Behavior Predictability Despite Non-Determinism in the SAPERE Ecosystem Preliminary Ideas

Gabriella Castelli, Marco Mamei, Alberto Rosi, Franco Zambonelli
Dipartimento di Scienze e Metodi dell'Ingegneria
University of Modena and Reggio Emilia, Italy
Email: {name.surname}@unimore.it

***Abstract*—How can we have confidence that self organizing systems actually do what we expect them to? In this position paper we overview some mechanisms at the basis of controlling and predicting the behavior of autonomous and self-organizing systems despite components' autonomy and non-deterministic behavior. In particular we focus the analysis on the SAPERE ecosystem as an exemplary model to frame the discussion. We identify three main directions with which to gain confidence on the overall system behavior: (i) confidence from layering, (ii) confidence from large numbers, (iii) confidence form the structure and dynamics of the state space. In the paper we describe this ideas and their implication in the design of self organizing applications.**

I. Introduction

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

In particular, such novel pervasive application scenarios call for adopting self-organizing service coordination approaches comprising autonomous and adaptive components to interact and coordinate with each other to provide services and applications.

A number of approaches, taking inspiration from swarm intelligent examples [9], [12], [8], try to achieve the above results by making use of a large number of simple autonomous components, that self-organize to achieve a desired application. Examples in this direction are the work on collective robotics [12], [13], autonomous and adaptive systems and distributed computing [17], [18], [6].

One of the main scientific questions in this kind of scenarios is:

How can we have confidence that self organizing systems actually do what we expect them to?

Providing convincing answers to that questions is fundamental to engineer robust and dependable systems based on the above self-organizing principles.

In this position paper we present thee main directions showing guidelines on to design self organizing applications so as to retain confidence in their behavior. In particular we identified three main mechanisms to be considered

1) **Confidence from layering.** System's reliable functionalities are realized on top of the self-aware layer. In this way the non determinism of the self-aware layer is shielded from the actual system functionalities.
2) **Confidence from large numbers.** Systems functionalities are realized on the basis of the average behavior of a large set of components. While the behavior of individual components can be erratic the overall average behavior is stable.
3) **Confidence from the structure and dynamics of the state space.** Analyzing the state space of the overall system, it is possible to identify more general mechanisms that guarantees the fulfillment of requested functionalities.

To ground the discussion we focus the analysis on the SAPERE model and middleware [17], as an exemplary self-organizing ICT system.

Despite this focus, we think that the proposed ideas are more general and could be fruitfully applied to a wider range of models and systems.

In the remaining of this paper we first present the SAPERE model in order to ground the discussion on a concrete setting. Then, in Section 3-5 we present the different approaches to obtain confidence in the behavior of the systems. Finally, Section 6 concludes discussing some research directions to exploit these ideas.

II. The SAPERE Model and Middleware

SAPERE takes its primary inspiration from natural ecosystems, and starts from the consideration that the dynamics and decentralization of future pervasive networks will make it suitable to model the overall world of services, data, and devices as a sort of distributed and spatially-situated computational *ecosystem*. However, unlike the many proposals that adopt the term ecosystem simply as a mean to characterize the complexity and dynamics of ICT systems [15], SAPERE brings the adoption of natural metaphors down to the core of its approach, by exploiting nature-inspired mechanisms (and in particular bio-chemical ones [16]) for actually ruling the overall system dynamics.

Specifically (see Figure 1), SAPERE models a pervasive service environment as a non-layered *spatial substrate*, laid above the actual pervasive network infrastructure. The substrate embeds the basic laws of nature (or *eco-laws*) that

rule the activities of the system. It represents the ground on which individuals of different species (i.e., the components of the pervasive service ecosystem) interact and combine with each other (in respect of the eco-laws and typically based on their spatial relationships), so as to serve their own individual needs as well as the sustainability of the overall ecology. Users can access the ecology in a decentralized way to use and consume data and services, and they can also act as "prosumers" by injecting new data or service components.

For the *components* living in the ecosystem, SAPERE adopts a common modeling and treatment of services, data, and devices. All "entities" living in the ecosystem will have an associated semantic representation (in the case of pure data items, the entity and its representation will coincide), which is a basic ingredient for enabling dynamic unsupervised interactions between components. To account for the high dynamics of the scenario and for its need of continuous adaptation, SAPERE will define such annotations as living, active entities, tightly associated to the component they describe, and capable of reflecting its current situation and context. Such *Live Semantic Annotations* (LSAs) will thus act as observable interfaces of resources and services, as well as the basis for enforcing semantic and self-aware forms of dynamic interactions (both for service aggregation/composition and for data/knowledge management).

For the *eco-laws* driving the dynamics of the ecosystem, SAPERE envisions them to define the basic policies to drive virtual *chemical reactions* among the LSAs of the various individuals of the ecology [2], [16]. In particular, the idea is to enforce, on a spatial basis and possibly relying on diffusive spatial mechanisms [10], dynamic networking and composition of data and services. In particular, data and services (as represented by their associated LSAs) will be sorts of chemical reagents, and interactions and compositions will occur via chemical reactions, i.e., semantic pattern-matching, between LSAs. Such reactions will contribute establishing virtual chemical bonds between entities (e.g., relating similar services with each other to produce a distributed service, or mining related data items) as well as producing new components (e.g. a composite service orchestrating the execution of atomic service components or a high-level knowledge concept derived from the aggregation of raw data items).

Adaptivity in SAPERE will not be in the capability of individual components, but rather in the overall dynamics of the ecosystem. In particular, adaptivity will be ensured by the fact that any change in the system (as well as any change in its components, as reflected by dynamic changes in their LSAs) will reflect in the firing of new chemical reactions, thus possibly leading to the establishment of new bonds and/or in the breaking of some existing bonds between components.

From an implementation viewpoint, SAPERE relies on lightweight and minimal middleware infrastructure (see Figure 2). In particular, it reifies LSAs in the form of tuples, dynamically stored and updated in a system of highly-distributed tuple spaces spread over the nodes of the network [10].

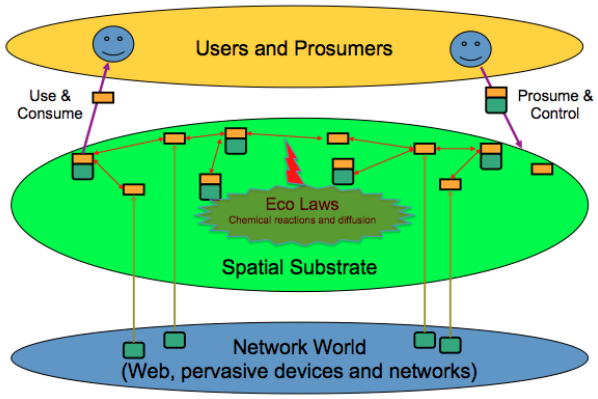

Figure 1. The SAPERE Conceptual Architecture

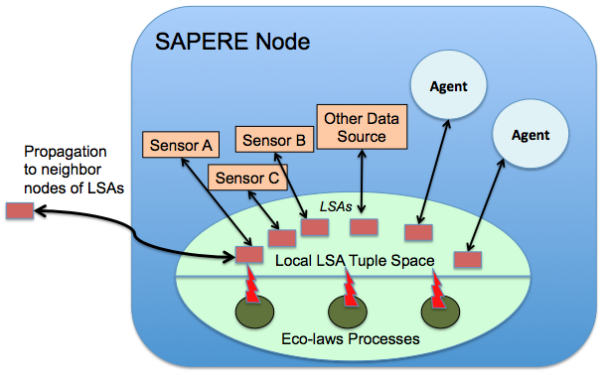

Figure 2. The SAPERE Middleware on a Node of the Network.

The active components of the ecosystem (whether services, software agents, sensing/actuating devices, or data sources) express their existence via LSAs injected in the local tuple space associated to their node. Then, they indirectly interact with each other via such tuple space by observing and accessing their own LSA.

In SAPERE an agent can see only its own LSA and the LSAs that are bonded to. There are not general read operations. An agent can inject an LSA. This LSA will form bonds with other LSAs (bond are created by means of eco-laws – see below). Only after that, that agent can read those other LSAs.

The eco-laws represent sorts of virtual chemical reactions between LSAs, and get activated by processes embedded in tuple spaces (which make SAPERE tuple spaces different from traditional tuple spaces). Such processes evaluate the potentials for establishing new chemical bonds between LSAs, the need for breaking some, or the need for generating new LSAs from the combination of existing

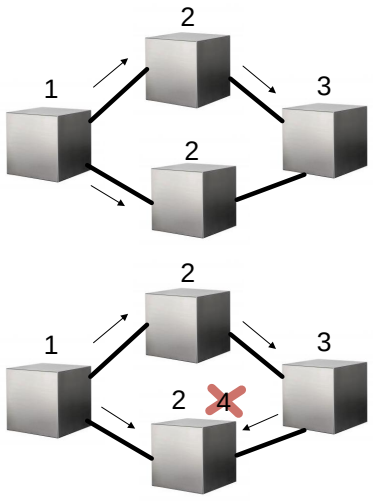

Figure 3. The gradient assumes a final coherent distribution, disregarding its unpredictable propagation.

ones. In addition, to support distributed spatial interactions, eco-laws can enforce the diffusion of LSAs to spatially close tuple spaces, e.g., to those tuple spaces that are neighbor to each other in the network, according to specific propagation patterns (gradient-based diffusion, broadcast, or multicast).

In this kind of systems, the dynamics in the tuple space tend to be rather complex, as all the interaction patterns are reified in pattern matching operations among LSAs and eco-laws.

Accordingly, the central question of this paper: *how can we have confidence that self organizing systems actually do what we expect them to?* is very relevant for this kind of systems.

In the next sections we present thee main directions showing guidelines on to design self organizing applications so as to retain confidence in their behavior: *(i)* confidence from layering, *(ii)* confidence from large numbers, *(iii)* confidence form state space analysis.

III. Confidence from layering

System's reliable functionalities are realized on top of the self-aware layer. In this way the actual system functionalities are shielded from the non determinism of the self-aware layer.

This kind of approach toward control is typical in spatial and amorphous computing [1], [10]. A number of applications in this area are built on the basis of interaction patterns arising from the creation and diffusion of gradients (a.k.a. fields) in the environment.

Gradients are distributed data structures propagated in a spatial computer and conveying spatial information about components. A typical example of the use of gradients is in crowd steering [11]. In this kind of task, gradients indicating direction to be followed are spread in the environment. Agents navigate the space by simply following the gradient uphill or downhill depending on the application.

Gradients distributed configuration can be maintained by a set of decentralized autonomous agents that propagate it in the environment. The typical process at the basis of this mechanisms is extremely non-deterministic. As the agents are not centrally coordinated nor synchronized the gradient can be propagated and maintained with different timings among different and un predictable network routes (see Fig. 3).

Is this kind of un-predictability an issue in our crowd steering applications?

It is not. Despite the unpredictability in *how* the gradient propagates, the final result is that eventually the gradient is properly laid out (see Fig. 3). In this example a reliable and dependable service: the steering gradient, is built on top on an unreliable and unpredictable substrate.

Another similar example is represented by gossip-based aggregation in sensor network [4]. In this kind of systems global aggregated values are computed in a decentralized way via the local exchange of messages among distributed nodes. For example, if nodes have to compute the average value of some sensed property over an area, they can follow this simple algorithm:

1) Each node sets its estimated average to its current sensor readings.
2) Each node selects a neighbor node. The two nodes exchange their current estimates.
3) Each node updates its estimate as the average of its current estimate and the neighbor's one.
4) Nodes cyclically repeat step 2 and 3

It is rather easy to show that this algorithm allows each estimate to rapidly converge to the actual average value [4].

Also in this case, despite interactions among devices can follow unpredictable dynamics, the final result is stable. Accordingly, application built on top of that computed average do not suffer from unpredictability in that the computed average "layer" shields the application from low-level unpredictability.

Several other examples of this same behavior can be found in [14]. In all of them, different kind of data decouples application functional requirements form the unpredictable agent dynamics that produces the data itself.

Insights for SAPERE

In SAPERE we developed a number of mechanisms to support such kind of "stable" data structures. In particular, a set of eco-laws in SAPERE act as aggregation operators. These eco-laws basically implement order and duplicate insensitive aggregation functions such as (min, max, average, etc.) [4]. These eco-laws can be used to properly propagate

and maintain a gradient LSA so that is is properly spread across the network [18]. Similarly, they can be used to aggregate distributed LSAs so as to provide a compact description of environmental properties (in term of a suitable LSA).

As discussed above, despite the dynamics of the SAPERE eco-system is highly non-deterministic, applications built on top such LSAs would be stable and predictable.

IV. Confidence from large numbers

Another complementary approach to get confidence over the behavior of the system is based on adopting a large number of components to average out unpredictability in the behavior of components.

System's functionalities are realized on the basis of the average behavior of a large set of components. While the behavior of individual components can be erratic, the overall average behavior is stable.

Algorithms and mechanisms proposed in the vision of swarm intelligence often rely on this kind of approach.

For example, ant based sorting [9] is an example of this technique. One self-organized behavior enabling this kind of sorting is that individual agents just wander randomly and pick up and drop items according to the number of similar surrounding objects. For example, if an individual agent finds a large cluster of similar items together with a different one, it will most likely pick up the misplaced item and start roaming around. That individual will probably deposit its load in a region containing other items similar to the one he is carrying. While the low level behavior of individual agents is largely erratic and non deterministic, it is possible to show that system evolves to a globally coherent state in which items are clustered.

Another example, very relevant in the context of the SAPERE vision, is related to artificial chemistry [18]. This example matches very closely the working of the SAPERE ecosystem: a large number of LSAs (metaphorically chemical components) are subject to a number of Eco-laws (metaphorically chemical reactions).

If the same eco-law can be applied to multiple LSAs, we have indeterminism, and thus unpredictability in the way in which the system will evolve.

One way to avoid this kind of situations is by relying – as in chemistry – on large (theoretically Avogadro-like) numbers of LSA. In this way, all the possible products of Eco-laws are produced and unpredictability vanishes as application designers are guaranteed that all the possible reactions will take place and all the possible products will appear.

In both the above examples it is possible to see that, if an application is built on top of such collectively-produced functionalities, then application's evolution is predictable despite the mechanisms underlying non determinism.

Insights for SAPERE

In SAPERE we developed algorithms relying on such a large-number effect [14]. These algorithms allow to transform and organize LSAs' populations in a coherent and reliable way despite underlying non deterministic pattern matching.

V. Confidence from the structure and dynamics of the state space

Thinking of the system's behavior in terms of its state space, it is possible to identify methods and mechanisms to understand how the system will evolve over time.

In particular, if we are able to identify some properties of the system than are maintained by all the possible system dynamic, then we can confidently build applications on such properties.

To ground the discussion, let's focus on the latter example in the previous section: we have indeterminism every time an eco-law can be applied to multiple LSAs. More in general, in Linda-like systems, unpredictability arises when multiple pattern matches can fire concurrently. In this case, the system will evolve differently depending on which pattern matching is triggered first.

Thinking of the state space of the system, there are two cases in which such an indeterminism does not lead to unpredictability: *(i)* the state space is modeled so that – from the application functional requirements' viewpoint – all the possible evolution of the system are the same. *(ii)* The underlying mechanisms ensure that all the possible states of the system are actually visited.

In simple terms: either the system visits only states that are indistinguishable from each other form the application viewpoint, or the systems visits all the possible states. In both the cases, unpredictability vanishes. In the following of this section, we consider the two cases separately.

A. Indistinguishable States

If the unpredictability in the system evolution involves states that are indistinguishable from the application perspective, there are not problems in controlling the system.

In the SAPERE framework, a typical example of this case is considering service-oriented scenarios. In this case, multiple services (e.g., $S1$ and $S2$) can expose via the LSA a given functionality X. Another service can express in its LSA the fact it wants to bind with X. Indeterminism in the way which eco-laws are applied does not allow the programmer to predict whether the service will bind with $S1$ or $S2$. However this is not a problem, since from the application viewpoint $S1$ and $S2$ are indistinguishable as they provide the same functionality X.

As another example, the mechanisms that lead to the diffusion of a gradient in the system can be interpreted as operations than move the system in the state space to a "point"

corresponding to the state in which the gradient is properly laid out. Because of the underlying non-determinism the system may take different trajectories to reach than point, but *eventually* the proper state will be reached. Disregarding transient behaviors, the trajectories followed by the system are indistinguishable by a "gradient-following" application – like crowd steering – that only relies on the resulting gradient.

From this viewpoint, the case of indistinguishable states actually generalizes the – confidence from layering – described in the previous section.

Insights for SAPERE

Following this kind of ideas, when creating an application in the SAPERE framework, it is important to design LSAs so that pattern matching can only happen among LSAs that are equivalent (indistinguishable) from the application perspective. In general, to achieve this property in open scenarios, it is important rely on common ontologies or namespaces in order to actually ensure the complete indistinguishability in the states of the system that can be possibly reached.

B. All States

Another condition under which unpredictability vanishes is in the case the system generates all the possible states.

To clarify this concept, let us focus on a SAPERE application in which an agent's LSA can bind with both LSA1 and LSA2. However, the two bindings are not indistinguishable (like the in the previous Section) and the agent will behave differently depending on which LSAs will be bound. Looking at Fig. 5, the agent will execute *function1* or *function2* non deterministically.

The problem is that since pattern matching is non deterministic the agent will be bound with LSA1 or LSA2 and it does not even know that the other LSA (LSA2 or LSA1) was existing. Looking at Fig. 5, the agent will be never able to execute *function3*. Recall from Section 2 that SAPERE agent cannot read the SAPERE space, they can just perceive LSAs with which they are bound.

On the contrary, if the agent could *see* that both LSA1 and LSA2 are present in the SAPERE space, then the system evolution would be predictable and specified by the agent code: looking at Fig. 5, the agent will deterministically chose *function3*.

This situation is the realm of ergodic processes and systems [19]. In signal processing, a stochastic process is said to be ergodic if its statistical properties can be deduced from a single, sufficiently long sample (realization) of the process. This implies that an ergodic process visits all the state space.

In general, pattern matching operations like in SAPERE are not ergodic as the creation of a bond between two LSAs can prevent other bonds from being created. When

```
If(bind with LSA1):
     function1()
If(bind with LSA2):
     function2()
If(there are both LSA1 and LSA2):
     function3()
```

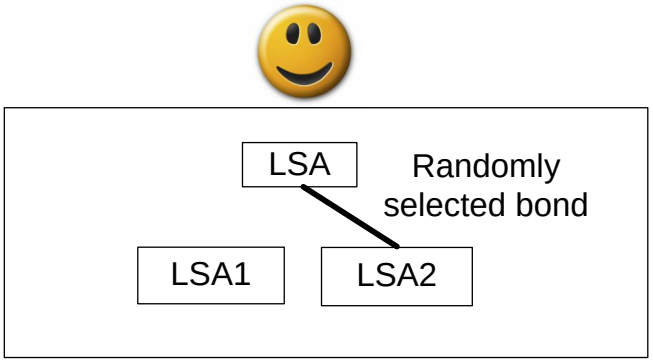

Figure 5. Since SAPERE agent cannot perform generic read operation on the SAPERE space, the agent executes non deterministically *function1* or *function2*. It can never execute *function3*. On the contrary, if the agent could *see* that both LSA1 and LSA2 are present in the SAPERE space, then it will deterministically chose *function3*

this happens, we have unpredictability in that the system randomly visit a state excluding the others (see Fig. 4.a).

To solve this issue and regain predictability the way in which the pattern matching process applies to eco-laws has to be changed. The intuition is that if the creation of a bond between two LSAs does not prevent other bonds from being created, the ergodicity is restored and the system is again predictable. Accordingly there are two possibilities:

1) For each eco-law there must exist the opposite eco-law that disrupts the bond being created. In this way, the disruption of a bond allows other kind of bonds to be realized. Thus it enables the exploration of the whole state space (see Fig. 4.b).
2) The bonding mechanisms does not prevent other bonds from happening and thus an LSA can always be bond with multiple other LSAs at the same time. Also in this case, all the reactions that could happen, actually happen and again the whole state space is visited (see Fig. 4.c).

It is possible to notice that this ergodic viewpoint generalizes the – confidence from large numbers – described in the previous section. In that case, ergodicity is simply guaranteed by the law of large numbers applied to the uniform random process that fires eco-laws to different LSAs. Also in this case all the reactions that could happen, actually happen because since the number of LSA is large the probability that one eco-law is excluded form pattern matching – because of the random schedule – goes to zero.

There is actually another definition of ergodicity that

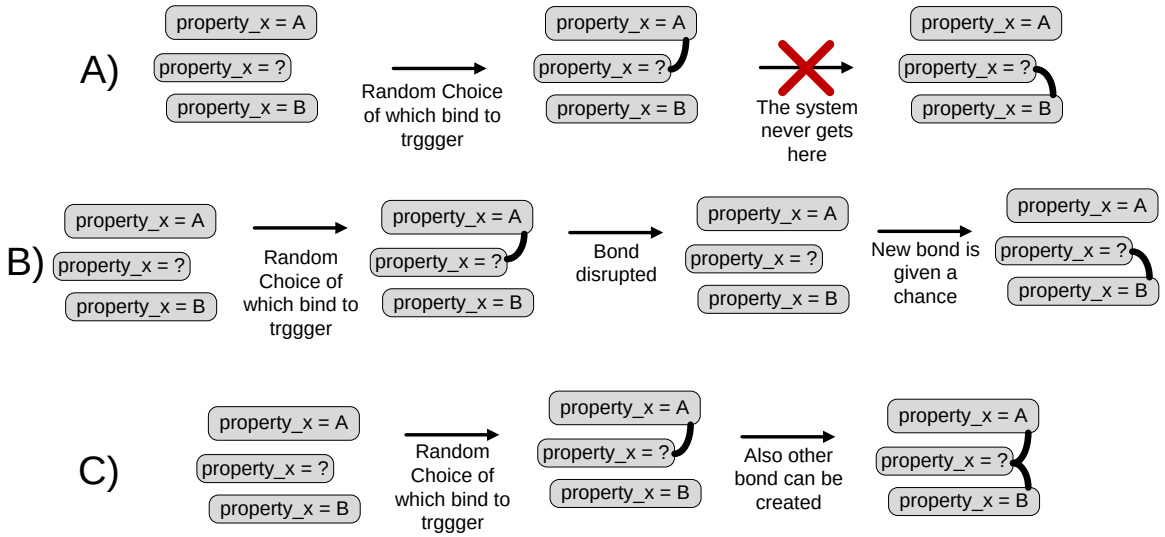

Figure 4. A) Non ergodicity in pattern matching, B) Ergodicity via bond disruption, C) ergodicity via multiple bonds

states that: *a system in which the phase-space averages correspond to the time averages is called an ergodic.*. The – confidence from large numbers – rely on phase-space average, while the previous two dynamics possibilities related to time averages.

Insights for SAPERE

Ergodicity allows a SAPERE agent to *see* the whole range of LSAs to be possibly bound. The agent will then decide how to behave on the basis of such an information. In this case, despite non determinism in the order in which pattern matching is fired, the agent is able to act deterministically by fully analyzing its context (i.e., the kind of bonds that can be established).

VI. Conclusions and Research Directions

In this position paper we tried to address one of the main challenges in the development of self-organizing applications comprising autonomous agents, namely: *How can we have confidence that systems which are self-aware and adapt according to their beliefs actually do what we expect them to?*.

We present three main research avenues on which to ground confidence on the system-level behavior of the system: *(i)* confidence via layering, *(ii)* confidence via large numbers, *(iii)* confidence from the structure and dynamics of the state space (that subsumes also the other two cases).

Al these mechanisms well apply to the SAPERE model and can have an impact on similar approaches in different areas.

In our future work we will detail and experiment the presented ideas. In particular we will run experiments using the SAPERE middleware and simulation tools to gather statistics on the expected behavior of the systems once the above control mechanisms are enforced.

In addition we will try to get better theoretical insights in the system's dynamic of SAPERE applications. In particular, we will try to apply techniques such as Petri nets [3] and model checking [7] to formally understand and describe the possible dynamics of the system.

Acknowledgements: Work supported by the SAPERE (*Self-Aware Pervasive Service Ecosystems*) project (EU FP7-FET, Contract No. 256873).

References

[1] J. Bachrach, J. Beal, and T. Fujiwara. Continuous space-time semantics allow adaptive program execution. In *IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, Boston (CA), USA, 2007.

[2] J.-P. Banâtre and T. Priol. Chemical programming of future service-oriented architectures. *Journal of Software*, 4(7):738–746, 2009.

[3] F. Bause and J. Kriege. Detecting non-ergodic simulation models of logistics networks. In *International conference on Performance evaluation methodologies and tools*, Nantes, France, 2007.

[4] N. Bicocchi, M. Mamei, and F. Zambonelli. Self-organizing virtual macro sensors. *ACM Transaction on Autonomous Adaptive Systems*, 7(1), 2012.

[5] R. Bird, W. Stewart, and E. Lightfoot. *Transport Phenomena*. Wiley, 1976.

[6] G. Cabri, M. Puviani, and F. Zambonelli. Towards a taxonomy of adaptive agent-based collaboration patterns for autonomic service ensembles. In *International Conference on Collaboration Technologies and Systems*, Philadelphia (PA), USA, 2011.

[7] E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.

[8] S. Dobson, S. Denazis, A. Fernandez, D. Gaiti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli. A survey of autonomic communications. *ACM Transactions on Autonomous and Adaptive Systems*, 1(2):223–259, 2006.

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

[10] M. Mamei and F. Zambonelli. Programming pervasive and mobile computing applications: the tota approach. *ACM Trans. Software Engineering and Methodology*, 18(4), 2009.

[11] M. Mamei, F. Zambonelli, and L. Leonardi. Co-fields: A physically inspired approach to distributed motion coordination. *IEEE Pervasive Computing*, 3(2):52–61, 2004.

[12] G. Pini, A. Brutschy, M. Frison, A. Roli, M. Dorigo, and M. Birattari. Task partitioning in swarms of robots: An adaptive method for strategy selection. *Swarm Intelligence*, 5(3).

[13] T. Schmickl, R. Thenius, C. Mslinger, J. Timmis, A. Tyrrell, M. Read, J. Hilder, J. Halloy, A. Campo, C. Stefanini, L. Manfredi, T. Dipper, D. Sutantyo, and S. Kernbach. Cocoro the self-aware underwater swarm. In *Awarenss Workshop*, Ann Arbor (MI), USA, 2011.

[14] A. Tchao, M. Risoldi, and G. Serugendo. Modeling self-* systems using chemically-inspired composable patterns. In *IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, Ann Arbor (MI), USA, 2011.

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

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

[17] M. Viroli, E. Nardini, G. Castelli, M. Mamei, and F. Zambonelli. A coordination approach to adaptive pervasive service ecosystems. In *Awarenss Workshop*, Ann Arbor (MI), USA, 2011.

[18] M. Viroli, D. Pianini, S. Montagna, and G. Stevenson. Pervasive ecosystems: a coordination model based on semantic chemistry. In *ACM Symposium on Applied Computing (SAC 2012)*, Riva del Garda, Italy, 2012. ACM.

[19] P. Walters. *An introduction to ergodic theory*. Springer, 1982.