Coordination Models and Technologies toward Self-Organising Systems

Andrea Omicini
andrea.omicini@unibo.it

Alma Mater Studiorum—Università di Bologna

CILC 2011
Università degli Studi "Gabriele D'Annunzio" di Chieti e Pescara
Pescara, 31st of August 2011

Outline

Complexity & Coordination

Complex computational systems. . .

- . . . intelligent, knowledge-intensive, pervasive, self-organising systems
- could be seen as the dynamic ensemble of a large number of distributed components, heterogeneous in nature, structure and behaviour
- put together *somehow* so as to build up a coherent overall system behaviour

What is "somehow"?

- This is the key issue in the research for abstractions, models, technologies and methodologies for the engineering of complex systems
- This is the issue of coordination models and languages [Papadopoulos and Arbab, 1998, Busi et al., 2001]

Evolution of Coordination Models I

Origins of coordination models and languages

- Coordination models originated in the context of *closed* and *parallel* systems
- E.g., generative communication [Gelernter, 1985] as a means to enable/promote parallel computations

Evolution of Coordination Models II

Coordination models and languages today

- After twenty-five years of literature on coordination models and languages. . .
- . . . they are now conceived as the potential sources for the abstractions and the technologies around which complex computational systems can be designed and built

? How did this happen?

- In this talk, I will attempt to provide you with a possible explanation and a perspective – based on [Omicini and Viroli, 2011] –, focussing in particular on the role of Computational Logics (CL henceforth)

Outline

Space-based Models I

Tuple-based models

- Tuple-based models [Rossi et al., 2001] represent the main class of space-based coordination models
- There, communication and coordination occur through a shared data space
  - as in the case of blackboard systems [Corkill, 1991]
- A shared data space for communication, whose life is independent of the interacting components, is the conceptual basis for generative communication [Gelernter, 1985]
- As such, it represents the essential environment abstraction for the support of openness in distributed systems

Space-based Models II

Persistent coordination abstraction

- The key idea of generative communication is a coordination abstraction persisting along with the messages exchanged
- This is the essential pre-requisite for a system where components may come and go at run-time. . .
- . . . and provides for time uncoupling, which makes it possible to conceive and design patterns of interaction that could survive the potential erraticism of component behaviour

Space-based Models III

Middleware coordination abstraction

- The notion of permanent coordination abstraction mandates for coordination middleware, whose life is independent of the coordinating component's life. . .
- . . . so that patterns of interaction could be enforced by suitably *shaping* the *computational environment* independently of the computational components

Space-based Models & Computational Logics

Pioneers: Shared Prolog

- Blackboard models [Corkill, 1991] largely worked as an inspiration for space-based coordination with computational logics
- Brogi and Ciancarini pioneered the *blackboard-based* interpretation of LP through their Shared Prolog [Brogi and Ciancarini, 1991]
    - where dynamics of logic DB is interpreted as communication
    - with `in` and `out` "replacing" `assert` and `retract`
- *In nuce*, Shared Prolog already contains some of the main features of CL-based approaches to coordination
    - unification as the matching mechanism
    - communication as/through a logic theory
    - distributed knowledge partition as multiple distributed logic programs (KS)

Origins of Tuple-based Models I

Linda

- The ancestor of all tuple-based models is Linda
- In Linda [Gelernter, 1985], components communicate and synchronise by exchanging tuples through a shared tuple space
- There, communication and coordination occur through a shared data space
  - communication via tuples
  - coordination via space behaviour in response to coordination primitives

! Linda was first conceived to support parallel computation in closed systems—at least, with no apparent concern for open systems

Origins of Tuple-based Models II

From closed to open systems

- LINDA introduces an environment abstraction devoted to the management of the (agent) interaction space
- As a conceptual consequence, computation and coordination
  - conceived as the management of interaction [Wegner, 1997]

  were to be
  - considered as two orthogonal dimensions of computer-based systems [Gelernter and Carriero, 1992]
  - handled – that is, analysed, modelled, designed, programmed – in an *independent* way, by adopting suitable abstractions and mechanisms

Basic Features of LINDA-based Models I

Tuples

- A tuple is an ordered collection of possibly-heterogeneous knowledge chunks

→ Synchronisation based on the availability of tuples means essentially synchronisation based on the availability of structured knowledge of some sort

→ Tuple-based coordination is first of all knowledge-based coordination

- where tuple spaces are possibly interpreted as knowledge repositories

Basic Features of LINDA-based Models II

Associative access

- Tuple spaces are accessed associatively
  - queries specify tuple templates that match tuples based on their structure and the data they contain

→ Complete uncoupling in communication
  - information neither on the sender nor on the structure of the share space is required for a message to be received

- Synchronisation possible over a partial representation of knowledge—the tuple template
  - a fundamental feature in all the contexts where information is often vague, inaccurate, incomplete, or partially specified—as is typical in knowledge-intensive systems

Basic Features of LINDA-based Models III

Logic tuple-space models

- Tuples as first-order logic (FOL) facts
  - Components coordinate through FOL tuples
  - Unification for associative access to tuple in the space
  - Same syntax for tuples and templates
- Tuple spaces are FOL theories
  - the shared communication space can be interpreted as a logic-based knowledge repository used for component coordination
  - each tuple space could be thought as the FOL theory representing some domain element relevant for component coordination
  - → "semantic" interpretation of logic tuple space (in engineering process acceptation)
- Examples
  - Shared Prolog [Brogi and Ciancarini, 1991]
  - ReSpecT [Omicini and Denti, 2001]

Essential Features of LINDA-derived Models I

Two other features characterise tuple-based models as they descend from the original LINDA ancestor

- distribution of the coordination abstractions
- expressiveness of the coordination abstractions

respectively termed as [Busi et al., 2001]

- "reshaping the coordination media"
- "programming the coordination rules"

Essential Features of LINDA-derived Models II

Reshaping the coordination media

- Distribution is essential for any complex system
- In the same way as components of a distributed system are spread all over the system topology, multiple tuple spaces fill the system environment, providing for distributed coordination abstractions
  - JavaSpaces [Freeman et al., 1999] by Sun
  - TSpaces [Wyckoff et al., 1998] by IBM
- This paves the way toward pervasive coordination systems
- Also, expressing the environment topology in a distributed setting is essential for the coordination of local interaction as well as of mobile components
  - LIME [Murphy et al., 2006]
  - KLAIM [De Nicola et al., 1998]

Essential Features of LINDA-derived Models III

Programming the coordination rules

- The expressiveness of coordination media often needs to be tailored to the complexity and peculiarity of the specific coordinated system
- So, a number of LINDA derivatives, e.g.
  - Law-Governed Interaction [Minsky and Ungureanu, 2000]
  - MARS [Cabri et al., 2000]
  - ReSpecT [Omicini and Denti, 2001]
- focus on the programmability of the tuple space, so as to
  - make it possible to explicitly express the rules of coordination
  - embed them within the coordination abstraction
- There, arbitrarily-complex coordination policies can be in principle associated to each and every coordination medium, which could be individually programmed so as to embed either global or local coordination policies, as required by the specific coordinated systems

Essential Features of LINDA-derived Models IV

Situatedness

- The ability to define arbitrarily-complex coordination policies and to embed them within the coordination media should be in principle coupled with the ability to capture and react to arbitrary environment events

! . . . otherwise, environment-based coordination would not be supported directly by the coordination medium

- This provides for the level of situatedness typically required by coordination in pervasive computational environments
  - Situated ReSpecT [Casadei and Omicini, 2009]

Outline

Not Only LINDA: The Case of Gamma I

Legacy beyond LINDA: Gamma

- The other ancestor of space-based models is Gamma [Banâtre et al., 2001]
- Not a derivative of LINDA
- In Gamma, a shared coordination space is ruled by chemical-like laws defined by the programmers
  - thus, Gamma reminds the features of programmable tuple space models

Not Only LINDA: The Case of Gamma II

Source of inspiration for Gamma

- Analogous to 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
- Gamma is a nature-inspired coordination model

Nature-inspired Coordination Models I

Nature-inspired computing

- The main idea is to extract models and patterns from natural systems of any sorts, and apply them within computational contexts
- Nature-inspired models includes neural networks, genetic algorithms, swarm intelligence, . . .
- Strict relationship between coordination and complexity of systems

→ nature-inspired models of coordination are of particular interest in the engineering of complex computational systems

Nature-inspired Coordination Models II

Nature-inspired coordination

- A whole class of coordination models is inspired by the extraction of patterns from natural and social complex systems
- Nature-inspired coordination models are mostly driven by the idea that
  - working complex systems exist in the real world
  - which we can observe so as
  - to understand their basic principles and mechanisms, to abstract them, and to bring them within our artificial systems
- Understanding the principles and mechanisms of coordination within complex natural systems

→ defining coordination models and technologies for complex artificial systems

Field-based Coordination Models

Field-based coordination

- Field-based coordination models are inspired by the way masses and particles move and self-organise according to gravitational/electromagnetic fields [Mamei and Zambonelli, 2006]
- Typically, a pervasive coordination infrastructure generates and maintains computational force fields which are sensed & modified by agents moving through the fields, according to the field intensity and sort

TOTA

Field-based coordination in TOTA

- In TOTA [Mamei and Zambonelli, 2004], computational force fields takes the form of distributed tuples
- Distributed tuples
  - are generated by both the active components and by the pervasive coordination infrastructure
  - propagate across the environment
  - drive the actions and motion of the component themselves—e.g. allowing two mobile agents to find each other in a dynamic network

Stigmergic Coordination I

Origins of nature-inspired coordination models

- Historically, nature-inspired models of coordination are grounded in studies on the behaviour of social insects, like ants or termites
- The key concept there is stigmergy, introduced by [Grassé, 1959] as an explanation for the coordination observed in termites societies, where

  > *"The coordination of tasks and the regulation of constructions are not directly dependent from the workers, but from constructions themselves."*

- Namely, the notion of stigmergy generally refers to a set of coordination mechanisms mediated by the environment...
- ... which leads to the emergent behaviours typical of self-organising systems

Stigmergic Coordination II

Example: Ants

- In ant colonies, chemical substances – namely pheromone – act as environment markers for specific social activities
- Pheromones drive both the individual & the social behaviour of ants
  - by the way, similarly to what happens e.g. in TOTA

Environment-based Coordination

Coordination through the environment

- Most of nature-inspired coordination models are characterised by the active role of the environment
- For instance, both field-based and stigmergic coordination are based on some notion of environment affecting the behaviour of coordinated components by shaping the space of component interaction
- Generally speaking, environment-based coordination systematically adopts structured abstractions for shaping the environment of system components so as to govern their interactions [Ricci et al., 2005]
- So, environment-based coordination generalises for instance upon both field-based and stigmergic coordination

Cognitive Stigmergy I

Beyond stigmergy

- Stigmergy concerns emergent coordination in societies composed by a large amount of ant-like, non-rational agents
- However, stigmergic patterns are can be observed also in the context of societies composed by cognitive / rational agents [Omicini et al., 2004]
- In this context
    - modifications to the environment are often amenable of an interpretation in the context of a shared, conventional system of signs
    - the interacting agents feature cognitive abilities that can be used in the stigmergy-based interaction

Cognitive Stigmergy II

From signals to signs: Cognitive stigmergy

- The notion of cognitive stigmergy was introduced as a first generalisation of stigmergic coordination to enable social activities of cognitive agents [Ricci et al., 2007]
- Multiple-level coordination between heterogeneous components
  - ordinary components perceive environment markers as mere signals and react accordingly
  - intelligent components can read them as signs, and behave according to their symbolic interpretation

Tuples & Stigmergy

Tuple-based models as a computational specimen for stigmergy

- multiple tuple spaces physically/logically distributed in a computational system could be seen as the building blocks of the system's environment—the "walls" upon which coordinating agents can read and write signs in the form of tuples [Omicini, 2011]
- suspensive coordination primitives could be used either by "stupid" processes to synchonise upon pre-determined tuple patterns, or by intelligent agents properly interpreting the symbolic content of tuples—as in the case of logic tuples, for instance [Denti and Omicini, 1999]

→ self-organising patterns based on either stigmergy or cognitive stigmergy can be in principle built upon tuple-based coordination models and technologies

Logic Tuples & Stigmergy

Logic symbols for stigmergy

- What if pheromones are represented by logic tuples in a tuple space?
- Non-logic (non-intelligent) agents could just react to pheromone, logic (intelligent) agents could also understand them and react appropriately
- . . . which fits the conceptual framework of cognitive stigmergy

Outline

Self-organising Coordination I

A shared legacy

- Nature-inspired coordination models
  - e.g., chemical, field-based, and stigmergic coordination models
- share a fundamental feature
  - they come from the core of complex natural self-organising systems
- As such, they are seemingly the most intuitive sources for abstractions and mechanisms around which *self-organising* artificial systems could be designed and built

Self-organising Coordination II

Towards self-organising coordination

- Generally speaking, self-organising coordination can be defined as the management of system interactions featuring self-organising properties
- . . . namely, where [Viroli et al., 2009]
  - interactions are local
  - global desired effects of coordination appear by emergence

Self-organising Coordination III

The problem of self-organising coordination

- In most classical coordination models the environment is filled with coordination media enacting coordination laws that are
  - typically reactive
  - (essentially) deterministic
  - global
- In self-organising systems coordination
  - patterns typically appear at the global level by emergence
  - from probabilistic, time-dependent coordination laws
  - based on local criteria

Features of Self-organising Coordination

Required features of self-organising coordination models

According to [Viroli et al., 2009], the required features of coordination models for self-organising systems are

- Topology & locality
- On-line character
- Time-dependency
- Probabilistic behaviour

Topology & Locality

Coordination middleware

- Topology & locality mostly affect the nature of the coordination middleware
- The coordination media provided should
  - be associated to distributed locations
  - mostly govern interaction among local components
  - not be merely reactive to interaction
  - instead, be enacted as always-running services able to adapt their coordinative behaviour at run time

  → as in the case of LIME, ReSpecT and TOTA, among the others

Time-dependency & Probabilistic Behaviour I

Classical coordination models

- Apparently classical coordination models apparently address the issues of time-dependency and probabilistic behaviour in some way
- For instance
  - tuple matching templates are returned in a non-deterministic way
  - chemical laws are known to be probabilistic and time-dependent

Time-dependency & Probabilistic Behaviour II

Nonetheless. . .

- On the one hand
  - non-determinism of classical tuple-based model is just a "don't know" non-determinism
  - instead, non-determinism in self-organising systems is typically stochastic
  - → models like TOTA, SwarmLinda [Tolksdorf and Menezes, 2004] and StoKlaim [Bravetti et al., 2009] have introduced stochastic mechanisms within tuple-based coordination
- On the other hand
  - classical chemical coordination models like Gamma and CHAM do *not* really reproduce chemical behaviours
  - since they can express neither stochastic behaviours nor time-dependent coordination rules

Chemical & Tuple-based Coordination I

Chemical tuple spaces

- As a result, a chemical tuple-space model and infrastructure have been defined [Viroli et al., 2010] . . .
- . . . that embodies all the typical features of self-organisation in natural chemical systems
- There, self-organisation could be achieved in two ways
  - either by means of the behaviour of an individual chemical tuple space (intra-space self-organisation)
  - or by means of a suitable pattern of interaction among chemical tuple spaces (inter-space self-organisation)

Chemical & Tuple-based Coordination II

Chemical tuple-based coordination for pervasive systems

- Chemical tuple-based coordination is based on tuples evolving by mimicking chemical systems—that is, in terms of reaction and diffusion rules that apply to tuples modulo semantic match
- General-purpose chemical reactions inspired by population dynamics are exploited that involve chemical tuples
- Such reactions can be used in pervasive applications to enact spatial computing patterns of competition and gradient-based interaction
- In [Viroli et al., 2011] such a model is tested against a self-adaptive display infrastructure providing people nearby with several visualization services (advertisements, news, personal and social content)—more generally, for self-aware pervasive service ecosystems

Tuple-based Models for KIE I

Toward knowledge-intensive environments (KIE)

- In the overall, suitably-extended tuple-based models provide a promising platform for the design and development of self-organising coordinated systems
- Nonetheless, *knowledge-intensive* application scenarios pose a huge challenge for tuple-based models
- There, the aforementioned benefits of tuple-based coordination in terms of knowledge-based coordination fade in front of the problems it induces in terms of syntax & (mostly) semantics
  - e.g., two tuples containing the same data may not match due to differences in the tuple structure
  - e.g., two tuples representing the same information may not match based on a different syntax adopted

Tuple-based Models for KIE II

Two lines of extension in the literature

- Exploiting tuple-based coordination within a middleware for KIE
  - e.g., [Tolksdorf et al., 2008] experiments with a tuple-based coordination within Semantic Web middleware
  - e.g., [Nixon et al., 2008] survey similar approaches
- Enhancing the tuple space abstraction with a semantic interpretation
  - e.g., [Nardini et al., 2010] extend tuple spaces with a description logic framework so as to equip each tuple, template, and operation over tuple spaces with a well-founded semantic interpretation

Self-organising Semantic Coordination (SOSC) I

Adding semantics to tuple spaces

- Generalisation of the basic principles and mechanisms of coordination and self-organisation for application to knowledge-intensive environments
- Everything still based on tuples and tuple spaces
- Now equipped with a semantic interpretation

→ Exploring a notion of self-organising semantic coordination

Self-organising Semantic Coordination (SOSC) II

SOSC

- SOSC as the management of interactions in knowledge-intensive systems
- where
  - interactions are local and involve sharing and processing of knowledge
  - the global desired effects of coordination over distributed knowledge appear by emergence and through self-organisation.
- Coordination middleware – in particular, tuple-based ones – should be adopted to support self-organising semantic coordination
  - as in the case of eternally adaptive service ecosystems for pervasive computing [Viroli and Zambonelli, 2010].

Computational Logics & SOSC: An Experiment I

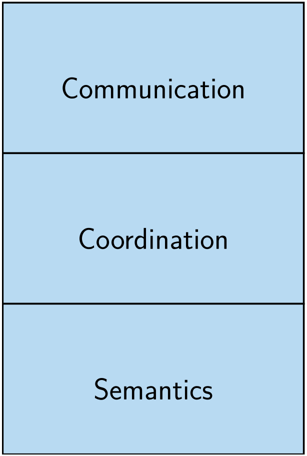
Semantic TuCSoN [Nardini et al., 2011]

- Based on logic tuples and tuple spaces
- Tuple centers equipped with a semantic interpretation
- In Semantic TuCSoN, a tuple centre is a three-space abstraction with three CL-based languages
  - Prolog ordinary tuples for communication
  - ReSpecT specification tuples for behavior specification
  - OWL DL TBoxes and ABoxes for ontology

Computational Logics & SOSC: An Experiment II

Semantic tuple centres

FOL for Communication (Prolog)
FOL for Coordination (ReSpecT)
DL for Semantics (OWL DL)



Outline

The Future of Coordination Models & Technologies I

Impact of coordination models

- Coordination models, languages, technologies and infrastructures are going to deeply impact on the engineering of complex systems
- Also in terms of methodologies and software processes
- and on related research as well

Challenges for coordination

- A huge number of technical challenges are waiting for the development of coordination middleware and infrastructures
- Such challenges will put the effectiveness of coordination-based approaches to test against many complex, real-world application scenarios

The Future of Coordination Models & Technologies II

Some of the main issues for coordination in complex systems

- Integration of organisational and security models in the coordination setting
- Full development and testing of nature-inspired coordination models
- Definition of knowledge-oriented coordination models and languages embodying international standards
- Construction of light-weight coordination technologies for pervasive scenarios
- Design of rich coordination frameworks providing developers with tools for the engineering of the interaction space in complex computational systems
- ...

The Future of CL for Coordination of Complex Systems

Goals of computational logics

- Injecting intelligence *pervasively* within complex distributed system...
- ... by providing models and technologies for coordinated components, coordination medium, communication languages and knowledge representation
- Providing a basis for formal verification of properties in complex systems

Non-compositional composition: a vision

- Complex systems obtained by composing a number of (pervasively) distributed *services* and *agents* connected through *coordination artifacts* within *knowledge-intensive environments*...
- ... most of which, logic-based ones

Self-aware Pervasive Service Ecosystems

SAPERE

- European Project FP7 – 2010-2013[a]
  - http://www.sapere-project.eu
  - http://apice.unibo.it/xwiki/bin/view/SAPERE/
- Under the hat of the Proactive Initiative AWARENESS
  - http://www.aware-project.eu/
- Based on chemical coordination for pervasive computing
- LSA (Live Semantic Annotation), as chemical tuples representing individuals, components, services in pervasive scenarios, and triggering eco-laws governing self-organisation of pervasive services

[a] This work has been supported by the EU-FP7-FET Proactive project SAPERE – Self-aware Pervasive Service Ecosystems, under contract no.256873

Thanks to. . .

- Everybody here for listening
- Fabio Fioravanti for inviting me
- Enea & Noè for being here with me!

Bibliography I

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.

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

Bravetti, M., Latella, D., Loreti, M., Massink, M., and Zavattaro, G. (2009).
Combining timed coordination primitives and probabilistic tuple spaces.
In Kaklamanis, C. and Nielson, F., editors, *Trustworthy Global Computing*, volume 5474 of *LNCS*, pages 52–68. Springer.
4th International Symposium (TGC 2008), Barcelona, Spain, 3-4 November 2008, Revised Selected Papers.

Brogi, A. and Ciancarini, P. (1991).
The concurrent language, Shared Prolog.
ACM Transactions on Programming Languages and Systems (TOPLAS), 13(1):99–123.

Bibliography II

Busi, N., Ciancarini, P., Gorrieri, R., and Zavattaro, G. (2001).
Coordination models: A guided tour.
In [Omicini et al., 2001], chapter 1, pages 6–24.

Cabri, G., Leonardi, L., and Zambonelli, F. (2000).
MARS: A programmable coordination architecture for mobile agents.
IEEE Internet Computing, 4(4):26–35.

Casadei, M. and Omicini, A. (2009).
Situated tuple centres in ReSpecT.
In Shin, S. Y., Ossowski, S., Menezes, R., and Viroli, M., editors, *24th Annual ACM Symposium on Applied Computing (SAC 2009)*, Honolulu, Hawai'i, USA. ACM.

Corkill, D. (1991).
Blackboard systems.
Journal of AI Expert, 9(6):40–47.

De Nicola, R., Ferrari, G., and Pugliese, R. (1998).
KLAIM: A kernel language for agent interaction and mobility.
IEEE Transaction on Software Engineering, 24(5):315–330.

Bibliography III

Denti, E. and Omicini, A. (1999).
Designing multi-agent systems around a programmable communication abstraction.
In Meyer, J.-J. C. and Schobbens, P.-Y., editors, *Formal Models of Agents*, volume 1760 of *LNAI*, pages 90–102. Springer-Verlag.
ESPRIT Project ModelAge Final Workshop. Selected Papers.

Freeman, E., Hupfer, S., and Arnold, K. (1999).
JavaSpaces Principles, Patterns, and Practice: Principles, Patterns and Practices.
The Jini Technology Series. Addison-Wesley Longman.

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.

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.

Bibliography IV

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.

Minsky, N. H. and Ungureanu, V. (2000).
Law-Governed interaction: A coordination and control mechanism for heterogeneous distributed systems.
ACM Transactions on Software Engineering and Methodology (TOSEM), 9(3):273–305.

Murphy, A. L., Picco, G. P., and Roman, G.-C. (2006).
LIME: A coordination model and middleware supporting mobility of hosts and agents.
ACM Transactions on Software Engineering and Methodology, 15(3):279–328.

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.

Bibliography V

Nardini, E., Viroli, M., and Panzavolta, E. (2010).
Coordination in open and dynamic environments with TuCSoN semantic tuple centres.
In Shin, S. Y., Ossowski, S., Schumacher, M., Palakal, M., Hung, C.-C., and Shin, D., editors, *25th Annual ACM Symposium on Applied Computing (SAC 2010)*, volume III, pages 2037–2044, Sierre, Switzerland. ACM.

Nixon, L., Simperl, E., Krummenacher, R., and Martin-recuerda, F. (2008).
Tuplespace-based computing for the Semantic Web: A survey of the state-of-the-art.
Knowledge Engineering Review, 23(2):181–212.

Omicini, A. (2011).
Agents writing on walls: Cognitive stigmergy and beyond.
In Paglieri, F., Tummolini, L., Falcone, R., and Miceli, M., editors, *The Goals of Cognition. Festschrift for Cristiano Castelfranchi*. College Publications, London.

Omicini, A. and Denti, E. (2001).
From tuple spaces to tuple centres.
Science of Computer Programming, 41(3):277–294.

Bibliography VI

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 Viroli, M. (2011).
Coordination models and languages: From parallel computing to self-organisation.
The Knowledge Engineering Review, 26(1).
Special Issue for the 25th Years of the Knowledge Engineering Review.

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

Papadopoulos, G. A. and Arbab, F. (1998).
Coordination models and languages.
In Zelkowitz, M. V., editor, *The Engineering of Large Systems*, volume 46 of *Advances in Computers*, pages 329–400. Academic Press.

Bibliography VII

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 *LNAI*, pages 124–140. Springer.
3rd International Workshop (E4MAS 2006), Hakodate, Japan, 8 May 2006. Selected Revised and Invited Papers.

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.

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.

Bibliography VIII

Tolksdorf, R., Nixon, L., and Simperl, E. (2008).
Towards a tuplespace-based middleware for the Semantic Web.
Web Intelligence and Agent Systems, 6(3):235–251.

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.

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., Casadei, M., and Omicini, A. (2009).
A framework for modelling and implementing self-organising coordination.
In Shin, S. Y., Ossowski, S., Menezes, R., and Viroli, M., editors, *24th Annual ACM Symposium on Applied Computing (SAC 2009)*, volume III, pages 1353–1360, Honolulu, Hawai'i, USA. ACM.

Bibliography IX

Viroli, M. and Zambonelli, F. (2010).
A biochemical approach to adaptive service ecosystems.
Information Sciences, 180(10):1876–1892.

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

Wyckoff, P., McLaughry, S. W., Lehman, T. J., and Ford, D. A. (1998).
T Spaces.
IBM Journal of Research and Development, 37(3 – Java Techonology):454–474.

Coordination Models and Technologies toward Self-Organising Systems

Andrea Omicini
andrea.omicini@unibo.it

Alma Mater Studiorum—Università di Bologna

CILC 2011
Università degli Studi "Gabriele D'Annunzio" di Chieti e Pescara
Pescara, 31st of August 2011