A Coordination Approach to Adaptive Pervasive Service Ecosystems

Mirko Viroli, Elena Nardini
ALMA MATER STUDIORUM – *Università di Bologna*
Cesena, Italy
{*mirko.viroli, elena.nardini*}*@unibo.it*

Gabriella Castelli, Marco Mamei, Franco Zambonelli
Università di Modena e Reggio Emilia
Reggio Emilia, Italy
{*gabriella.castelli, marco.mamei, franco.zambonelli*}*@unimore.it*

***Abstract*—Technology evolution is providing new pervasive service scenarios characterised by a huge number of distributed and dynamic devices. Accordingly, a new generation of services and infrastructures are emerging which support situatedness, adaptivity and diversity. In this paper we model the overall world of services, data and devices, as a distributed computational ecosystem. As such, each entity will be modelled as an autonomous, spatially-situated individual of the ecosystem, whose existence and state is reified by an *LSA* (Live Semantic Annotation). Ecosystem behaviour is controlled by coordination rules called *eco-laws*, which are chemical-like reactions evolving the population of LSAs. We describe an architecture supporting this vision along with a model of eco-laws, and show their usefulness in a scenario of adaptive pervasive displays.**

I. INTRODUCTION

The increasing evolution and spread of pervasive computing technologies is defining the basis for the emergence of a dense and global decentralised infrastructure for the creation of general-purpose pervasive services.

Such a scenario will feature a number of diverse sensing devices, personal and public displays, personal mobile devices, and humans, all of which to be dynamically engaged in very dynamic and flexible coordinated activities. In particular, such novel pervasive application scenarios call for adopting self-organising service coordination approaches supporting: *(i)* the capability to naturally match the spatial nature of the environment and of the services within, which involves managing coordinating activities between components that are physically co-located or close to each other; and *(ii)* the capability to inherently facilitate spontaneous interactions among components, without requiring an a priori knowledge of each other, and making the resulting patterns of interactions self-adaptive and self-managing.

Recent proposals in the area of coordination models and middleware for pervasive computing scenarios try to account for issues related to spatiality [10], [11], spontaneous and opportunistic coordination [1], [6], self-adaptation and self-management [15]. In most of the cases, however, such approaches propose *one-of* solutions to specific problems in specific areas, and lack generality.

To tackle the problem in a more comprehensive way, we follow the line of research of nature-inspired solutions to the implementation of specific middleware or distributed services [9]. Spatial concepts and features of self-adaptation and self-management are inherent to many natural systems because of the basic "rules of the game"—independently of the metaphor adopted (e.g., physical [10], chemical [18], biological [2] or social [8]). In short, nature accounts for a spatial environmental substrate, and autonomous individuals (i.e., components and agents) of different kinds get in touch, interact, compete, and combine with each other – in one word, coordinate – in respect of some basic "laws of nature". Accordingly, we claim that a truly self-adaptive shared pervasive substrate will have to be conceived as the space in which bringing to life an ecosystem of service components coordinated by some basic laws (which we call "*eco-laws*") of the ecosystem. The ecological approach we undertake goes beyond most of current nature-inspired studies, which typically are limited to ensembles of homogeneous components: we define a framework for supporting the vision of novel pervasive and Internet scenarios as made up of self-adaptive devices and services, that autonomously cooperate for the creation of global functionality [16]. Along these lines, the key contributions of this article are as follows:

- we frame the key concepts of the infrastructure at the basis of our coordination approach and of the SAPERE project ("Self-aware Pervasive Service Ecosystems", *www.sapere-project.eu*), (Section 2);
- we detail the model and language of eco-laws, which resemble chemical-like semantic reactions over the individuals of the pervasive ecosystem, enacted on top of the shared distributed space of interactions so as to promote spontaneous and opportunistic interactions among the various components (Section 3);
- we exemplify our approach presenting a case study of coordinated context-awareness and visualisation in an environment pervaded by interactive displays (Section 4).

Related works (Section 5) and final remarks (Section 6) conclude the paper.

II. THE SAPERE APPROACH

From the architectural viewpoint, we consider structuring a pervasive service environment as a non-layered *spatial substrate*, mapped above the actual pervasive network infrastructure. The SAPERE architecture is composed of a dynamic set of computational *nodes* spread in the pervasive computing environment, to which all the *individuals* (i.e., devices, users, software services) that take part in the ecology connect to, based on proximity.

Any individual – typically referred to as a *component*, when focussing on its computational counterpart – will have an associated semantic representation inside the ecosystem called *Live Semantic Annotation* (LSA). To account for high dynamics of the scenario and for its need of continuous holistic adaptation, LSAs are handled as living, dynamic entities, capable of reflecting the current situation and context of the component they describe. They can encapsulate data relevant to the ecology, reify events, act as actual observable interfaces of components, and ultimately be the basis for enforcing semantic and self-aware forms of dynamic interactions (both for service aggregation/composition and for data/knowledge management). In short, the interaction of any individual with the ecology is enacted *through* their LSA, by modification/observation of their structure and of the information they hold.

Each SAPERE node embeds a so-called *LSA-space*, in which self-adaptive coordination mechanisms take place so as to mediate the interaction between components. Whenever a component approaches a node, its own LSA is automatically injected into the LSA-space of that node, making the component part of that space and of its local coordination dynamics, it gets immersed in the ecosystem—similarly, when a component moves away from a node, its LSA is (eventually) automatically removed from that space. Figure 1 shows a pictorial representation of an ecosystem with 4 nodes in evidence, each hosting one display located in the physical environment, featuring some software service running in the node (e.g., visualisation services to be shown in the display), and having some people (each with her own PDA) in front of the display. Note that display, users, and services are all associated with their own LSA. SAPERE nodes are linked together based on physical or logical proximity: in each node it is possible to get a reference to other nodes in the neighbourhood, and interact with them by exchange of LSAs.

Other than the LSA-space, each node embeds the set of *eco-laws* driving and ruling the activities of the ecosystem. They define the basic policies to rule sorts of virtual *chemical reactions* among LSAs, thus enforcing dynamic concept-based (i.e., semantic and goal-oriented) networking, composition, and coordination of data and services. We will consider data and services (as represented by their associated LSAs) as sorts of reagents in an ecology in which interactions and composition occur via chemical-like reactions working based on pattern-matching between LSAs. Such reactions can contribute to: *(i)* establish virtual chemical bonds between entities (e.g., a display is connected to the PDA of all users standing in front of it), *(ii)* produce new LSAs (e.g. some data to log the events occurring in a given context), *(iii)* diffuse some LSA in the network (e.g. making an event be observable in a whole region of the network). Eco-laws represent the basic patterns of interactions between components of the ecosystem. In order to provide complex (cognitive, intelligent) coordination functionalities, such as situation recognition, planning or orchestration, the SAPERE architecture also accounts for software agents that run inside a node as part of the infrastructure. They offer services available inside the node itself through their LSA as usual, injected in the ecology to improve its overall functionality. In the left-bottom node in Figure 1 an eco-law (blue cloud) is depicted in action, which properly combines the LSA of a display, of a user nearby, and of a visualisation service; the right-bottom node shows instead what could be the outcome of the eco-law, establishment of a bond between display and service (e.g., a communication channel for data to be shown) and a bond between display and user (e.g., to let the former inspect the user's profile).

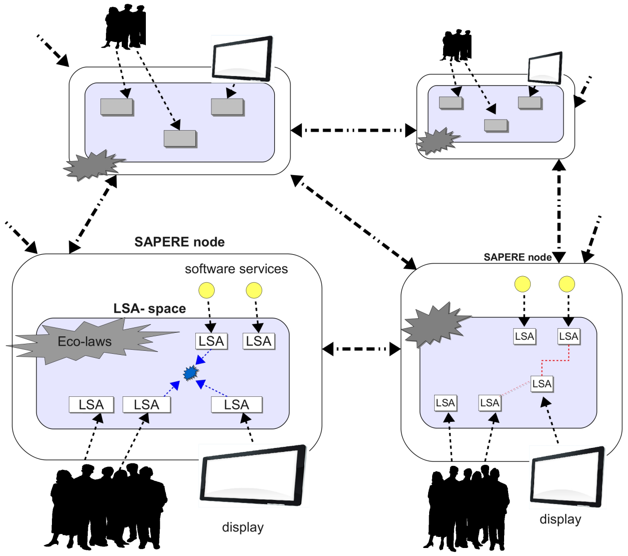

Figure 1. A representation of SAPERE logic architecture

Note that self-adaptivity and self-organisation in the SAPERE framework are not bound inside the capability of individual components, but are rather meant to emerge in the overall dynamics of the ecosystem, by proper design of eco-laws. This is achieved by the fact that any change in the system (as well as any change in its components, as reflected by dynamic changes in their LSA) will reflect in the firing of some eco-law possibly leading to the establishment of new bonds and/or in the breaking of some existing bonds. In short, we promote the philosophy: "any component is an LSA, any interaction is by eco-laws".

III. THE ECO-LAWS FRAMEWORK

To define in a more precise way ecosystems behaviour, and shade light to some key details of the SAPERE coordination model, in this section we introduce an incarnation of the eco-laws language, focussing on issues including syntactic structure of LSAs and eco-laws, application of eco-laws to LSAs (and related matching mechanisms), bond establishment, and management of locality and topology. For the sake of readability and space, we present the language informally[1].

Live Semantic Annotations

In the SAPERE framework, LSAs are *semantic* annotations expressing information with same expressiveness as standard frameworks like RDF: we here focus on the abstract syntax

$$\mathtt{i:[p_1{=}v_1,..,p_n{=}v_n]}$$

where `i` is the unique (ecosystem-wide) LSA identifier, $\mathtt{p}_i$ is the name of a property, and $\mathtt{v}_i$ is the associated value (any atomic or structured data)— $\mathtt{p}_i$ need not be different, as some property can be multi-valued. Constraints to the type of values associated to a property, or cardinality of a property, can be specified and enforced via OWL-like ontologies as in [13], though we do not focus on these aspects in this paper.

Some values/properties (starting with symbol "#") are related to a special management by the infrastructure, automatic reification of some aspect of the environment inside the LSA-space. First of all, each LSA features the (single-valued) property `#location`, associated to the identifier of the SAPERE node it is stored in. Secondly, each LSA-space maintains: an LSA of kind i`:[type=#time,value=`t`]` where t is a numeric representation of current time; and one LSA of kind i`:[type=#neighbour,where=`id`,distance=`d`]` per neighbouring LSA-space, carrying the neighbour identifier id and its estimated distance d.

Eco-laws

An eco-law is a chemical-resembling reaction working over patterns of LSAs. An LSA pattern P should be initially understood as an LSA which may have some variable (written inside a pair of curly brackets) in place of one (or more) values, and an LSA L is said to match the pattern P if there exists a substitution of variables to values that applied to P gives L. An eco-law is hence of the kind

$$P_1 + \ldots + P_n \overset{r}{\mapsto} P'_1 + \ldots + P'_m$$

where: *(i)* the left-hand side (reagents) specifies patterns that should match the LSAs $L_1, \ldots, L_n$ to be extracted from the space; *(ii)* the right-hand side (products) specifies patterns of LSAs which are accordingly to be inserted back in the space; and *(iii)* rate expression r is a numerical positive value indicating the average frequency at which the eco-law is to be fired. In other words, we model execution of the eco-law as a CTMC transition with Markovian rate (average frequency) r. When the rate expression is omitted, the transition is expected to occur as soon as possible.

An eco-law is applied as follows: *(i)* iteratively, one reagent pattern P_i is non-deterministically extracted from the eco-law, a matching LSA L_i is found, and the resulting substitution is applied to the remainder of the eco-law; *(ii)* when (and if) iteration is over, products form the set of LSAs to be inserted back in the space — the only variable they could still contain is automatically bound to a fresh LSA identifier. For instance, eco-law

```
{a}:[ctx={c},p=null] + {b}:[ctx={c},connect=true]
-->
{a}:[ctx={c},p={b}] + {b}:[ctx={c},connect=true]
```

could be applied to LSAs `1001:[ctx=c1,p=null]` and `1002:[ctx=c1,connect=true]` (with substitution `a/1001`, `b/1002`, `c/c1`), transforming the former LSA into `1001:[ctx=c1,p=1002]`, and leaving the latter unchanged. Roughly, this eco-law can be used to take two LSAs in the same context (or condition) `c`, and fill the dangling bond `p` of the former with the identifier of the latter. Other than by references, an additional way of creating a connection between two LSAs is by a channel, which will be used in next section to make two components exchange data: special values `#sink` and `#source` can be used into an eco-law to represent the endpoints of a new channel and store them into properties of the two LSAs to be connected.

As far as topology is concerned, the framework imposes that an eco-law applies to LSAs belonging to the same space, and constraints products to be inserted in that space or in a neighbouring one (to realise space-space interaction) — these aspects can be controlled by the `#location` property described above.

Matching

Patterns in eco-laws are meant to be partial specifications, hence: *(i)* reagent patterns do not need to specify all property/value associations, but only those needed to filter the required LSAs; *(ii)* product patterns need only mention the property/value associations that are to be changed. Hence, the following eco-law

```
{a}:[ctx={c},p=null] + {b}:[ctx={c},connect=true]
-->
{a}:[p={b}] + {b} + {b′ clones b}:[connect=false]
```

works exactly as the one previously seen but with two differences: first, reagent LSAs can have a possibly high number of properties (e.g., only `ctx` and `p` are mandatory for LSA `a`); second, a new LSA `b′` is created which, due to modifier `clones`, is identical to `b`, but with `connect` property set to `false`.

Additionally, while in the above examples operator "=" is the only one used to connect a (single-valued) property to a

---

[1] A formal description of a significant fragment of the presented language, associating an operational semantics to eco-laws based on Continuous-Time Markov Chains, is presented in the companion paper available at `http://www.ingce.unibo.it/˜mviroli/SASO-full.pdf`.

value, others working on multi-valued operators can be used, which are described in turn. Operator "$=\star$" accepts on the right a variable to be bound to the *set of all* values associated to the property on the left. Operator "`has`" accepts on the left a property p and on the right a value v, and filters those LSAs whose values associated to p include v — operator "`has-not`" is dual. Operator "+=" ("−=") can be used only in product LSAs: it accepts a single value on the left, and its effect is to add (remove) it from those associated to the property. As an example, the following eco-law

```
{a}:[rs=*{X},rs has {x}] + {b}:[r={x}]
-->
{a}:[rs-={x},rs2=*{X}]
```

takes LSAs `1001:[rs=1,rs=2,rs=3]` and `1002:[r=2]`, transforms the former into `1001:[rs=1,rs=3,rs2=1,rs2=2,rs2=3]` and removes the latter.

A final mechanism we introduce is devoted to add to the eco-law language the ability to perform simple computations and/or (semantic) matching operations. A variable `x` can be annotated by the syntax {`x: x` fp exp}, where fp is a fuzzy predicate and exp is an expression. Examples of this construct include: {`x: x >= 10`}, filtering numeric values greater or equal to 10; {`x: x is a+b`}, filtering numeric values equal to the sum of variables `a` and `b`; and finally {`x: x matches y`}, filtering values matching with variable `y`. The idea of using using fuzzy predicates, which yields a factor between 0 and 1 instead of yes/no, is that the decision about filtering is probabilistic – the higher the factor, the higher the probability of filtering – which can be used to support the expressiveness as fuzzy matching [12]. In particular, operator `matches` can be used to perform matches such as, e.g., to check whether the profile description of a user could match the content of a visualisation service — relying on standard Web Ontologies as in [13].

IV. THE ADAPTIVE DISPLAYS USE CASE

Let us consider the application scenario of a public area (a mall, an airport) where some people wander around with their portable device (e.g. a smartphone). Other than user preferences, such devices embed a number of sensors that may be used for inferring what the user is doing (e.g., the microphone can be used for inferring if the user is talking with someone, the accelerometer if the user is standing or walking, and so on). A number of public displays are deployed to show different kinds of visualisation services to users, e.g. news, advertisements, directions. Such displays have to acquire data about the users standing in front of them, and use it to provide the best possible content.

This scenario can be addressed in the SAPERE framework as follows. SAPERE nodes hosting LSA-spaces are meant to be spread in the environment in a way such that when a user is in front of a display there is an LSA-space holding the LSA of the public display (specifying the display characteristics), the LSA of the user (its profile and other information obtained by sensors), and the LSA of the visualisation services that the display can show, as depicted in Figure 1. This displacement of nodes can be obtained in different ways, e.g. one node per display.

We here incrementally describe a set of eco-laws which can be used to enact several self-adaptive coordination patterns. The goal is to make public displays able to adapt the information they visualize to users around (either by an internal choice, or by leveraging an external recommender).

A. Display context-aware visualisation

We initially consider the case in which a display gathers information about nearby users and available services, accordingly decides to visualise a given service, and is consequently connected to that service by the infrastructure.

Eco-law [CTX-USR] takes (the LSA of) a display `d` with property `contextualising` set to `true`, and (the LSA of) a user `u`, and – when applied – includes `u` in the (multi-valued) `context` of `d`: when the display's agent manifests the intention to gather contextual information, `d` is linked to any user who is currently available near the display. Similarly, eco-law [CTX-SER] bonds the display's LSA to that of any visualisation service: in particular, a service is added only if the required screen properties `p'` match with the available screen properties `p`—e.g. with respect to screen resolution. Note that: *(i)* if a user or a visualisation service would at some point become unavailable in current context, the infrastructure automatically garbage collects their LSA, and eventually dropping any reference to them from other LSAs, so as to undo the effect to the above eco-laws; *(ii)* the Reaction Manager into the LSA-space never schedules eco-laws whose overall effect is void, hence the above eco-laws do not keep connecting a display with e.g. the same user over time; and *(iii)* the display's agent can decide whether contextualisation is always enabled or not, by simply updating LSA property `contextualising`.

The decision about which service is to be visualised is enacted by changing the display property to `ready`—we shall now assume that the display's agent does so, after having read all the necessary information into the LSAs referenced in the `context` property. Properties `showService` and (multi-valued) `showUser` will at that point hold information about which service `s` to visualise and what users `U` this is directed to. Eco-law [ACT] retrieves the service content semantic description `c` and current time `tt`, and accordingly connects display and service by a newly created channel (the source is on the service, the target on the display), moves display's status to `showing`, sets `showTime` property, and finally creates a log LSA keeping track of this event. As this eco-law is executed, the display's agent intercepts the new status by observation, and interacts with the service

Display context-aware visualisation:

```
{d}:[type=display, contextualizing=true]  +  {u}:[type=user]
-->[CTX-USR-IN]
{d}:[context+={u}]  +  {u}

{d}:[type=display, contextualizing=true, screenprops={p}]  +
{s}:[type=service, screenprops={p’:p’ matches p}]
-->[CTX-SER]
{d}:[context+={s}]  +  {s}

{d}:[type=display, status=ready, showService={s}, showUser=⋆{U}]  +
{s}:[content={c}]  +
{t}:[type=#time, value={tt}]
-->[ACT]
{d}:[channel=#sink, status=showing, showTime={tt}, showContent={c}]  +  {s}:[channel+=#source]  +  {t}  +
{l}:[type=log, time={tt}, service={s}, display={d}, showUser=⋆{U}]
```

Relying on a recommendation:

```
{d}:[type=display, status=ask]  +  {r}:[type=recommender,
question={q’:q’ matches service-request}] -->[ASK]
{d}:[status=asked]  +  {r}:[pending+={d}]

{d}:[type=display, status=asked]  + {r}:[type=recommendation,
device={d}, answerService={s}, answerUser=⋆{U}, answerState={ss}]
-->[REP] {d}:[status=ready, showService={s}, showState={ss},
showUser=⋆{U}]
```

Figure 2. Eco-laws for the Adaptive Displays Use Case

agent through the channel until visualisation is over: at that point it is in charge of changing the state back to `ready` as soon as a new decision about the next visualisation has been made. Finally, note that a service can concurrently hold a number of channels towards different displays available in the current space—as e.g. in the case of several adjacent displays showing the same ad.

B. *Relying on a recommendation*

The reasoning necessary for a display agent to take proper choices regarding which visualisation service better matches current context (users, services) can be quite complex, and involve techniques of situation identification which it does not likely incorporate [4]. In SAPERE, it is expected that intelligent agents able to perform this kind of reasoning are deployed separately to improve the quality and overall performance of the ecosystem. The display could be willing to externalise the above decision to a recommender agent possibly available through the current space. If this is the case, display property `status` is moved by the display agent to `ask`: eco-law [ASK] then fires which seeks for an available recommender agent (one supporting a matching recommendation service as written in the property `question`), and adds display reference to multi-valued property `pending` in the recommender's LSA.

At that point, the recommender agent (which might handle many recommendations at once) can start reading the display LSA and its context, reason about them, and eventually formulate the suggestion of visualising service `s` in state `ss` to users `U`—we shall assume a visualisation can be started in different ways, called states. Such a suggestion is written into a recommendation LSA, after which eco-law [REP] becomes applicable, causing the display LSA to reflect the decision as if it were internally taken, so that eco-law [ACT] seen above can fire.

V. RELATED WORK

The issue we face in this article can be framed as the problem of finding the proper coordination model for enabling and ruling interactions of pervasive services.

We take as ground the archetypal LINDA model [7], which simply provides for a blackboard with associative matching for mediating component interactions through insertion/retrieval of tuples.

In a sense, the SAPERE approach can be seen as a disciplined use of tuple spaces, constraining which and how tuples are to be injected/read/removed, and prescribing the use of sort of "Reaction Manager Agents" to enact eco-laws.

More specifically, we followed the idea of engineering the coordination space by some laws *inside* tuple spaces, following the pioneer works of approaches like TuCSoN [14].

Our proposal applies this idea adopting a bio-inspired approach by chemical-like semantic reactions.

From that viewpoint, this work was inspired by the chemical tuple space model in [17], though with some notable differences: *(i)* here we provide a detail notational framework to flexibly express eco-laws that work on patterns of LSAs and affect their properties; *(ii)* the chemical concentration mechanisms proposed in [17] to exactly mimic chemical dynamics is not mandatory here—though it can be achieved by a suitable design of rate expressions; *(iii)* the way we conceive the overall infrastructure, and relationship between agents and their LSAs goes beyond the mere definition of the tuple-space model.

Chemistry has been a source of inspiration for several other works in (distributed) computing and coordination like in the Gamma language and its extensions [3] and associated infrastructures [5].

Although they originated with the goal of writing concurrent, general-purpose programming languages, we share with them the idea of conferring a high-level, abstract, and nature-inspired character to the language used to program system behaviour.

Unlike them, our approach aims at specifically tackling coordination infrastructures for pervasive systems, which calls for dictating specific and novel mechanisms of matching, spatial diffusion, context- and spatial-awareness, and agent-LSA interaction.

VI. CONCLUSION AND FUTURE WORK

In this paper we outlined the SAPERE architecture, discussed an incarnation of the eco-laws framework for the self-adaptive, context-aware coordination of pervasive computing systems, and presented a case study of pervasive display ecosystems—the framework could be applicable also to other pervasive computing domains, including advanced adaptive and real-time traffic control, and social and augmented reality services.

The road towards the full realisation of the SAPERE framework is necessarily longer than the scope of this paper, and includes the study of a number of additional issues: *(i)* relying on standard technologies like RDF and OWL for the actual incarnation of LSAs and eco-laws; *(ii)* adopting advanced spatial mechanisms into eco-laws, to support higher-level self-organisation structures of LSAs.

Acknowledgments: This work has been supported by the EU-FP7-FET Proactive project SAPERE – Self-aware Pervasive Service Ecosystems – under contract no. 256874.

REFERENCES

[1] M. Autili, P. Benedetto, and P. Inverardi. Context-aware adaptive services: The plastic approach. In *FASE '09: Proceedings of the 12th International Conference on Fundamental Approaches to Software Engineering*, pages 124–139, Berlin, Heidelberg, 2009. Springer-Verlag.

[2] O. Babaoglu, G. Canright, A. Deutsch, G. A. D. Caro, F. Ducatelle, L. M. Gambardella, N. Ganguly, M. Jelasity, R. Montemanni, A. Montresor, and T. Urnes. Design patterns from biology for distributed computing. *ACM Trans. Auton. Adapt. Syst.*, 1(1):26–66, 2006.

[3] J.-P. Banâtre and T. Priol. Chemical programming of future service-oriented architectures. *Journal of Software*, 4:738–746, September 2009.

[4] P. D. Costa, G. Guizzardi, J. P. A. Almeida, L. F. Pires, and M. van Sinderen. Situations in conceptual modeling of context. In *Tenth IEEE International Enterprise Distributed Object Computing Conference (EDOC 2006), 16-20 October 2006, Hong Kong, China, Workshops*, page 6. IEEE Computer Society, 2006.

[5] P. Couderc and M. Banatre. Ambient computing applications: an experience with the spread approach. In *International Conference on System Sciences*, Hawaii, USA, 2003.

[6] C.-L. Fok, G.-C. Roman, and C. Lu. Enhanced coordination in sensor networks through flexible service provisioning. In J. Field and V. T. Vasconcelos, editors, *Coordination Languages and Models*, volume 5521 of *LNCS*, pages 66–85. Springer-Verlag, June 2009.

[7] D. Gelernter. Generative communication in linda. *ACM Trans. Program. Lang. Syst.*, 7(1):80–112, 1985.

[8] M. P. Locatelli, M. Loregian, and G. Vizzari. Artificial Societies in a Community-Based Approach to Ambient Intelligence. *The Computer Journal*, 53(8):1152–1168, 2010.

[9] M. Mamei, R. Menezes, R. Tolksdorf, and F. Zambonelli. Case studies for self-organization in computer science. *Journal of Systems Architecture*, 52(8):443–460, 2006.

[10] M. Mamei and F. Zambonelli. Programming pervasive and mobile computing applications: The tota approach. *ACM Trans. Softw. Eng. Methodol.*, 18(4):1–56, 2009.

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

[12] E. Nardini, A. Omicini, and M. Viroli. Description spaces with fuzziness. In *26th Annual ACM Symposium on Applied Computing (SAC 2011)*, volume II, pages 869–876. ACM, 21–25 Mar. 2011.

[13] E. Nardini, M. Viroli, and E. Panzavolta. Coordination in open and dynamic environments with TuCSoN semantic tuple centres. In *25th Annual ACM Symposium on Applied Computing (SAC 2010)*, volume III, pages 2037–2044. ACM, 22–26 Mar. 2010. Awarded as Best Paper.

[14] A. Omicini and F. Zambonelli. Coordination for Internet application development. *Autonomous Agents and Multi-Agent Systems*, 2(3):251–269, Sept. 1999.

[15] P. V. Roy, S. Haridi, A. Reinefeld, J.-B. Stefany, R. Yap, and T. Coupaye. Self-management for large-scale distributed systems: an overview of the selfman project. In *Formal Methods for Components and Objects, LNCS No. 5382*, pages 153–178. Springer Verlag, 2008.

[16] M. Ulieru and S. Grobbelaar. Engineering industrial ecosystems in a networked world. In *5th IEEE International Conference on Industrial Informatics*, pages 1–7. IEEE Press, 2007.

[17] M. Viroli and M. Casadei. Biochemical tuple spaces for self-organising coordination. In *Coordination Languages and Models*, volume 5521 of *LNCS*, pages 143–162. Springer-Verlag, June 2009.

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