

examples
: semaphors, monitors, channels, tuple spaces, blackboards, pipes, . . .

focus
: the core around which the components of the system are organised

question
: which are the possible computational models for coordination media?

Coordination Laws I

Original definition [Ciancarini, 1996]

A coordination model should dictate a number of laws to describe how agents coordinate themselves through the given coordination media and using a number of coordination primitives. Examples are laws that enact either synchronous or asynchronous behaviors or exploit explicit or implicit naming schemes for coordination entities.

Coordination Laws II

- coordination laws rule the observable behaviour of coordination media and coordinables, as well as their interaction
  - a notion of (*admissible interaction*) *event* is required to define coordination laws
- the interaction events are (also) expressed in terms of
  - the communication language, as the syntax used to express and exchange data structures

  examples tuples, XML elements, FOL terms, (Java) objects, ...

  - the coordination language, as the set of the admissible interaction primitives, along with their semantics

  examples in/out/rd (LINDA), send/receive (channels), push/pull (pipes), ...

Coordination as a Multi-Agent System Issue I

Agent-based engineering for complex software systems

- *encapsulation of control* makes agents overcome the abstraction gap of OOP [Odell, 2002]
- *agent* abstractions inherently deal with *distribution* [Jennings, 2000]
- agent *autonomy* — along with the associated agent features such as sociality, pro-activeness, situatedness, and the like – make agent-oriented software engineering (AOSE) a viable approach for engineering complex software systems [Jennings, 2001]
- extra agent features such as *mobility* and *intelligence* make AOSE the most effective approach available for the engineering of intelligent and pervasive distributed systems [Zambonelli and Omicini, 2004]

Coordination as a Multi-Agent System Issue II

A MAS meta-model [Mariani and Omicini, 2014b]

- activities are the *goal-directed/oriented* proceedings resulting into *actions* of any sort, which "make things happen" in a MAS
  - through actions, activities in a MAS are *social* [Castelfranchi, 1998] and *situated* [Suchman, 1987]
  - in MAS, activities are modelled through the *agent* abstraction
- environment change represents the (possibly unpredictable) variations in MAS environment
  - environment is usually modelled through the *resource* abstraction, as a non-goal-driven entity producing events and/or reactively waiting for requests to perform its function
- since activities *depend* on other activities (*social dependencies*), as well as on environment change (*situated dependencies*)

→ dependencies motivate and cause *interaction*, both social and situated, based on the sort of dependency taking place

Coordination in MAS

Given that

- artefacts are the *reactive* abstractions in MAS [Omicini et al., 2008]
- *coordination artefacts* are the social abstractions in charge of managing dependencies [Malone and Crowston, 1994]

the social abstractions in charge of coordinating multiple event flow according to their mutual dependencies in MAS are the *coordination artefacts* [Omicini et al., 2004]

Coordination models in MAS

The role of coordination models in MAS [Ciancarini et al., 2000] is to provide event-driven *coordination media* as the *coordination artefacts* governing event coordination in MAS

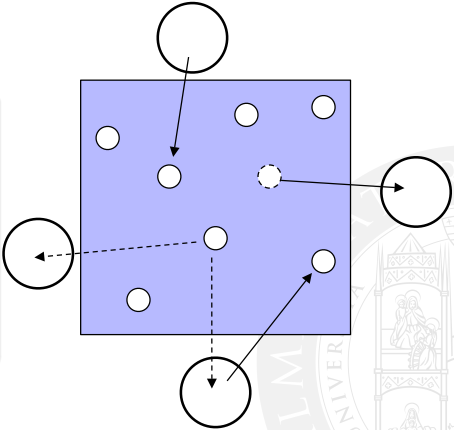
Next in Line. . .

The Tuple-space Meta-model

The most relevant class of coordination models is represented by tuple-based / space-based coordination models [Rossi et al., 2001]

The basics

- *coordinables* synchronise, cooperate, compete
  - based on *tuples*
  - available in the *tuple space*
  - by *associatively* accessing, consuming and producing tuples



Tuple-based / Space-based Coordination Systems

Adopting the constructive coordination meta-model [Ciancarini, 1996]

coordination media *tuple spaces*

- as multisets/bags of data objects/structures called *tuples*

communication language *tuples*

- as ordered collections of (possibly heterogeneous) information items

coordination language *tuple space primitives*

- as a set of operations to put, browse and retrieve tuples to/from the space

Linda: The Communication Language [Gelernter, 1985]

Communication Language

tuples ordered collections of possibly heterogeneous information chunks

- examples: p(1), printer('HP',dpi(300)), [0,0.5], matrix(m0,3,3,0.5), tree_node(node00,value(13),left(_),right(node01)), ...

templates / anti-tuples specifications of set / classes of tuples

- examples: p(X), [?int,?int], tree_node(N), ...

tuple matching mechanism the mechanism that matches tuples and templates

- examples: pattern matching, *unification*, ...

Linda: The Coordination Language [Gelernter, 1985] I

out(T)

- out(T) puts tuple T into the tuple space

  examples `out(p(1))`, `out(0,0.5)`, `out(course('Antonio Natali','Poetry',hours(150))` ...

Linda: The Coordination Language [Gelernter, 1985] II

in(TT)

- `in(TT)` retrieves a tuple matching template `TT` from to the tuple space

  destructive reading the tuple retrieved is removed from the tuple centre

  non-determinism if more than one tuple matches the template, one is chosen non-deterministically

  suspensive semantics if no matching tuples are found in the tuple space, operation execution is suspended, and woken when a matching tuple is finally found

  examples `in(p(X))`, `in(0,0.5)`, `in(course('Antonio Natali',Title,hours(X))` ...

Linda: The Coordination Language [Gelernter, 1985] III

rd(TT)

- `rd(TT)` retrieves a tuple matching template `TT` from to the tuple space

  non-destructive reading: the tuple retrieved is left untouched in the tuple centre

  non-determinism: if more than one tuple matches the template, one is chosen non-deterministically

  suspensive semantics: if no matching tuples are found in the tuple space, operation execution is suspended, and awakened when a matching tuple is finally found

  examples: `rd(p(X))`, `rd(0,0.5)`, `rd(course('Alessandro Ricci','Operating Systems',hours(X))` ...

LINDA Extensions: Predicative Primitives

`inp(TT)`, `rdp(TT)`

- both `inp(TT)` and `rdp(TT)` retrieve tuple `T` matching template `TT` from the tuple space

$=$ `in(TT)`, `rd(TT)` (non-)destructive reading, non-determinism, and syntax structure is maintained

$\neq$ `in(TT)`, `rd(TT)` suspensive semantics is lost: this *predicative* versions primitives just fail when no tuple matching `TT` is found in the tuple space

success / failure predicative primitives introduce *success / failure semantics*: when a matching tuple is found, it is returned with a success result; when it is not, a failure is reported

LINDA Extensions: Bulk Primitives I

`in_all(TT), rd_all(TT)`

- LINDA primitives deal with one tuple at a time
  - some coordination problems require more than one tuple to be handled by a single primitive
- `rd_all(TT)`, `in_all(TT)` get all tuples in the tuple space matching with `TT`, and returns them all
  - no suspensive semantics: if no matching tuple is found, an empty collection is returned
  - no success / failure semantics: a collection of tuple is always successfully returned—possibly, an empty one
  - in case of logic-based primitives / tuples, the form of the primitive are `rd_all(TT,LT)`, `in_all(TT,LT)` (or equivalent), where the (possibly empty) list of tuples unifying with `TT` is unified with `LT`
  - (non-)destructive reading: `in_all(TT)` consumes all matching tuples in the tuple space; `rd_all(TT)` leaves the tuple space untouched

LINDA Extensions: Bulk Primitives II

Other bulk primitives

- many other bulk primitives have been proposed and implemented to address particular classes of problems
- most of them too specific to be considered as a general extension to LINDA, and for inclusion in tuple-based models in general

LINDA Extensions: Multiple Tuple Spaces

`ts ? out(T)`

- LINDA tuple space might be a bottleneck for coordination
- many extensions have focussed on making a multiplicity of tuple spaces available to processes
  - each of them encapsulating a portion of the coordination load
  - either hosted by a single machine, or distributed across the network
- syntax required, and dependent on particular models and implementations
  - a space for tuple space names, possibly including network location
  - operators to associate LINDA operators to tuple spaces
- for instance, `ts @ node ? out(p)` may denote the invocation of operation `out(p)` over tuple space `ts` on node `node`

Main Features of Tuple-based Coordination

Main features of the LINDA model

tuples a tuple is an ordered collection of knowledge chunks, possibly heterogeneous in sort

generative communication until explicitly withdrawn, the tuples generated by coordinables have an independent existence in the tuple space; a tuple is equally accessible to all the coordinables, but is bound to none

associative access tuples in the tuple space are accessed through their content & structure, rather than by name, address, or location

suspensive semantics operations may be suspended based on unavailability of matching tuples, and be woken up when such tuples become available

Features of Linda: Tuples

tuple an ordered collection of knowledge chunks, possibly heterogeneous in sort

- a record-like structure
- with no need of field names
- easy aggregation of knowledge
- raw semantic interpretation: a tuple contains all information concerning a given item

tuple structure based on

- arity
- type
- position
- information content

tuple templates / anti-tuples

- to describe / define sets of tuples

matching mechanism

- to define belongingness to a set

Features of Linda: Generative Communication

Communication orthogonality

- both senders and the receivers can interact even without having prior knowledge about each others

space uncoupling
: no need to coexist in space for two processes to interact

time uncoupling
: no need for simultaneity for two processes to interact

name uncoupling
: no need for names for processes to interact

Features of Linda: Associative Access

Content-based coordination

synchronisation based on tuple *content* & *structure*

- absence / presence of tuples with some content / structure determines the overall behaviour of the coordinables, and of the coordinated system in the overall
- based on tuple templates & matching mechanism

information-driven coordination

- patterns of coordination based on data / information availability
- based on tuple templates & matching mechanism

reification

- making events become tuples
- grouping classes of events with tuple syntax, and accessing them via tuple templates

Features of Linda: Suspensive Semantics

Blocking primitives

- `in` & `rd` primitives in LINDA have a suspensive semantics
  - the coordination medium makes the primitives waiting in case a matching tuple is not found, and wakes it up when such a tuple is found
  - the coordinable invoking the suspensive primitive is expected to wait for its successful completion
- twofold wait

  in the coordination medium the operation is first (possibly) suspended, then (possibly) served: coordination based on absence / presence of tuples belonging to a given set

  in the coordination entity the invocation may cause a wait-state in the invoker: hypothesis on the internal behaviour of the coordinable

Data- vs. Control-driven Coordination

- what if we need to start an activity after, say, at least N processes have asked for a resource?
    - more generally, what if we need, in general, to coordinate based on the coordinable actions, rather than on the information available / exchanged?
- classical distinction in the coordination community
    - *data-driven* coordination vs. *control-driven* coordination
- in more advanced scenario, these names do not fit
    - information-driven coordination vs. action-driven coordination fits better
    - but we might as well use the old terms, while we understand their limitations

Hybrid Coordination Models

- generally speaking, control-driven coordination does not fit so well information-driven contexts, like Web-based ones, for instance
  - control-driven models like Reo [Arbab, 2004] need to be adapted to contexts like agent-based ones, mainly to deal with the issue of autonomy in distributed systems [Dastani et al., 2005]
  - control should not pass through the component boundaries in order to avoid coupling in distributed systems
- features of both approaches to coordination are required
  - *hybrid* coordination models [Omicini, 2000]
  - adding for instance a control-driven layer to a LINDA-based one
- examples
  - LGI/LGL [Minsky et al., 2001]
  - TOTA [Mamei and Zambonelli, 2004]
  - TuCSoN [Omicini and Zambonelli, 1999]

Next in Line...

Lessons Learnt

- *interaction* as the source of complexity
- focus on open, distributed, heterogeneous systems
- *coordination* as the discipline of governing interaction
- *tuple-based models* as the most relevant / expressive ones

Part II

Nature-inspired Models of Coordination

Next in Line...

Why Nature-inspired Models?

Complex natural systems

- such as physical, chemical, biochemical, biological, social systems
- natural system exhibit *features*
  - such as distribution, opennes, situation, fault tolerance, robustness, adaptiveness, ...
- which we would like to understand, capture, then bring to *computational* systems

Nature-inspired computing (NIC)

- for instance, NIC [Liu and Tsui, 2006] summarises decades of research activities, putting emphasis on
  - autonomy of components
  - self-organisation of systems

Why Coordination Models?

Interaction

- most of the complexity of computational systems comes from interaction [Omicini et al., 2006]
- along with an essential part of their expressive power [Wegner, 1997]

Coordination

- since coordination is essentially the science of managing the space of interaction [Wegner, 1997]
- coordination models and languages [Ciancarini, 1996] provide abstractions and technologies for the engineering of complex computational systems [Ciancarini et al., 2000]

Why Nature-inspired Coordination?

Coordination issues in natural systems

- coordination issues did not first emerge in computational systems
- [Grassé, 1959] noted that in termite societies *"The coordination of tasks and the regulation of constructions are not directly dependent from the workers, but from constructions themselves."*

Coordination as the key issue

- many well-known examples of natural systems – and, more generally, of complex systems – seemingly rely on simple yet powerful coordination mechanisms for their key features—such as self-organisation
- it makes sense to focus on nature-inspired coordination models as the core of complex nature-inspired computational systems

Next in Line...

Focus on. . .

Stigmergy I

Stigmergy in insect societies

- nature-inspired models of coordination are grounded in studies on the behaviour of social insects, like ants or termites
- [Grassé, 1959] introduced the notion of stigmergy as the fundamental coordination mechanism in termite societies
- in ant colonies, pheromones act as environment markers for specific social activities, and drive both the *individual* and the *social* behaviour of ants

Stigmergy II

Stigmergy in computational systems

- nowadays, stigmergy generally refers to a set of nature-inspired coordination mechanisms mediated by the *environment*
- *digital pheromones* [Parunak et al., 2002] and other *signs* made and sensed in a shared environment [Parunak, 2006] can be exploited for the engineering of adaptive and self-organising computational systems

Chemical Coordination

Chemical reactions as (natural) coordination laws

- inspiration comes from the idea that complex physical phenomena are driven by the (relatively) simple chemical reactions
- coordinating the behaviours of a huge amount of components, as well as the global system evolution

Chemical reactions as (computational) coordination laws

- Gamma [Banâtre and Le Métayer, 1990] is a *chemistry-inspired coordination* model—as for the CHAM (chemical abstract machine) model [Berry, 1992]
- coordination in Gamma is conceived as the evolution of a space governed by chemical-like rules, globally working as a rewriting system [Banătre et al., 2001]

Focus on...

Field-based Coordination

Computational fields as coordination laws

- field-based coordination models like Co-fields [Mamei and Zambonelli, 2006] are inspired by the way masses and particles move and self-organise according to gravitational/electromagnetic fields
- there, computational force fields – generated either by the mobile agents or by the pervasive coordination infrastructure – propagate across the environment, and drive the actions and motion of the agent themselves

(Bio)chemical Coordination

Chemical reactions as coordination laws

- chemical tuple spaces [Viroli et al., 2010] exploit the chemical metaphor at its full extent—beyond Gamma
- data, devices, and software agents are represented in terms of chemical reactants, and system behaviour is expressed by means of chemical-like laws
- which are actually time-dependent and stochastic
- embedded within the coordination medium
- biochemical tuple spaces [Viroli and Casadei, 2009] add *compartments*, *diffusion*, and *stochastic behaviour* of coordination primitives

Aggregate Computing

Aggregates as programmable units [Beal et al., 2013]

Alternate approach to the standard device-centred development methodology, aimed at simplifying the design, creation, and maintenance of *large-scale* software systems [Beal et al., 2015]—i.e. IoT, cyber-physical systems, pervasive computing, robotic swarms.

- roots in computational fields [Mamei and Zambonelli, 2006], chemical coordination [Viroli et al., 2010], spatial computing [Viroli et al., 2011]
- the reference computing machine is no longer the single device but an aggregate *collection of devices*
- the details of behaviour, position, and number of devices are largely abstracted away to be replaced with a *space-filling computational environment*

Swarm Robotics

Swarms as coordinated ensambles

Swarm intelligence [Bonabeau et al., 1999] has a long tradition of models and algorithms drawing inspiration from ecological systems – most notably ant colonies, birds flocks, schools of fishes – to devise out efficient and fully decentralised cooperation/coordination mechanisms—mostly exploited in swarm robotics [Brambilla et al., 2013].

- exploiting the idea within tuple-based coordination models already tried, i.e. SwarmLinda [Tolksdorf and Menezes, 2004]
- many applications in the general area of swarm robotics [Şahin, 2005], i.e. cooperative transport [Kube and Bonabeau, 2000]

Focus on. . .

Basic Issues of Nature-inspired Coordination I

Environment

- environment is essential in nature-inspired coordination
  - it works as a mediator for agent interaction — through which agents can communicate and coordinate indirectly
  - it is active — featuring autonomous dynamics, and affecting agent coordination
  - it has a structure — requiring a notion of *locality*, and allowing agents of any sort to *move* through a topology

Basic Issues of Nature-inspired Coordination II

Stochastic behaviour

- complex systems typically require probabilistic models
  - *don't know* / *don't care* non-deterministic mechanisms are not expressive enough to capture all the properties of complex systems such as biochemical and social systems
  - probabilistic mechanisms are required to fully capture the dynamics of coordination in nature-inspired systems
  - coordination models should feature (possibly simple yet) expressive mechanisms to provide coordinated systems with stochastic behaviours

Next in Line...

The Ancestor

Linda [Gelernter, 1985]

- Linda is the ancestor of all tuple-based coordination models [Rossi et al., 2001]
- in Linda, agents synchronise, cooperate, compete
  - based on tuples
  - available in the tuple spaces, working as the coordination media
  - by *associatively* accessing, consuming and producing tuples
- the same holds for any tuple-based coordination model

LINDA is *not* a Nature-inspired Model

Warning

LINDA is *not* a Nature-inspired Model

So, *why* LINDA?

Why tuple-based models?

Why Tuple-based Models? I

Expressiveness

- LINDA is sort of a *core* coordination model
- making it easy to face and solve many typical problems of complex distributed systems
- *complex* coordination problems are solved with *few, simple* primitives
- whatever the model used to measure expressiveness of coordination, tuple-based languages are highly-expressive [Busi et al., 1998]

Why Tuple-based Models? II

Environment-based coordination

- generative communication [Gelernter, 1985] requires *permanent* coordination abstractions
- so, the *coordination infrastructure* provides agents with tuple spaces as coordination services
  - *coordination as a service* (CaaS) [Viroli and Omicini, 2006]
- they can be interpreted as coordination artefacts shaping computational *environment* [Omicini et al., 2004]
  - and used with different levels of awareness by both intelligent and "stupid" agents [Omicini, 2012]
- as such, they can be exploited to support environment-based coordination [Ricci et al., 2005]

Why Tuple-based Models? III

Extensibility

- whatever its expressiveness, LINDA was conceived as a coordination model for closed, parallel systems
- so, in fact, some relevant problems of today open, concurrent systems cannot be easily solved with LINDA either in practice or in theory
- as a result, tuple-based models have been extended with new simple yet powerful mechanisms
- generating a plethora of tuple-based coordination models [Rossi et al., 2001]

Why Tuple-based Models? IV

Nature-inspired extensions

- LINDA may *not* be nature-inspired, but many of its extensions *are*
- many of the coordination models depicted before
  - stigmergy [Parunak, 2006]
  - field-based [Mamei and Zambonelli, 2004]
  - chemical [Viroli et al., 2010] and biochemical [Viroli and Casadei, 2009]
- along with many others, such as
  - cognitive stigmergy [Ricci et al., 2007]
  - pervasive ecosystems [Viroli et al., 2012]
  - knowledge self-organisation [Mariani and Omicini, 2012a]
- are actually nature-inspired tuple-based coordination models

Next in Line...

Lessons Learnt

- natural systems have features we would like to have in computational ones — self-org, adaptiveness, robustness
- most of such features depend on correct management of basic interactions between many simple components
- stigmergy and chemical / field-based coordination as most solid examples, swarm robotics as a typical application scenario
- environment and stochastic dynamics as essential requirements
- tuple-based coordination lends itself well to nature-inspired extensions

Part III

Research Trends in NIC Models

Next in Line...

Complexity as a Multi-disciplinary Notion

Complex systems everywhere

- the notion of *complexity* is definitely a *multi-disciplinary* one, ranging from physics to biology, from economics to sociology and organisation sciences
- systems that are said *complex* are both natural and artificial ones

Natural vs. artificial complex systems

- we *observe* and *model* complex physical systems
- we *design* and *build* complex computational systems

Question

- which features do *all* complex systems share independently of their nature?

Complexity & Interaction

> *. . . by a complex system I mean one made up of a large number of parts that interact in a non simple way* [Simon, 1962]

Laws of complexity

- some "laws of complexity" exists that characterise any complex system, *independently* of its specific nature [Kauffman, 2003]
- the precise source of what all complex systems share is still unknown in essence

Interaction

- we argue that interaction – its *nature*, *structure*, *dynamics* – is the key to understand some fundamental properties of complex systems of any kind

Interaction in Statistical Mechanics I

Independence from interaction

- some physical systems are described under the assumption of mutual independence among particles—that is, the behaviour of the particles is unaffected by their mutual interaction
  - e.g., ideal gas [Boltzmann, 1964]
- there, the probability distribution of the whole system is the product of those of each of its particles
- in computer science terms, the *properties of the system* can be compositionally derived by the *properties of the individual components* [Wegner, 1997]

→ neither macroscopic sudden shift nor abrupt change for the system as a whole: technically, those systems have no phase transitions—of course, while the "independence from interaction" hypothesis holds

Interaction in Statistical Mechanics II

Interacting systems

- introducing interaction among particles structurally *changes* the *macroscopic properties*, along with the *mathematical* ones
- interacting systems are systems where particles *do not behave independently* of each other
- the probability distribution of an interacting system does not factorise anymore
- in computer science terms, an interacting system is *non-compositional* [Wegner, 1997]

Interaction in Statistical Mechanics III

Interacting vs. non-interacting systems

- only interacting systems can describe real cases beyond the idealised ones
  - e.g., they can explain phase transitions – like liquid-gas transition – and much more, such as collective emerging effects
- while a system made of independent parts can be represented by isolated single nodes, an *interacting system* is better described by *nodes connected by lines* or higher-dimensional objects
- from the point of view of information and communication theories, an ideal non-interacting gas is a system of *non-communicating nodes*, whereas an interacting system is made of *nodes connected by channels*

Complexity in Statistical Mechanics I

The case of magnetic particles

- the simplest standard prototype of an interacting system is the one made of magnetic particles
- there, individual particles can behave according to a magnetic field which leaves their probabilistic independence undisturbed
- at the same time, two magnetic particles interact with each other, and the strength of their interaction is a crucial tuning parameter to observe a phase transition
  - if interaction is weak, the effect of a magnetic field is smooth on the system
  - instead, if the interaction is strong – in particular, higher than a threshold – even a negligible magnetic field can cause a powerful *cooperative effect* on the system

Complexity in Statistical Mechanics II

Interaction is not enough

- interaction is a necessary ingredient for complexity in statistical mechanics but definitely not a sufficient one
- complexity arises when the possible equilibrium states of a system grow very quickly with the number of particles, regardless of the simplicity of the laws governing each particle and their mutual interaction
- roughly speaking, complexity is much more related to size in number, rather than to complexity of the laws ruling interaction

→ we do *not* need *complex interaction* to make interaction lead to complexity

From Statistical Mechanics to Social Systems I

Large numbers

- the key point in statistical mechanics is to relate the *macroscopic* observables quantities – like pressure, temperature, etc. – to suitable *averages* of *microscopic* observables—like particle speed, kinetic energy, etc.
- based on the *laws of large numbers*, the method works for those systems made of a large number of particles / basic components

From Statistical Mechanics to Social Systems II

Beyond the boundaries

- *methods for complex systems* from statistical mechanics have expanded from physics to fields as diverse as biology [Kauffman, 1993], economics [Bouchaud and Potters, 2003, Mantegna and Stanley, 1999], and computer science itself [Mézard and Montanari, 2009, Nishimori, 2001]
- recently, they have been applied to *social sciences* as well: there is evidence that the complex behaviour of many observed socio-economic systems can be approached with the *quantitative tools* from statistical mechanics
  - e.g., *Econophysics* for crisis events [Stanley, 2008]

From Statistical Mechanics to Social Systems III

Social systems as statistical mechanical systems

- a group of isolated individuals neither knowing nor communicating with each other is the typical example of a *compositional* social system
- no sudden shifts are expected in this case at the collective level, unless it is caused by strong external exogenous causes
- to obtain a *collective behaviour* displaying *endogenous* phenomena, the individual *agents* should meaningfully *interact* with each other
- the foremost issue here is that the nature of the interaction determines the nature of the collective behaviour at the aggregate level
  - e.g., a simple *imitative* interaction is capable to cause strong polarisation effects even in presence of extremely small external inputs

Modelling vs. Engineering

Physical vs. computational systems

- physical systems are to be observed, understood, and possibly *modelled*
  - → for physical systems, the laws of interaction, and their role for complexity, are to be *taken as given*, to be possibly formalised mathematically by physicists
- computational systems are to be *designed* and built
  - → for computational systems, the laws of interaction have first to be *defined* through amenable abstractions and computational models by computer scientists, then exploited by computer engineers in order to build systems

Coordinated Systems as Interacting Systems I

Coordination media for ruling interaction

- defining the abstractions for ruling the interaction space in computational systems basically means to define their *coordination model* [Gelernter and Carriero, 1992, Ciancarini, 1996, Ciancarini et al., 1999]
- *global properties* of complex coordinated systems depending on interaction can be enforced through the *coordination model*, essentially based on its expressiveness [Zavattaro, 1998, Denti et al., 1998]
    - for instance, tuple-based coordination models have been shown to be expressive enough to support self-organising coordination patterns for nature-inspired distributed systems [Omicini, 2013a]

Coordinated Systems as Interacting Systems II

The role of coordination models

Coordination models could be exploited

- to *rule* the *interaction space*
- so as to *define* new sorts of *global*, macroscopic *properties* for *computational systems*, possibly inspired by physical ones

Coordinated Systems as Interacting Systems III

Research perspectives

One should understand

- how to relate methods from statistical mechanics with coordination models
- whether notions such as *phase*, *phase transition*, or any other macroscopic system property, could be transferred from statistical mechanics to computer science
- what such notions would imply for computational systems
- whether new, original notions could apply to computational systems
- which sort of coordination model could support such notions

Socio-Technical Systems

Humans vs. software

- nowadays, a particularly-relevant class of *social systems* is represented by socio-technical systems (STS) [Whitworth, 2006]
- in STS
  - active components are mainly represented by *humans*
  - whereas interaction is almost-totally regulated by the *software infrastructure*
  - where *software agents* often play a key role
- this is the case, for instance, of *social platforms* like FaceBook [FaceBook, 2014] and LiquidFeedback [LiquidFeedback, 2014]

Physical & Computational Social Systems I

A twofold view of socio-technical systems

- the nature of STS is twofold: they are both social systems and computational systems [Verhagen et al., 2013, Omicini, 2012]
- as *complex social systems*, their complex behaviour is in principle amenable of mathematical modelling and prediction through notions and tools from statistical mechanics
- as *complex computational systems*, they are designed and built around some (either implicit or explicit) notion of coordination, ruling the interaction within components of any sort—be them either software or human ones

Physical & Computational Social Systems II

Computational systems meet physical systems

- in STS, macroscopic properties could be
  - described by exploiting the conceptual tools from physics
  - enforced by the coordination abstractions
- STS could exploit both
  - the notion of complexity by statistical mechanics, along with the mathematical tools for behaviour modelling and prediction, and
  - coordination models and languages to suitably shape the interaction space

Physical & Computational Social Systems III

Vision

Complex socio-technical systems could be envisioned

- whose implementation is based on suitable coordination models
- whose macroscopic properties can be modelled and predicted by means of mathematical tools from statistical physics

thus reconciling the scientist and the engineer views over systems

Next in Line...

Coordination for Simulation I

Simulation of complex systems is a multidisciplinary issue

- . . . ranging from physics to biology, from economics to social sciences
- no complex system of any sort can be studied nowadays without the support of suitable simulation tools
- nowadays, experiments done *in silico* are at least as relevant as those *in vitro* and *in vivo*

Coordination for Simulation II

Interaction issues are prominent in complex systems

- coordination technologies potential core of agent-based simulation frameworks
- in particular, self-organising nature-inspired coordination models are well suited for the simulation of complex systems
- so, coordination middleware could play a central role in the development of rich agent-based simulation frameworks for complex systems

Case Study: Simulating Intracellular Signalling Pathways I

Intracellular signalling pathways

- intracellular signalling involves several molecular processes along with a huge amount of signalling elements, including several kinds of proteins
- signal transduction pathways activated by G-proteins interact with one another to form a complex network that regulates diverse cellular components and controls a wide range of cellular processes [Neves et al., 2002]
- the Ras-regulated signal transduction pathways are a classical example of this kind of network [Downward, 2003]

Case Study: Simulating Intracellular Signalling Pathways II

Interaction issues in intracellular signalling pathways

- to model intracellular signalling systems, complex interaction that governs their behaviour should be first of all considered and understood
- though determining the kinetic equations of the biochemistry involved in vital functions is important, managing *interactions* for the cell to make the correct physiological decisions is even more so
- simulation of intracellular signalling pathways could be framed as mostly a coordination issue

Case Study: Simulating Intracellular Signalling Pathways III

Biochemical coordination

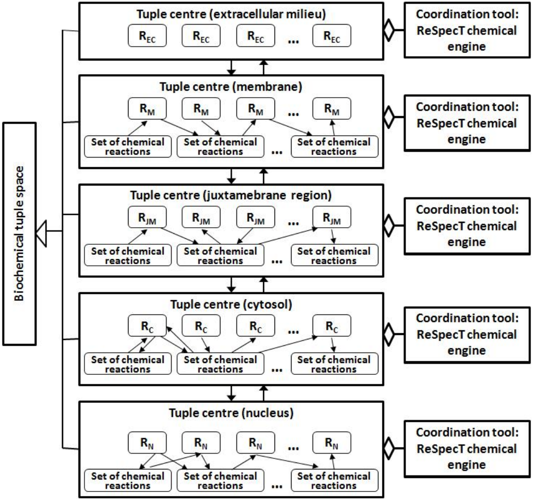
- biochemical tuple spaces [Viroli and Casadei, 2009] are the core of a model for *self-organising coordination* (BTS-SOC)
- a biochemical tuple space is a tuple space working as a compartment where *biochemical reactions* take place
- tuples in BTS-SOC are associated with an activity/pertinency value, resembling chemical *concentration*, and allowing chemical reactants to be represented as *tuples*
- biochemical laws are represented as *coordination laws* by the coordination abstraction, evolving tuple concentration over time according to a rate in the same way as chemical substances into a solution
- also, BTS-SOC laws allow for tuple *diffusion*, making it possible for products to cross compartment boundaries as a result of biochemical reactions

Case Study: Simulating Intracellular Signalling Pathways IV

Mapping cellular components and structures involved in intracellular signalling onto BTS-SOC abstractions [González Pérez et al., 2013]

Cellular components and structures involved in intracellular signalling	**Computational abstractions of the BTS-SOC model**
Extracellular milieu and intracellular compartments *(i.e., membrane, juxtamembrane region, cytosol, nucleus)*	Tuple centres
Signalling components *(i.e., membrane receptors, proteins, enzymes and genes)*	Chemical reactions sets
Signalling molecules *(i.e., first and secondary messengers)*, activation and deactivation signals	Reactants and concentrations recorded as tuples in the tuple centre

Case Study: Simulating Intracellular Signalling Pathways V



Some Final Remarks on Trans-disciplinary Research I

- the results of trans-disciplinary research efforts may appear quite obvious, once they are seen a posteriori
- just above, a nature-inspired model developed in computational terms (biochemical tuple spaces) is exploited as a computational support to the simulation of a natural system (intracellular signalling pathways)
    - in other terms, from the natural world to the computational one, and back—and it works, as one might expect

Some Final Remarks on Trans-disciplinary Research II

- however, one should also understand that trans-disciplinary research succeeds when each translation of findings between the different fields involved actually enriches the associated concepts and techniques
  - above, the BTS-SOC approach features the properties deriving from its biochemical inspiration along with those of tuple-based coordination for complex computation systems
  - when brought back to the 'natural domain' as a tool for biochemical simulation, BTS-SOC fits well for its natural inspiration, but its good performance in terms of expressive capabilities and computational efficiency *also* depends on its tuple-based structure

Some Final Remarks on Trans-disciplinary Research III

- so, while natural inspiration does not per se ensure the appropriateness of a computational approach to natural system simulation, it may in principle provide a sound grounding for the simulation of natural systems
  - biochemical inspiration of the BTS-SOC model seems to couple well with the properties of tuple-based coordination
  - BTS-SOC turns out to be a suitable framework for the simulation of biochemical systems

Next in Line. . .

Probability

- probability measures how likely some event will occur
- at its core, probability provides a mathematical framework to describe *casual* events
- . . . where casual essentially means non-deterministic
- by definition, probability deals with *single occurrences* of events
- from a scientific viewpoint, a probabilistic description *per se* has *no predictive value*: it cannot really predict the precise outcome of a phenomenon
- in any case, probability provides an *a priori* model for non-deterministic phenomena

Statistics & Probability

- statistics describes / analyses / interprets phenomena starting from the data available about them
- whenever a phenomenon has no *a priori* mathematical model (at least, not yet), statistics is concerned with getting one *a posteriori* from the available data
- accordingly, statistics is concerned with *several occurrences* of (non-deterministic) events
- probability typically provides the mathematical tools to explain data & build a model

Stochastic Systems

- stochastic systems are non-deterministic systems
- a stochastic system is one whose states are determined probabilistically
- more generally, any phenomenon requiring probability for its description is (at least in part) stochastic by definition
- roughly speaking, a probabilistic model for a stochastic system provides a *predictive framework* for a non-deterministic system

→ we cannot predict the single occurrence of a non-deterministic event, but we can predict the overall outcome of repeated occurrences of a non-deterministic event

Non-determinism, Coordination & Stochastic Behaviour

- autonomous systems such as *adaptive* and *self-** ones are *stochastic systems* at their very heart
- accordingly, a foremost feature of computational models for adaptive and self-* systems is *non-determinism*
- in order to obtain stochastic behaviours of computational systems, suitable mechanisms for non-determinism should be provided
- since most of the complexity featured by adaptive and self-* systems depends on the interaction among components, coordination models should feature non-deterministic coordination mechanisms for stochastic behaviour

Issues

- devising out some *basic mechanisms* for stochastic coordination
- finding a *minimal* set of primitives for most (all) of the most relevant stochastic systems
- showing how such mechanisms could be embedded as *tuple-based* co-ordination primitives, in order to address the general need of complex computational system engineering
- defining their formal semantics and actually measure their expressiveness

Don't Care Non-determinism in Tuple-based Models

- LINDA features *don't know* non-determinism handled with a *don't care* approach:

  don't know which tuple among the matching ones is retrieved by a getter operation (`in`, `rd`) can be neither specified nor predicted

  don't care nonetheless, the coordinated system is designed so as to keep on working whichever is the matching tuple returned

- instead, adaptive and self-organising systems require stochastic behaviours like *"most of the time do this"*, *"sometimes do that"*
- possibly with some quantitative specification of *"most of the time"* and *"sometimes"*

→ as it is, non-determinism in tuple-based models does not fit the need of stochastic behaviour specification

LINDA "Local" Nature – In Time & Space

no context — in a single getter operation, only a *local*, point-wise property affects tuple retrieval: that is, the conformance of a tuple to the template, independently of the *spatial* context

- in fact, standard getter primitives return a matching tuple independently of the other tuples currently in the same space—so, they are *"context unaware"*

no history — furthermore, in a sequence of getter operations, don't know non-determinism makes any prediction of the overall behaviour impossible—again, then, only a point-wise property can be ensured even in *time*

- sequences of standard getter operations present no meaningful distribution over time

LINDA: How to Roll a Dice?

- we define tuple space `dice`
- we represent a six-face dice as a collection of six tuples: `face(1)`, ..., `face(6)`
- we roll a dice by `rd`-ing a `face/1` tuple from `dice`:

```
dice ? rd(face(X))
```

! we do *not* obtain the overall (stochastic) behaviour of a dice: for instance, it may reasonably happen that rolling the dice 10^9 times *always* results in `X / 1`—that is, we get “1” 10^9 times in a row.

uLinda: Probabilistic Non-determinism

- we define uniform coordination primitives (`uin`, `urd`) – first mentioned in [Gardelli et al., 2007] – as the *specialisation* of Linda getter primitives featuring probabilistic non-determinism instead of don't know non-determinism
- we call the new model uLinda [Mariani and Omicini, 2013d]
- uniform primitives allow programmers to both specify and (statistically) predict the probability to retrieve one specific tuple among a bag of matching tuples
- uniform primitives are the *"basic mechanisms enabling self-organising coordination"*—that is, a minimal set of constructs able (*alone*) to impact the observable properties of a coordinated system

uLinda: "Global" Nature

Situation & prediction

Uniform primitives replace don't know non-determinism with *probabilistic non-determinism* to

- situate a primitive invocation in space
  - uniform getter primitives return matching tuples based on the other tuples in the space—so, their behaviour is *context-aware*
- predict its behaviour in time
  - sequences of uniform getter operations tend to globally exhibit a *uniform distribution* over time

uLinda: How to Roll a Dice?

- again, we define tuple space `dice`
- again, we represent a six-face dice as a collection of six tuples: `face(1)`, . . . , `face(6)`
- we roll a dice by `urd`-ing a `face/1` tuple from `dice`:

```
dice ? urd(face(X))
```

! now, we *do* obtain the overall (stochastic) behaviour of a dice:

context — at every roll, the six faces of the dice `X / 1`, . . . , `X / 6` have the same *probability* $P = 1/6$ to be selected

history — in the overall, repeating several times a roll, the six faces will tend to converge towards a uniform distribution

Informal Semantics

Operationally, uniform primitives behave as follows:

1. when executed, a uniform primitive takes a *snapshot* of the tuple space, "freezing" its state at a certain point in time—and space, being a single tuple space the target of basic Linda primitives
2. the snapshot is then exploited to assign a probabilistic value $p_i \in [0, 1]$ to any tuple $t_{i \in \{1..n\}}$ in the space—where n is the total number of tuples in the space
3. there, non-matching tuples have value $p = 0$, matching tuples have value $p = 1/m$ (where $m \leq n$ is the number of matching tuples), and the overall sum of probability values is $\sum_{i=1..n} p_i = 1$
4. the choice of the matching tuple to be returned is then statistically based on the computed probabilistic values

Formal Semantics I

[!] In order to define the semantics of (getter) uniform primitives, we rely upon a simplified version of the process-algebraic framework in [Bravetti, 2008], in particular the $\uparrow$ operator, dropping multi-level priority probabilities.

uin semantics

$$[\textsc{Synch-C}] \quad \texttt{uin}_T.P \mid \langle t_1, .., t_n \rangle \xrightarrow{T} \texttt{uin}_T.P \mid \langle t_1, .., t_n \rangle \uparrow \{(t_1, v_1), .., (t_n, v_n)\}$$

$$[\textsc{Close-C}] \quad \texttt{uin}_T.P \mid \langle t_1, .., t_n \rangle \uparrow \{(t_1, v_1), .., (t_n, v_n)\} \hookrightarrow \texttt{uin}_T.P \mid \langle t_1, .., t_n \rangle \uparrow \{(t_1, p_1), .., (t_n, p_n)\}$$

$$[\textsc{Exec-C}] \quad \texttt{uin}_T.P \mid \langle t_1, .., t_n \rangle \uparrow \{.., (t_j, p_j), ..\} \xrightarrow{t_j}_{p_j} P[t_j/T] \mid \langle t_1, .., t_n \rangle \backslash t_j$$

Formal Semantics II

[!] As for standard LINDA getter primitives, the only difference between uniform reading (`urd`) and uniform consumption (`uin`) is the non-destructive semantics of the reading primitive—transition EXEC-R.

urd semantics

$$[\text{SYNCH-C}]\quad \mathtt{urd} \mid \langle t_1, .., t_n \rangle \xrightarrow{T} \mathtt{urd}_T.P \mid \langle t_1, .., t_n \rangle \uparrow \{(t_1, v_1), .., (t_n, v_n)\}$$

$$[\text{CLOSE-C}]\quad \mathtt{urd}_T.P \mid \langle t_1, .., t_n \rangle \uparrow \{(t_1, v_1), .., (t_n, v_n)\} \hookrightarrow \mathtt{urd} \mid \langle t_1, .., t_n \rangle \uparrow \{(t_1, p_1), .., (t_n, p_n)\}$$

$$[\text{EXEC-R}]\quad \mathtt{urd} \mid \langle t_1, .., t_n \rangle \uparrow \{.., (t_j, p_j), ..\} \xrightarrow{t_j}_{p_j} P[t_j / T] \mid \langle t_1, .., t_n \rangle$$

Expressiveness: uLinda vs Linda

In [Bravetti et al., 2005], authors demonstrate that Linda-based languages *cannot* implement probabilistic models.

PME proof

The gain in expressiveness brought by uLinda is formally proven in [Mariani and Omicini, 2013b], where uniform primitives are shown to be strictly more expressive than standard Linda primitives according to probabilistic modular embedding (PME) [Mariani and Omicini, 2013c].

In particular

$$\text{uLinda} \succeq_p \text{Linda} \quad \wedge \quad \text{Linda} \not\succeq_p \text{uLinda}$$
$$\Longrightarrow \quad \text{uLinda} \not\equiv_o \text{Linda}$$

where

- $\succeq_p$ stands for "probabilistically embeds"
- $\equiv_o$ means "(PME) observational equivalence"

Next in Line...

Expressing Full Dynamics

Expressing the *full dynamics* of complex natural systems

- mostly, coordination models just capture *some* of the overall system dynamics
- which makes them basically *fail*
  - for instance, Gamma mimics chemical reactions, but does not capture essential issues in chemical processes such as reaction rates and concentration [Banâtre and Le Métayer, 1990, Banǎtre et al., 2001]
  - instead, *(bio)chemical tuple spaces* fully exploit the chemical metaphor by providing time-dependent and stochastic chemical laws [Viroli et al., 2010, Viroli and Casadei, 2009]
- more generally, the goal is to allow coordinated MAS to capture and express the full dynamics of complex natural systems

Core Mechanisms

Understanding the basic elements of expressiveness

- LINDA is a glaring example of a minimal set of coordination mechanisms providing a wide range of coordination behaviours
- the goal is understanding the minimal set of coordination primitives required to design complex stochastic behaviours
- for instance, *uniform coordination primitives* – that is, LINDA-like coordination primitives returning tuples matching a template with a uniform distribution [Gardelli et al., 2007] – seemingly capture the full-fledged dynamics of real chemical systems within the coordination abstractions

Blending Metaphors

Mixing abstractions & mechanisms from different conceptual sources

- most natural systems, when observed in their whole complexity, exhibit *layers* each one featuring its own metaphors and mechanisms
- correspondingly, many novel approaches to complex MAS coordination integrate diverse sources of inspiration, e.g.:
    - TOTA [Mamei and Zambonelli, 2004] exploits mechanisms from both stigmergic and field-based coordination
    - the SAPERE coordination model for pervasive service ecosystems [Zambonelli et al., 2011, Viroli et al., 2012] integrates
        - the *chemical* metaphor for driving the evolution of coordination abstractions
        - *biochemical* abstractions for topology and diffusion
        - the notion of *ecosystem* in order to model the overall system structure and dynamics

Predicting Complex Behaviours

Engineering unpredictable systems around predictable abstractions

- coordination models are meant to harness the complexity of complex MAS [Ciancarini et al., 2000]
- coordination abstractions are often *at the core* of complex MAS
- while this does not make complex MAS generally predictable, it makes it possible in principle to make them *partially predictable*, based on the predictably of the core coordinative behaviour
- suitably-formalised coordination abstractions, along with a suitably-defined engineering methodology, could in principle ensure the predictability of given MAS properties within generally-unpredictable MAS

Next in Line...

Knowledge-oriented Coordination I

Integrating nature-inspired with knowledge-oriented coordination

- intelligent MAS in knowledge intensive environments – as well as complex socio-technical systems, in general – require automatic understanding of data and information
- knowledge-oriented coordination exploits coordination abstractions enriched so as to allow for semantic interpretation by intelligent agents [Fensel, 2004, Nardini et al., 2013]
- for instance
  - chemical tuple spaces
  - SAPERE coordination abstractions and mechanisms
  - *semantic tuple centres* [Nardini et al., 2011]

all relay on the semantic interpretation of coordination items

Knowledge-oriented Coordination II

Self-organisation of knowledge

- explicit search of information is going to become ineffective while the amount of available knowledge grows at incredible rates
- knowledge should autonomously organise and flow from producers to consumers
- knowledge self-organisation for knowledge-intensive MAS

Knowledge-oriented Coordination III

MoK (Molecules of Knowledge) [Mariani and Omicini, 2012a]

- Molecules of Knowledge is a a nature-inspired coordination model promoting knowledge self-organisation, where
  - sources of knowledge continuously produce and inject *atoms of knowledge* in biochemical compartments
  - knowledge atoms may then aggregate in *molecules* and diffuse
  - knowledge producers, managers and consumers are modelled as *catalysts*, whose workspaces are biochemical compartments, and their knowledge-oriented actions become enzymes influencing atoms aggregation and molecules diffusion
  - so as to make relevant knowledge spontaneously aggregate and autonomously move towards potentially interested knowledge workers
- the first application scenario for experimenting with MoK is *news management* [Mariani and Omicini, 2012b]

Next in Line. . .

Lessons Learnt

In the overall...

- complex interaction is not strictly required to make complex systems
- the study of interaction is essential in any research field dealing with complexity (e.g., statistical mechanics): all contributions should be gathered and exploited in computer science to model and build complex (computational) systems
- expressiveness of coordination models determines expressiveness of systems, so it should be suitably understood, modelled, and properly measured

Part IV

Molecules of Knowledge

Next in Line...

Context, Motivation, and Goal I

- ✓ modern ICT systems go beyond Turing Machine like computation [Turing, 1939] ⇒ *computation* = *algorithm* + *interaction* [Wegner, 1997]
- ⇒ how to manage interactions? ⇒ coordination models [Malone and Crowston, 1994]
- ! *open*, highly *dynamic*, and (mostly) *unpredictable* systems present novel challenges demanding innovative coordination approaches

MoK deals with coordination issues in such a sort of systems by leveraging chemical-inspired and situated approaches, to promote *self-organisation*

Context, Motivation, and Goal II

- ✓ Socio-Technical Systems (STS) and Knowledge-Intensive Environments (KIE) combine processes, technologies, and *people*'s skills [Whitworth, 2006] to handle large repositories of *information* [Bhatt, 2001]
- ⇒ managing their interaction space is of paramount importance for both functional and *non-functional* properties
- ! engineering coordination mechanisms and strategies is far from trivial ⇒ *unpredictability* of agents' behaviour, *pace* of interactions, ...

MoK integrates Behavioural Implicit Communication (BIC) in its approach, taming unpredictability to promote *anticipatory coordination*

Context, Motivation, and Goal III

✓ data-driven approaches to coordination [Di Pierro et al., 2005], e.g. tuple space based [Gelernter, 1985] ⇒ coordinate interacting agents by managing *access to information*

⇒ why data should be viewed as *passive*, "dead" things to run algorithms upon, as in the traditional I/O paradigm?

Molecules of Knowledge (MoK) is proposed as an innovative coordination model for *self-organising knowledge management*, interpreting *information as a living entity*

Next in Line...

Overview I

Molecules of Knowledge (MoK) is a coordination model for *self-organisation of knowledge* in knowledge-intensive STS

[Mariani and Omicini, 2012a]

- MoK promotes the idea that *data is alive* [Ciancarini et al., 2002, Zambonelli et al., 2015], spontaneously interacting with other information and its prosumers (producer + consumer)

⇒ MoK pursues two main goals

  ✓ self-aggregation of information into meaningful heaps, possibly reifying relevant knowledge previously hidden

  ✓ spontaneous diffusion of information toward (potentially) interested agents

Overview II

- a MoK-coordinated system is a network of information containers (*compartments*), in which sources of information (*seeds*) continuously and spontaneously inject atomic information pieces (*atoms*)...
- ...which may aggregate into composite information chunks (*molecules*), diffuse to neighbouring compartments, lose relevance as time flows, gain relevance when exploited, and the like ...
- ...according to decentralised and spontaneous processes dictating how the system evolves (*reactions*), influenced by agents' actions (*enzymes*) and their side effects (*traces*) ...
- ...which are transparently, and possibly unintentionally, caused by human or software agents (*catalysts*) while performing their activities

Core Abstractions I

atoms *atomic units of information*, representing data along with its meta-data, decoorated with a concentration value resembling relevance

seeds *sources of information*, representing data sources as the collection of information they may make available

molecules *composite units of information*, representing collections of (semantically) related information

Core Abstractions II

catalysts *knowledge workers*, representing agents undertaking (epistemic) actions

enzymes *reification of actions*, representing the epistemic nature of actions and their context, enabling catalysts' to influence knowledge evolution

traces *reification of actions' (side) effects*, representing any (side) effect due to the action but not as its intentional primary effect

Core Abstractions III

perturbations *reactions to actions' side effects*, representing the computational functions enacted in response to agents' (inter-)actions and their side effects

reactions *knowledge dynamics processes*, representing the spontaneous computational processes supporting (meta-)information handling and evolution, as well as knowledge inference, discovery, and sharing, driven by (semantic) similarity of information

Core Abstractions IV

compartments *knowledge containers*, representing the computational abstraction responsible for handling information lifecycle, provisioning data to agents, and executing reactions

membranes *interaction channels*, representing the communication abstraction enabling 1 : 1 exchange of information, while defining the notions of locality and neighbourhood

Reactions in a Nutshell I

MoK reactions

- ✓ reactions are chemical-like coordination laws executed according to *dynamic rate expressions* [Mariani, 2013]
  - ⇒ awareness of contextual information which may affect reactions application
  - ⇒ adaptiveness to external influences put by interacting agents
- ✓ the rationale driving reactions application is *(semantic)* similarity between reactant *templates* and *actual* reactants
  - according to $\mathcal{F}_{\mathcal{M}o\mathcal{K}}$ similarity measure

Reactions in a Nutshell II

injection generates atoms from seeds

aggregation ties together *(semantically)* related atoms, or molecules, into molecules

diffusion moves atoms, molecules, and traces among neighbouring compartments

decay decreases relevance of atoms, molecules, enzymes, and traces

reinforcement increases relevance of atoms and molecules according to catalysts' *(inter-)actions*

deposit generates traces from enzymes

perturbation carries out the processes reacting to (side) effects of (interaction) activities undertaken by catalysts

(Inter-)actions in a Nutshell I

From actions to perturbations [Mariani and Omicini, 2015]

1. catalysts' actions transparently release enzymes
   - ! each action $\Rightarrow$ one $\mathit{Species}$ of enzyme
2. enzymes spontaneously and temporarily deposit traces
   - ! each enzyme $\Rightarrow$ different traces $\Rightarrow$ different perturbation actions
3. traces diffuse to neighbouring compartments to apply perturbation actions
   - ! depending on availability of matching reactants and contextual information
4. perturbation actions have different effects based on the trace they originate from and the current system state
   - ! different $\mathit{Msg} + \mathit{Context} \Rightarrow$ different behaviour

(Inter-)actions in a Nutshell II

Catalysts' actions

`share` any action adding information to the system — posting information, sharing someone else's, ...

`mark` any action marking information as relevant or not — liking a post, voting a question/answer, bookmarking a publication, ...

`annotate` any action attaching information to other information — commenting posts, replying to comments, answering questions, and ...

`connect` any action adding relationships with sources of information — adding friends, following people or posts, ...

`harvest` any action acquiring knowledge — all kinds of search actions

(Inter-)actions in a Nutshell III

MoK traces

MoK interprets (inter-)actions according to Behavioural Implicit Communication (BIC) theory [Castelfranchi et al., 2010]

! communication occurs (unintentionally) through *practical behaviour*

⇒ actions themselves, along with traces, *become the message* [Mariani and Omicini, 2013a]

✓ tacit messages, reified by MoK traces, describe these kind of messages

MoK exploits tacit messages through perturbation actions
[Mariani and Omicini, 2015]

(Inter-)actions in a Nutshell IV

MoK perturbations

`approach/repulse` *facilitating/impeding interactions* between compartments whose agents interact more often than others

`attract/drift-apart` bringing to / taking from the compartment where the action took place *information (dis)similar* to the one target of the original action

`boost/wane` *increasing/decreasing rate* of specific MoK reactions to improve MoK coordinative behaviour

`strengthen/weaken` *increasing/decreasing relevance* of information (un)related to the one target of the original action

Matchmaking in a Nutshell

! MoK needs a similarity measure for matchmaking

⇒ so as to promote *content-based* aggregation, reinforcement, diffusion, and perturbation

✓ $\mathcal{F}_{\mathcal{M}o\mathcal{K}}$ function represents the *fuzzy matchmaking* mechanism measuring similarity between atoms, molecules, etc.

⇒ *text-mining related* measures are exploited, e.g., cosine similarity, euclidean distance, average quadratic difference, ...

! $\mathcal{F}_{\mathcal{M}o\mathcal{K}}$ depends on information representation

- e.g., for documents and excerpts of documents, experimented techniques include vector-spaces, key-phrases extraction, concept-based, ...

MoK in Action: Interaction-driven Clustering

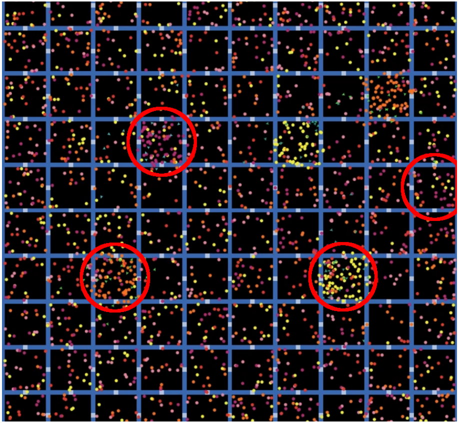


Figure: Whereas atoms and molecules are initially randomly scattered across compartments, as soon as catalysts interact clusters appear by emergence, thanks to *BIC-driven self-organisation*. Whenever new actions are performed by catalysts, MoK adaptively re-organises the spatial configuration of information so as to better tackle the new coordination needs

Next in Line. . .

Focus on. . .

Chemical Reactions as Coordination Laws

- LINDA model [Gelernter, 1985] ⇒ simple yet expressive model for *fully uncoupled* coordination in *distributed* systems [Ciancarini, 1996]
- socio-technical systems ⇒ uncertainty, unpredictability, adaptiveness
  - ✓ *unpredictability*, *uncertainty* ⇒ stochastic decision making
  - ✓ *adaptiveness* ⇒ programmability of the coordination machinery
  - ⇒ Biochemical tuple spaces [Viroli and Casadei, 2009], SAPERE [Zambonelli et al., 2011], ...
- survey regarding bio-inspired design patterns [Fernandez-Marquez et al., 2012a]
  - ✓ [Nagpal, 2004, De Wolf and Holvoet, 2007, Fernandez-Marquez et al., 2012b, Fernandez-Marquez et al., 2011, Tchao et al., 2011, Viroli et al., 2011]

1. mechanism ⇒ artificial chemical reaction ⟺ coordination law
2. evolution of the resulting "chemical solution" (coordination process) is simulated [Mariani, 2014]
   - ✓ different custom kinetic rates ⇒ different *emergent behaviours*

Probabilistic Coordination Primitives

- uniform coordination primitives (`uin`, `urd`) are *specialisations* of LINDA getter primitives featuring *probabilistic non-determinism* in returning matching tuples
- uniform primitives feature global properties
  - space: LINDA returns tuples *independently* of others, uniform primitives return tuples based on *relative multiplicity*
  - time: sequences of LINDA operations exhibit no properties, sequences of uniform operations exhibit *uniform distribution*

- *bio-inspired mechanisms* implemented on top of uniform primitives ⇒ behavioural expressiveness of uniform primitives

  [Mariani and Omicini, 2014a]

Focus on. . .

From A&A to Computational Smart Environments I

There is still a gap in current approaches to STS engineering [Schmidt and Simone, 2000], which can be closed by dealing with

mutual awareness
: as the basis for *opportunistic*, ad hoc alignment and improvisation, which ensure *flexibility*

coordinative artefacts
: *encapsulating* those portions of the coordination responsibilities that is better to *automatise*

From A&A to Computational Smart Environments II

- Activity Theory (AT) is a social psychological theory for conceptualising human activities
  - ⇒ the A&A meta-model [Omicini et al., 2008] as a *reference framework* for designing the computational part of a STS for knowledge management
- cognitive stigmergy [Ricci et al., 2007] is a first generalisation of *stigmergy* where traces are amenable of a *symbolic interpretation*
  - ⇒ cognitive stigmergy directly supports both awareness and peripheral awareness in socio-technical systems
- Behavioural Implicit Communication (BIC) is a cognitive theory of communication [Castlefranchi, 2006], where tacit messages describe the kind of messages a practical action (and its traces) may *implicitly* send to its observers [Castelfranchi et al., 2010]
  - ⇒ BIC provides a sound cognitive and social model of action and interaction for both human agents and computational agents

From A&A to Computational Smart Environments III

BIC seem to provide mutual awareness, while *coordination artefacts* the required coordinative capabilities, paving the way toward computational smart environments [Tummolini et al., 2005]

Next in Line. . .

Lessons Learnt

We are in the perfect spot to start a paradigm shift toward self-organising knowledge, where

- user-centric adaptiveness of knowledge discovery processes is the foremost goal
- measures and algorithms exploited for knowledge discovery, inference, management, and analysis natively account for users' goals
- seamlessly scale up/down/out/in naturally, being operating on the assumption that only local information is available consistently

Part V

Conclusion

Lessons Learnt

interaction growing complexity in artificial systems shifts the focus of computer science on interaction

coordination governing interaction mandates for suitably-expressive coordination models

tuples tuple-based models are the best sources for abstractions and technologies for the coordination of complex systems

NIC nature-inspired models have the potential to inject features of natural and social systems within computations systems through coordination

trends most advanced trends of NIC include dealing with complexity, stochasticity, simulation, expressiveness, knowledge-intensive environments

MoK self-organisation of knowledge is one of the most promising challenges for NIC—e.g., Molecules of Knowledge

Nature-inspired Coordination
Current Status and Research Trends

Andrea Omicini [•]
andrea.omicini@unibo.it

Stefano Mariani [•]
stefano.mariani@unimore.it

[•] Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

[•] Dipartimento di Scienze e Metodi dell'Ingegneria (DISMI)
Università di Modena e Reggio Emilia

Leipzig, Germany
23 August 2017

URLs I

Slides

- On APICe

→ http://apice.unibo.it/xwiki/bin/view/Talks/NicWi2017

- On SlideShare

→ http://www.slideshare.net/andreaomicini/natureinspired-coordination-current-status-and-research-trends

URLs II

Reference works

NIC surveys

- [Omicini, 2013a]

  `http://link.springer.com/10.1007/978-3-642-32524-3_1`

- [Omicini, 2013b]

  `http://www.hindawi.com/journals/isrn.software.engineering/2013/384903/`

MoK book

- [Mariani, 2016]

  `http://link.springer.com/10.1007/978-3-319-47109-9`

References I

Arbab, F. (2004).
Reo: A channel-based coordination model for component composition.
Mathematical Structures in Computer Science, 14:329–366.

Banătre, J.-P., Fradet, P., and Le Métayer, D. (2001).
Gamma and the chemical reaction model: Fifteen years after.
In Calude, C. S., Păun, G., Rozenberg, G., and Salomaa, A., editors, *Multiset Processing. Mathematical, Computer Science, and Molecular Computing Points of View*, volume 2235 of *LNCS*, pages 17–44. Springer.

Banâtre, J.-P. and Le Métayer, D. (1990).
The GAMMA model and its discipline of programming.
Science of Computer Programming, 15(1):55–77.

Beal, J., Dulman, S., Usbeck, K., Viroli, M., and Correll, N. (2013).
Organizing the aggregate: Languages for spatial computing.
In Mernik, M., editor, *Formal and Practical Aspects of Domain-Specific Languages: Recent Developments*, chapter 16, pages 436–501. IGI Global.
A longer version available at: `http://arxiv.org/abs/1202.5509`.

References II

Beal, J., Pianini, D., and Viroli, M. (2015).
Aggregate programming for the internet of things.
Computer, 48(9):22–30.

Berry, G. (1992).
The chemical abstract machine.
Theoretical Computer Science, 96(1):217–248.

Bhatt, G. D. (2001).
Knowledge management in organizations: Examining the interaction between technologies, techniques, and people.
Journal of Knowledge Management, 5(1):68–75.

Boltzmann, L. (1964).
Lectures on Gas Theory.
University of California Press.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999).
Swarm Intelligence: From Natural to Artificial Systems.
Santa Fe Institute Studies in the Sciences of Complexity. Oxford University Press, 198 Madison Avenue, New York, New York 10016, United States of America.

References III

Bouchaud, J.-P. and Potters, M. (2003).
Theory of Financial Risk and Derivative Pricing: From Statistical Physics to Risk Management.
Cambridge University Press, Cambridge, UK, 2nd edition.

Brambilla, M., Ferrante, E., Birattari, M., and Dorigo, M. (2013).
Swarm robotics: a review from the swarm engineering perspective.
Swarm Intelligence, 7(1):1–41.

Bravetti, M. (2008).
Expressing priorities and external probabilities in process algebra via mixed open/closed systems.
Electronic Notes in Theoretical Computer Science, 194(2):31–57.

Bravetti, M., Gorrieri, R., Lucchi, R., and Zavattaro, G. (2005).
Quantitative information in the tuple space coordination model.
Theoretical Computer Science, 346(1):28–57.

References IV

Brooks, R. A. (1991).
Intelligence without reason.
In Mylopoulos, J. and Reiter, R., editors, *12th International Joint Conference on Artificial Intelligence (IJCAI 1991)*, volume 1, pages 569–595, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Busi, N., Gorrieri, R., and Zavattaro, G. (1998).
A process algebraic view of Linda coordination primitives.
Theoretical Computer Science, 192(2):167–199.

Castelfranchi, C. (1998).
Modelling social action for AI agents.
Artificial Intelligence, 103(1-2):157–182.

Castelfranchi, C., Pezzullo, G., and Tummolini, L. (2010).
Behavioral implicit communication (BIC): Communicating with smart environments via our practical behavior and its traces.
International Journal of Ambient Computing and Intelligence, 2(1):1–12.

References V

Castlefranchi, C. (2006).
From conversation to interaction via behavioral communication: For a semiotic design of objects, environments, and behaviors.
Theories and practice in interaction design, pages 157–79.

Ciancarini, P. (1996).
Coordination models and languages as software integrators.
ACM Computing Surveys, 28(2):300–302.

Ciancarini, P., Omicini, A., and Zambonelli, F. (1999).
Coordination technologies for Internet agents.
Nordic Journal of Computing, 6(3):215–240.

Ciancarini, P., Omicini, A., and Zambonelli, F. (2000).
Multiagent system engineering: The coordination viewpoint.
In Jennings, N. R. and Lespérance, Y., editors, *Intelligent Agents VI. Agent Theories, Architectures, and Languages*, volume 1757 of *LNAI*, pages 250–259. Springer.
6th International Workshop (ATAL'99), Orlando, FL, USA, 15–17 July 1999. Proceedings.

Ciancarini, P., Tolksdorf, R., and Zambonelli, F. (2002).
A survey of coordination middleware for xml-centric applications.
The Knowledge Engineering Review, 17:389–405.

References VI

Dastani, M., Arbab, F., and de Boer, F. S. (2005).
Coordination and composition in multi-agent systems.
In Dignum, F., Dignum, V., Koenig, S., Kraus, S., Singh, M. P., and Wooldridge, M. J., editors, *4rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005)*, pages 439–446, Utrecht, The Netherlands. ACM.

De Wolf, T. and Holvoet, T. (2007).
Design patterns for decentralised coordination in self-organising emergent systems.
In *Engineering Self-Organising Systems*, pages 28–49. Springer.

Denti, E., Natali, A., and Omicini, A. (1998).
On the expressive power of a language for programming coordination media.
In *1998 ACM Symposium on Applied Computing (SAC'98)*, pages 169–177, Atlanta, GA, USA. ACM.
Special Track on Coordination Models, Languages and Applications.

Deugo, D., Weiss, M., and Kendall, E. (2001).
Reusable patterns for agent coordination.
In [Omicini et al., 2001], chapter 14, pages 347–368.

References VII

Di Pierro, A., Hankin, C., and Wiklicky, H. (2005).
Probabilistic Linda-based coordination languages.
In de Boer, F. S., Bonsangue, M. M., Graf, S., and de Roever, W.-P., editors, *3rd International Conference on Formal Methods for Components and Objects (FMCO'04)*, volume 3657 of *LNCS*, pages 120–140. Springer, Berlin, Heidelberg.

Downward, J. (2003).
Targeting RAS signalling pathways in cancer therapy.
Nature Reviews Cancer, 3(1):11–22.

FaceBook (2014).
Home page.
`http://www.facebook.com`.

Fensel, D. (2004).
Triple-space computing: Semantic web services based on persistent publication of information.
In Aagesen, F. A., Anutariya, C., and Wuwongse, V., editors, *Intelligence in Communication Systems*, volume 3283 of *LNCS*, pages 43–53.
IFIP International Conference (INTELLCOMM 2004), Bangkok, Thailand, 23–26 November 2004. Proceedings.

References VIII

Fernandez-Marquez, J., Marzo Serugendo, G., Montagna, S., Viroli, M., and Arcos, J. (2012a).
Description and composition of bio-inspired design patterns: a complete overview.
Natural Computing, pages 1–25.

Fernandez-Marquez, J. L., Di Marzo Serugendo, G., and Arcos, J. L. (2011).
Infrastructureless spatial storage algorithms.
ACM Transactions on Autonomous and Adaptive Systems (TAAS), 6(2):15.

Fernandez-Marquez, J. L., Serugendo, G. D. M., and Montagna, S. (2012b).
Bio-core: Bio-inspired self-organising mechanisms core.
In *Bio-Inspired Models of Networks, Information, and Computing Systems*, volume 103 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 59–72. Springer Berlin Heidelberg.

Fredriksson, M. and Gustavsson, R. (2004).
Online engineering and open computational systems.
In Bergenti, F., Gleizes, M.-P., and Zambonelli, F., editors, *Methodologies and Software Engineering for Agent Systems: The Agent-Oriented Software Engineering Handbook*, volume 11 of *Multiagent Systems, Artificial Societies, and Simulated Organization*, pages 377–388. Kluwer Academic Publishers.

References IX

Gardelli, L., Viroli, M., Casadei, M., and Omicini, A. (2007).
Designing self-organising MAS environments: The collective sort case.
In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Environments for MultiAgent Systems III*, volume 4389 of *LNAI*, pages 254–271. Springer.
3rd International Workshop (E4MAS 2006), Hakodate, Japan, 8 May 2006. Selected Revised and Invited Papers.

Gelernter, D. (1985).
Generative communication in Linda.
ACM Transactions on Programming Languages and Systems, 7(1):80–112.

Gelernter, D. and Carriero, N. (1992).
Coordination languages and their significance.
Communications of the ACM, 35(2):97–107.

Goldin, D. Q., Smolka, S. A., and Wegner, P., editors (2006).
Interactive Computation: The New Paradigm.
Springer.

References X

González Pérez, P. P., Omicini, A., and Sbaraglia, M. (2013).
A biochemically-inspired coordination-based model for simulating intracellular signalling pathways.
Journal of Simulation, 7(3):216–226.
Special Issue: Agent-based Modeling and Simulation.

Grassé, P.-P. (1959).
La reconstruction du nid et les coordinations interindividuelles chez Bellicositermes natalensis et Cubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs.
Insectes Sociaux, 6(1):41–80.

Jennings, N. R. (2000).
On agent-based software engineering.
Artificial Intelligence, 117(2):277–296.

Jennings, N. R. (2001).
An agent-based approach for building complex software systems.
Communications of the ACM, 44(4):35–41.

References XI

Kauffman, S. A. (1993).
The Origins of Order: Self-organization and Selection in Evolution.
Oxford University Press.

Kauffman, S. A. (2003).
Investigations.
Oxford University Press.

Kube, C. R. and Bonabeau, E. (2000).
Cooperative transport by ants and robots.
Robotics and Autonomous Systems, 30(1):85 – 101.

LiquidFeedback (2014).
Home page.
http://liquidfeedback.org.

Liu, J. and Tsui, K. C. (2006).
Toward nature-inspired computing.
Communications of the ACM, 49(10):59–64.

References XII

Malone, T. and Crowston, K. (1994).
The interdisciplinary study of coordination.
ACM Computing Surveys, 26(1):87–119.

Mamei, M. and Zambonelli, F. (2004).
Programming pervasive and mobile computing applications with the TOTA middleware.
In *Pervasive Computing and Communications*, pages 263–273.
2nd IEEE Annual Conference (PerCom 2004), Orlando, FL, USA, 14–17 March 2004. Proceedings.

Mamei, M. and Zambonelli, F. (2006).
Field-Based Coordination for Pervasive Multiagent Systems. Models, Technologies, and Applications.
Springer Series in Agent Technology. Springer.

Mantegna, R. N. and Stanley, H. E. (1999).
Introduction to Econophysics: Correlations and Complexity in Finance.
Cambridge University Press, Cambridge, UK.

References XIII

Mariani, S. (2013).
Parameter engineering vs. parameter tuning: the case of biochemical coordination in MoK.
In Baldoni, M., Baroglio, C., Bergenti, F., and Garro, A., editors, *From Objects to Agents*, volume 1099 of *CEUR Workshop Proceedings*, pages 16–23, Turin, Italy. Sun SITE Central Europe, RWTH Aachen University.

Mariani, S. (2014).
On the "local-to-global" issue in self-organisation: Chemical reactions with custom kinetic rates.
In *Eighth IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops, SASOW 2014*, Eighth IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops, SASOW 2014, pages 61 – 67, London, UK. IEEE.
Best student paper award.

Mariani, S. (2016).
Coordination of Complex Sociotechnical Systems: Self-organisation of Knowledge in MoK.
Artificial Intelligence: Foundations, Theory, and Algorithms. Springer International Publishing, 1st edition.

References XIV

Mariani, S. and Omicini, A. (2012a).
Molecules of knowledge: Self-organisation in knowledge-intensive environments.
In *Intelligent Distributed Computing VI*, pages 17–22. Springer, Calabria, Italy.
6th International Symposium on Intelligent Distributed Computing (IDC 2012).

Mariani, S. and Omicini, A. (2012b).
Self-organising news management: The *Molecules of Knowledge* approach.
In Fernandez-Marquez, J. L., Montagna, S., Omicini, A., and Zambonelli, F., editors, *1st International Workshop on Adaptive Service Ecosystems: Natural and Socially Inspired Solutions (ASENSIS 2012)*, pages 11–16, SASO 2012, Lyon, France.
Pre-proceedings.

Mariani, S. and Omicini, A. (2013a).
MoK: Stigmergy meets chemistry to exploit social actions for coordination purposes.
In Verhagen, H., Noriega, P., Balke, T., and de Vos, M., editors, *Social Coordination: Principles, Artefacts and Theories (SOCIAL.PATH)*, pages 50–57, AISB Convention 2013, University of Exeter, UK. The Society for the Study of Artificial Intelligence and the Simulation of Behaviour.

References XV

Mariani, S. and Omicini, A. (2013b).
Probabilistic embedding: Experiments with tuple-based probabilistic languages.
In *28th ACM Symposium on Applied Computing (SAC 2013)*, pages 1380–1382, Coimbra, Portugal.
Poster Paper.

Mariani, S. and Omicini, A. (2013c).
Probabilistic modular embedding for stochastic coordinated systems.
In Julien, C. and De Nicola, R., editors, *Coordination Models and Languages*, volume 7890 of *LNCS*, pages 151–165. Springer.
15th International Conference (COORDINATION 2013), Florence, Italy, 3–6 June 2013. Proceedings.

Mariani, S. and Omicini, A. (2013d).
Tuple-based coordination of stochastic systems with uniform primitives.
In Baldoni, M., Baroglio, C., Bergenti, F., and Garro, A., editors, *From Objects to Agents*, volume 1099 of *CEUR Workshop Proceedings*, pages 8–15, Turin, Italy. Sun SITE Central Europe, RWTH Aachen University.
XIV Workshop (WOA 2013). Workshop Notes.

References XVI

Mariani, S. and Omicini, A. (2014a).
Coordination mechanisms for the modelling and simulation of stochastic systems: The case of uniform primitives.
SCS M&S Magazine.

Mariani, S. and Omicini, A. (2014b).
TuCSoN coordination for MAS situatedness: Towards a methodology.
In Santoro, C. and Bergenti, F., editors, *WOA 2014 – XV Workshop Nazionale "Dagli Oggetti agli Agenti"*, volume 1260 of *CEUR Workshop Proceedings*, pages 62–71. Sun SITE Central Europe, RWTH Aachen University.

Mariani, S. and Omicini, A. (2015).
Anticipatory coordination in socio-technical knowledge-intensive environments: Behavioural implicit communication in MoK.
In Gavanelli, M., Lamma, E., and Riguzzi, F., editors, *AI*IA 2015, Advances in Artificial Intelligence*, volume 9336 of *Lecture Notes in Computer Science*, chapter 8, pages 102–115. Springer International Publishing.
XIVth International Conference of the Italian Association for Artificial Intelligence, Ferrara, Italy, September 23–25, 2015, Proceedings.

References XVII

Mézard, M. and Montanari, A. (2009).
Information, Physics, and Computation.
Oxford University Press, Oxford, UK.

Milner, R. (1993).
Elements of interaction: Turing award lecture.
Communications of the ACM, 36(1):78–89.

Minsky, N. H., Ungureanu, V., and Minsky, Y. M. (2001).
Safe tuplespace-based coordination in multiagent systems.
Applied Artificial Intelligence: An International Journal, 15(1):11–33.
Special Issue: Coordination Models and Languages in AI.

Nagpal, R. (2004).
A catalog of biologically-inspired primitives for engineering self-organization.
In *Engineering Self-Organising Systems*, pages 53–62. Springer.

Nardini, E., Omicini, A., and Viroli, M. (2013).
Semantic tuple centres.
Science of Computer Programming, 78(5):569–582.
Special section: Self-Organizing Coordination.

References XVIII

Nardini, E., Omicini, A., Viroli, M., and Schumacher, M. I. (2011).
Coordinating e-health systems with TuCSoN semantic tuple centres.
Applied Computing Review, 11(2):43–52.

Neves, S. R., Ram, P. T., and Iyengar, R. (2002).
G protein pathways.
Science, 296(5573):1636–1639.

Nishimori, H. (2001).
Statistical Physics of Spin Glasses and Information Processing: An Introduction, volume 111 of *International Series of Monographs on Physics*.
Clarendon Press, Oxford, UK.

Odell, J. J. (2002).
Objects and agents compared.
Journal of Object Technology, 1(1):41–53.

Omicini, A. (2000).
Hybrid coordination models for handling information exchange among Internet agents.
In Bonarini, A., Colombetti, M., and Lanzi, P. L., editors, *Workshop "Agenti intelligenti e Internet: teorie, strumenti e applicazioni"*, pages 1–4, 7th AI*IA Convention (AI*IA 2000), Milano, Italy.

References XIX

Omicini, A. (2012).
Agents writing on walls: Cognitive stigmergy and beyond.
In Paglieri, F., Tummolini, L., Falcone, R., and Miceli, M., editors, *The Goals of Cognition. Essays in Honor of Cristiano Castelfranchi*, volume 20 of *Tributes*, chapter 30, pages 565–578. College Publications, London.

Omicini, A. (2013a).
Nature-inspired coordination for complex distributed systems.
In Fortino, G., Bădică, C., Malgeri, M., and Unland, R., editors, *Intelligent Distributed Computing VI*, volume 446 of *Studies in Computational Intelligence*, pages 1–6. Springer.
6th International Symposium on Intelligent Distributed Computing (IDC 2012), Calabria, Italy, 24-26 September 2012. Proceedings. Invited paper.

Omicini, A. (2013b).
Nature-inspired coordination models: Current status, future trends.
ISRN Software Engineering, 2013.
Review Article.

Omicini, A. and Papadopoulos, G. A. (2001).
Editorial: Why coordination models and languages in AI?
Applied Artificial Intelligence: An International Journal, 15(1):1–10.
Special Issue: Coordination Models and Languages in AI.

References XX

Omicini, A., Ricci, A., and Viroli, M. (2006).
The multidisciplinary patterns of interaction from sciences to Computer Science.
In Goldin, D. Q., Smolka, S. A., and Wegner, P., editors, *Interactive Computation: The New Paradigm*, pages 395–414. Springer.

Omicini, A., Ricci, A., and Viroli, M. (2008).
Artifacts in the A&A meta-model for multi-agent systems.
Autonomous Agents and Multi-Agent Systems, 17(3):432–456.
Special Issue on Foundations, Advanced Topics and Industrial Perspectives of Multi-Agent Systems.

Omicini, A., Ricci, A., Viroli, M., Castelfranchi, C., and Tummolini, L. (2004).
Coordination artifacts: Environment-based coordination for intelligent agents.
In Jennings, N. R., Sierra, C., Sonenberg, L., and Tambe, M., editors, *3rd international Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)*, volume 1, pages 286–293, New York, USA. ACM.

Omicini, A. and Zambonelli, F. (1999).
Coordination for Internet application development.
Autonomous Agents and Multi-Agent Systems, 2(3):251–269.
Special Issue: Coordination Mechanisms for Web Agents.

References XXI

Omicini, A., Zambonelli, F., Klusch, M., and Tolksdorf, R., editors (2001).
Coordination of Internet Agents: Models, Technologies, and Applications.
Springer.

Parunak, H. V. D. (2006).
A survey of environments and mechanisms for human-human stigmergy.
In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Environments for Multi-Agent Systems II*, volume 3830 of *LNCS*, pages 163–186. Springer.

Parunak, H. V. D., Brueckner, S., and Sauter, J. (2002).
Digital pheromone mechanisms for coordination of unmanned vehicles.
In Castelfranchi, C. and Johnson, W. L., editors, *1st International Joint Conference on Autonomous Agents and Multiagent systems*, volume 1, pages 449–450, New York, NY, USA. ACM.

Ricci, A., Omicini, A., Viroli, M., Gardelli, L., and Oliva, E. (2007).
Cognitive stigmergy: Towards a framework based on agents and artifacts.
In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Environments for MultiAgent Systems III*, volume 4389 of *LNCS*, pages 124–140. Springer.
3rd International Workshop (E4MAS 2006), Hakodate, Japan, 8 May 2006. Selected Revised and Invited Papers.

References XXII

Ricci, A., Viroli, M., and Omicini, A. (2005).
Environment-based coordination through coordination artifacts.
In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Environments for Multi-Agent Systems*, volume 3374 of *LNAI*, pages 190–214. Springer.
1st International Workshop (E4MAS 2004), New York, NY, USA, 19 July 2004. Revised Selected Papers.

Rossi, D., Cabri, G., and Denti, E. (2001).
Tuple-based technologies for coordination.
In [Omicini et al., 2001], chapter 4, pages 83–109.

Şahin, E. (2005).
Swarm robotics: From sources of inspiration to domains of application.
In Şahin, E. and Spears, W. M., editors, *Swarm Robotics*, pages 10–20. Springer.
SAB 2004 International Workshop, Santa Monica, CA, USA, July 17, 2004, Revised Selected Papers.

References XXIII

Schmidt, K. and Simone, C. (2000).
Mind the gap! Towards a unified view of CSCW.
In Dieng, R., Giboin, A., Karsenty, L., and De Michelis, G., editors, *Designing Cooperative Systems: The Use of Theories and Models*, volume 58 of *Frontiers in Artificial Intelligence and Applications*, Sophia Antipolis, France. IOS Press.
4th International Conference on the Design of Cooperative Systems (COOP 2000), Proceedings.

Simon, H. A. (1962).
The architecture of complexity.
Proceedings of the American Philosophical Society, 106(6):467–482.

Stanley, H. E. (2008).
Econophysics and the current economic turmoil.
American Physical Society News, 17(11):8.
The Back Page.

Suchman, L. A. (1987).
Plans and Situated Actions: The Problem of Human-Machine Communication.
Cambridge University Press, New York, NYU, USA.

References XXIV

Tchao, A.-E., Risoldi, M., and Di Marzo Serugendo, G. (2011).
Modeling self-* systems using chemically-inspired composable patterns.
In *Self-Adaptive and Self-Organizing Systems (SASO), 2011 Fifth IEEE International Conference on*, pages 109–118.

Tolksdorf, R. and Menezes, R. (2004).
Using Swarm Intelligence in Linda Systems.
In Omicini, A., Petta, P., and Pitt, J., editors, *Engineering Societies in the Agents World IV*, volume 3071 of *LNCS*, pages 49–65. Springer.
4th International Workshops (ESAW 2003), London, UK, 29-31 October 2003. Revised Selected and Invited Papers.

Tummolini, L., Castelfranchi, C., Ricci, A., Viroli, M., and Omicini, A. (2005).
"Exhibitionists" and "voyeurs" do it better: A shared environment approach for flexible coordination with tacit messages.
In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Environments for Multi-Agent Systems*, volume 3374 of *Lecture Notes in Artificial Intelligence*, pages 215–231. Springer.

Turing, A. M. (1937).
On computable numbers, with an application to the entscheidungsproblem.
Proceedings of the London Mathematical Society, s2-42(1):230–265.

References XXV

Turing, A. M. (1939).
Systems of logic based on ordinals.
Proceedings of the London Mathematical Society, 2(1):161–228.

Verhagen, H., Noriega, P., Balke, T., and de Vos, M., editors (2013).
Social Coordination: Principles, Artefacts and Theories (SOCIAL.PATH), AISB Convention 2013, University of Exeter, UK. The Society for the Study of Artificial Intelligence and the Simulation of Behaviour.

Viroli, M. and Casadei, M. (2009).
Biochemical tuple spaces for self-organising coordination.
In Field, J. and Vasconcelos, V. T., editors, *Coordination Languages and Models*, volume 5521 of *LNCS*, pages 143–162. Springer, Lisbon, Portugal.
11th International Conference (COORDINATION 2009), Lisbon, Portugal, June 2009. Proceedings.

Viroli, M., Casadei, M., Montagna, S., and Zambonelli, F. (2011).
Spatial coordination of pervasive services through chemical-inspired tuple spaces.
ACM Transactions on Autonomous and Adaptive Systems, 6(2):14:1–14:24.

References XXVI

Viroli, M., Casadei, M., Nardini, E., and Omicini, A. (2010).
Towards a chemical-inspired infrastructure for self-* pervasive applications.
In Weyns, D., Malek, S., de Lemos, R., and Andersson, J., editors, *Self-Organizing Architectures*, volume 6090 of *LNCS*, chapter 8, pages 152–176. Springer.
1st International Workshop on Self-Organizing Architectures (SOAR 2009), Cambridge, UK, 14-17 September 2009, Revised Selected and Invited Papers.

Viroli, M. and Omicini, A. (2006).
Coordination as a service.
Fundamenta Informaticae, 73(4):507–534.
Special Issue: Best papers of FOCLASA 2002.

Viroli, M., Pianini, D., Montagna, S., and Stevenson, G. (2012).
Pervasive ecosystems: a coordination model based on semantic chemistry.
In Ossowski, S., Lecca, P., Hung, C.-C., and Hong, J., editors, *27th Annual ACM Symposium on Applied Computing (SAC 2012)*, Riva del Garda, TN, Italy. ACM.

Wegner, P. (1997).
Why interaction is more powerful than algorithms.
Communications of the ACM, 40(5):80–91.

References XXVII

Wegner, P. and Goldin, D. (1999).
Mathematical models of interactive computing.
Technical report, Brown University, Providence, RI, USA.

Wegner, P. and Goldin, D. (2003).
Computation beyond Turing machines.
Communications of the ACM, 46(4):100–102.

Whitworth, B. (2006).
Socio-technical systems.
In Ghaou, C., editor, *Encyclopedia of Human Computer Interaction*, pages 533–541. IGI Global.

Zambonelli, F., Castelli, G., Ferrari, L., Mamei, M., Rosi, A., Di Marzo Serugendo, G., Risoldi, M., Tchao, A.-E., Dobson, S., Stevenson, G., Ye, Y., Nardini, E., Omicini, A., Montagna, S., Viroli, M., Ferscha, A., Maschek, S., and Wally, B. (2011).
Self-aware pervasive service ecosystems.
Procedia Computer Science, 7:197–199.
Proceedings of the 2nd European Future Technologies Conference and Exhibition 2011 (FET 11).

References XXVIII

Zambonelli, F. and Omicini, A. (2004).
Challenges and research directions in agent-oriented software engineering.
Autonomous Agents and Multi-Agent Systems, 9(3):253–283.
Special Issue: Challenges for Agent-Based Computing.

Zambonelli, F., Omicini, A., Anzengruber, B., Castelli, G., DeAngelis, F. L., Di Marzo Serugendo, G., Dobson, S., Fernandez-Marquez, J. L., Ferscha, A., Mamei, M., Mariani, S., Molesini, A., Montagna, S., Nieminen, J., Pianini, D., Risoldi, M., Rosi, A., Stevenson, G., Viroli, M., and Ye, J. (2015).
Developing pervasive multi-agent systems with nature-inspired coordination.
Pervasive and Mobile Computing, 17:236–252.
Special Issue "10 years of Pervasive Computing" In Honor of Chatschik Bisdikian.

Zavattaro, G. (1998).
On the incomparability of Gamma and Linda.
Technical Report SEN-R9827, CWI, Amsterdam, The Netherlands.