Spray Computers: Frontiers of Self-Organization for Pervasive Computing

Franco Zambonelli[1], Marie-Pierre Gleizes[2], Marco Mamei[1], Robert Tolksdorf[3]
1) DISMI - Università di Modena e Reggio Emilia – Reggio Emilia – Italy
2) IRIT – Universitè Paul Sebatier – Toulouse – France
3) Institut für Informatik – Freie Universität of Berlin – Berlin – Germany
franco.zambonelli@unimore.it, gleizes@irit.fr, mamei.marco@unimore.it, tolk@inf.fu-berlin.de

Abstract

We envision a future in which clouds of microcomputers can be sprayed in an environment to provide, by spontaneously networking with each other, an endlessly range of futuristic applications. However, beside the vision, spraying may also act as a powerful metaphor for scenarios such as ad-hoc networks and P2P computing. In this paper we: detail the different spray computers scenarios and their applications; discuss the issues related to the design and development of spray computer applications, calling for novel approaches exploiting self-organization and emergent behaviors as first-class tools; present the key research efforts being taken in the area; try to define a research agenda.

1. Introduction

It is not hard to envision a future in which networks of micro computers will be *literally* sold as spray cans, to be sprayed in an environment or on specific artifacts and enrich them with functionalities that, as of today, may appear futuristic and visionary [11,13,18]. The number of potential applications of the scenario is endless, ranging from smart and invisible clothes, intelligent interactive environments, self-assembly materials and self-repairing artifacts.

However, the above vision of spray computers may also act as a powerful metaphor for a range of other scenarios that are already under formation. These include distributed applications in embedded, possibly mobile, ad hoc networks, as well as distributed service- and data-oriented activities on the Internet [14,15]. In fact, besides the different physical scale of the components involved and of their interactions (from micro-computers interacting within networks extending across a few meters, to Internet hosts interacting at a world-wide scale), all these types of spray computer networks raise the same challenges as far as development and deployment of applications is involved, calling for radically novel approaches to distributed systems development and management.

On the one hand, such systems require approaches enabling application deployment with zero a priori configuration effort, in which components can self-organize their application activities and self-retune their overall behavior depending on specific contingencies [8]. On the other hand, the autonomous and decentralized nature of the activities in such scenarios, together with the possibly unpredictable dynamics of the operating environments, is likely to make those systems exhibit unexpected, "emergent" behaviors, calling for methodologies to predict and control the emergence of such behaviors and possibly offensively exploit them for the otherwise impossible achievement of complex distributed tasks [3,12,19].

After having detailed a bit our vision about Spray computers (Section 2), we explore the main issues arising when exploiting self-organization for the design and development of applications for spray computers (Section 3). A short survey of the efforts being taken in this area (Section 4) identifies limitations in current researches and enables to identify a few key activities to be undertaken in the area of spray computers (Section 5).

2. The Spray Computers Vision

The concept of spray computers will soon pervade all ICT scenarios, from the micro-scale (literally spray computers), to the medium-scale (handheld and wearable computers) and the global scale (worldwide computing).

2.1 The Micro Scale

As proved in the context of the Smart Dust project at Berkeley [13], it is already possible to produce fully-fledged computer-based systems of a few mm^3, and even much smaller ones will be produced in the next few years. Such computers, which can be enriched with communication capabilities (radio or optical), local sensing (e.g., optical, thermal, or inertial) and local effecting (e.g., optical and mechanical) capabilities, are the basic ingredients of our spray computers vision.

Spray computers, as we imagine them, are clouds of sub-millimeter-scale microcomputers, to be deployed in an environment or onto specific artifacts via a spraying or a painting process. Once deployed, such components will spontaneously network with each other and will coordinate their actions to provide specific "smart" functionalities via local sensing and effecting. We imagine it will be possible, say in 2020, to go to the local store and there buy a "pipe repairing" spray, made up of a cloud of MEMS microcomputers capable of navigating in a pipeline, recognizing the presence of holes, and self-assembling with each other so as repair the pipe. Similarly, we could imagine a "spray of invisibility" [18] made up of micro devices capable of receiving and transmitting light emissions, and by interacting with each other via short-range wireless communications, of rendering invisible an object on which it is painted. Other applications one could envision include any type of self-assembly artifact [11] and MEMS-based artificial immune systems and drugs [13].

Whatever the applications one may envision, the key characteristics that will distinguish spray computer applications from traditional distributed computing systems are not the scale at which processes take place and the fact that processes are likely to be situated in a physical environment. After all: *(i)* the fact that a process executes on a micro device rather than on a high-end computer does not change it basic nature; *(ii)* distributed computing systems traditionally have to carry on their activities while being situated in a (computational) environment and have to interact with it. Rather, what we think distinguish spray computers are the facts that:

- their computational activities take place in a network whose structure derives from an almost random deployment process (as a spraying process is), and that is likely to change over time with unpredictable dynamics (due to environmental contingencies, failure of components, or simply mobility of components);
- the number of (hardware and, consequently, software) components involved in a distributed application is dramatically high and hardly controllable. It is neither possible to enforce a strict configuration of software components nor to control their behavior at a fine-grained level during execution.

Both the above characteristics compulsory call for execution models in which applications are made capable of self-configuring and self-tuning their activities in a spontaneous and unsupervised way, adapting to whatever network structure and surviving network dynamics.

2.2 The Medium Scale

Spray computing, other than a likely incoming technology, can also act as a power metaphor for describing the key characteristics of the emerging scenarios of ubiquitous and pervasive computing.

We already typically carry on two or three computers (i.e., a cell phone, a laptop, and possibly a PDA). Also, our houses are already populated by a variety of microprocessor-based furniture (e.g. TVs, phones, etc.). Very soon, all the world around us will be densely populated by personal-area networks (e.g., the ensemble of Bluetooth enabled interacting computer-based devices we could carry on or we could find in our cars), local ad-hoc networks of handheld computers (e.g., networks of interacting PDAs carried by a team that have to directly interact and coordinate with each other in an open space), and furniture networks (e.g., Web-enabled fridges and ovens able to interact with each other and effectively support our cooking activities in a coordinated way).

The above types of networks, although being formed by different types of computer-based devices (i.e., medium-end computers) and at different physical scales than literally spray computers, shares the same issues as far as the development and management of distributed applications is concerned. In fact, most of these networks will be wireless, with structures dynamically varying depending on the relatives positions of devices, all of which intrinsically mobile and ephemeral. In addition, it is not commercially viable to consider deploying applications for common everyday use if they require explicit configuration and control efforts to meet the dynamics of the operational environment. Rather, new approaches are needed to develop applications as if they were to execute on a network of spray computers.

2.3 The Global Scale

The dramatic growth in complexity and size of the Internet have raised researchers' attention to the need of novel approaches to worldwide applications management. In fact, traditional approaches requiring human configuration efforts and supervision fall short when the applications have to deal with an environment that grows and evolves in a fully decentralized way (as the Internet and the Web do), and when both nodes, data and services to be accessed are intrinsically of an ephemeral nature (consider, e.g., PDAs, wireless Internet connection, non-commercial data and services). In other words, when the network in which applications have to operate can be assimilated to a spray computers network.

The need to access data and services according to a variety of patterns and independently of the availability/location of specific servers and of the dynamics of the network have suggested the adoption of P2P approaches [14,15]. In P2P computing, instead of promoting a strict control over the execution of software components and of their interactions (an almost impossible task given the dynamics of the scenario), the

idea is to promote and support adaptive self-organization and maintenance of a structured network of logical relationships among components (i.e., an overlay network), to abstract from the physical "sprayed" nature of the actual network and survive events such the arrival of new nodes or the dismissing of some nodes.

Overlay networks are currently the most widely investigated approach to distributed application development and management in worldwide computing, and are leveraging a variety of useful applications facilitating access to (and coordination over) a variety of world-wide distributed data and services. Although self-organizing overlay networks – as they are studied today – are not necessarily the only and best approach, the attention towards them is the body of evidence of the need – also in this scenario – for new self-organizing approaches to distributed application development.

3 Programming Spray Computers

Programming a spray computer (i.e., developing applications to be deployed on networks of spray computers) means engineering pre-specified and useful global behaviors from the cooperation of a large number of unreliable parts interconnected in unknown, irregular, and time-varying ways. This translates in devising algorithms and control methodologies to let the (software running on) sprayed computing devices *self-organize* their interaction patterns and their activities towards the achievement of a global application goal without any a-priori global supervisor or centralized control .

The basic low-level mechanism upon which to rely to enable self-organization appears quite well-understood and are basically the same whatever the scale (sensor networks or P2P applications): dynamic discovery of communication partners and services via broadcasting; localization and navigation in some sorts of spatial environment, whether physical (as in sensor networks) or computational (as in P2P systems). What is still missing is an assessed understanding of how to design, develop, and manage, self-organizing applications for these kinds of systems, leading to some general purpose methodologies and programming environments.

Identifying some general and abstract solution to enable the design and development – via a proper programming of self-organizing activities – of specific global application goals, would have a dramatic impact in all sketched scenarios (micro, medium and global scale). In this paper, without having solutions at hand, we can try to identify some key directions to investigate.

3.1 Direct Engineering of Self Organization

Direct engineering approaches to self-organization basically aims at defining distributed algorithms that, starting from a few basic mechanisms (e.g., broadcast and localization), and exploiting local interactions and local computations, can provably lead a system (or parts of it) to a final coherent global state. Unlike traditional distributed algorithms, self-organizing algorithms disregard micro-level issues such as ordering of events, process synchronization, and structure of the underlying networks (issues for which no possibility of control is assumed). Rather, they focus on the fact that the algorithm will eventually converge despite micro-level contingencies, and that it will keep the system in a stable state despite perturbations (e.g., network dynamics).

A typical example of a direct engineering approach to self-organization is distributed self-localization. A number of randomly distributed particles can determine their geographical position starting from a few "beacon" particles (self-determined via leader election and acting as reference frame) and recursively applying a local triangulation mechanism to determine their position w.r.t. to close particles, until a global coherent positioning of all particles in the reference frame is reached [11].

Another example relates to the formation of regular spatial patterns in mobile particles [17]. Given a number of particles distributed in an environment, it is possible to devise distributed algorithms that, by locally driving the movements of the particles, eventually lead the system to self-organize in globally regular shapes. For instance (Figure 1): a simple leader election algorithm can determine the center of gravity of the particles; then, the center of gravity can propagate in the network with a sort of "gravitational field" attracting all other particles toward the center, ending in a circular organization of particles.



Figure 1: Sequences of direct self-organization of mobile robots into a circle

Direct engineering approaches to self-organization have the great advantage of enabling engineers to achieve "by design" a specific robust self-organized behavior. Unfortunately, such approaches are feasible only for a limited number of application needs. In fact, with direct approaches, only very simple global states of equilibrium (or, which is the same, only very regular patterns of activity) can be enforced, i.e., all those that can be modeled in simple linear terms. Engineering very complex behavior involving non-linear phenomena such as differentiation in activities, navigation and localization in complex manifolds, enforcement of complex

coordination patterns, requires facing very higher complexities. Most importantly, it requires a priori assumptions on the configuration of the system that would limit its self-organizing nature and, consequently, its degree of adaptivity and its robustness. For instance, making the particles in Figure 1 self-organize into non-symmetrical patterns by direct self-organization requires either a very complex code to be executed by particles or some particles to have some a priori information on where to go, that is, some a priori configuration efforts. For all these cases, a reverse engineering approach may be required.

3.2 Reverse Engineering of Emergent Behaviors

Reverse engineering approaches to self-organization aims at achieving complex coordinated behaviors in spray computers by recreating in spray computers (and adapting to specific application needs) the conditions to make some complex coordinated behaviors observed in other systems and in nature emerge in the computational system. In these cases, due to the complexity (and non-linearity) of the phenomena involved, engineers have no direct control on the evolution of the system, nor they can somehow prove that the system will behave as needed. Simply, they can be reasonably (i.e., probabilistically) confident that the evolution of the system will eventually lead to the desired globally coordinated behavior.

Clearly, simulations will be the workhorse of reverse engineering approaches: they will not only provide a framework on which to test the functionalities of a developed system, but they will be an integral part of the design and development process. Since the behavior of the components and of their interactions can hardly be modeled and predicted "on paper", simulations appear to be the only tool with which to have feedback on the actual working of a system. In other words, in reverse engineering approaches, the modeling phase consists in verifying via simulations the correctness of an idealized model suitable to the target scenario (e.g., a biological or social model), then to refine the model and the simulations (that also realize a prototype implementation of the model) to rend them similar to the actual scenario for which a candidate solution is needed.

In the past few years, several approaches to self-organization relying on the reverse engineering of diverse natural phenomena have been proposed in different areas and have shown their effectiveness in achieving difficult global coordination tasks. For instance, the phenomena of ant foraging [2,10] and gossiping [5] turn out to be useful to discover path to and diffuse information, respectively, in dynamic networks of spray computers.

Coming back to the example of mobile cooperative robots, to enforce the emergence of non-symmetrical patterns by getting inspiration from the biological formation of morphogenesis. There, it can be observed that differentiation from regular symmetrical patterns occur due to the contrasting forces induced by cells increasing in number and still have to adhere to each other in a limited physical space. Similarly, in mobile computational particles, one can impose a constraint on the average density of cell. This constraint, contrasting with the gravitational attracting them toward the center of gravity, forces them to organize in non-symmetrical patterns (see Figure 2).

Reverse engineering approaches to self-organization have several advantages. First, it is possible to rely on results from other disciplines to explore a variety of complex coordination phenomena to be exploited in spray computers systems. Second, once the basic mechanisms underlying an emergent behavior are understood, programming an actual system to exhibit such behavior is dramatically simple, and it reduces to programming typically simple local rules and local interactions. In addition, the resulting system is intrinsically robust and adaptive.

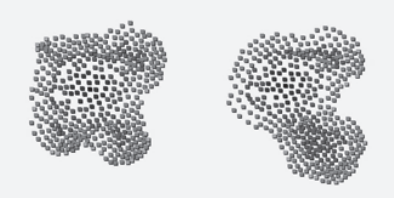


Figure 2. Emergence of non-symmetrical patterns in mobile robots

Unfortunately, approaches relying on complex emergent behavior also incur in potential drawbacks. The non-linearity involved in the evolution of the system may cause several potential final states to be reached by a system, without the possibility of predicting which ones will be actually reached after the self-organization process. In some cases, all of these states may be equivalent from the application viewpoint (e.g., in ant foraging, any reasonably short path to food/information that is found by the system is acceptable). Also, the presence of multi stable states may be sometimes advantageous, because it would ensure that the system, even if strongly perturbed, will be able to soon re-organize its activity into another stable state. However, in several other cases, the designer may wish that its system self-organizes to a specific global state, not to any one. For instance, in the case of cooperative mobile robots, specific application needs of self-assembly or of landscape, may require robot to assume a specific non-symmetric form, not any of the ones that could emerge from the self-organization process (Figure 2).

3.3 Control of Emergent Behaviors

When the evolution of a system can lead to several final global states, of which only a limited set are useful, the problem arise of how to control/direct the evolution of the system so as to ensure that it will self-organize as desired. The problem is not easy and introduce the key issue of somewhat mixing together forms of reverse and of direct engineering of self-organization.

Introducing some sorts of direct engineered control should be done without undermining the basic advantages of the reverse engineering approach, i.e., its capability to promote the spontaneous formation of complex and robust patterns of activity with little design and coding efforts. In addition, for such a mixed approach to be possible, it is necessary that both direct self-organizing algorithms and emergent behaviors are modeled and coded with the same set of basic abstractions.

In the example of cooperative mobile robots, we have successfully enforced some sort of distributed control of emergent behavior, by properly mixing the phenomena making non-symmetrical patterns emerge (described in Subsection 3.2) and sorts of leader election algorithm (as a form of direct self-organization). In particular, we have been able to let the system self-select a specific number of particles – equidistant from each other and from the center – to act as if they had a higher gravitational mass, so as to control the emergence of polygon-like structures (Figure 3) with any required number of vertices. Overall, the solution preserves both simplicity and robustness.

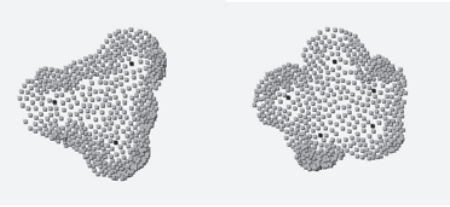


Figure 3. Controlling the emergence of specific non-symmetrical patterns in mobile robots.

Besides this example, the general problem of controlling emergent behaviors in a complex self-organizing system – which in our opinion will represent one of the key challenges for the whole research area of autonomic and self-organizing computing – is still open and widely uninvestigated. The urge for appropriate control models and for a uniform approach to direct and reverse engineering of self-organization appears even more compulsory when considering another factor. In several cases, even simple systems engineered with a direct approach may – due to simplifications in their modeling – exhibit unexpected emergent behaviors. Although sometimes such unexpected behaviors may be irrelevant or even useful and can be offensively exploited (consider, e.g., the emergence of scaling in complex networks and the advantages it carries in terms of robustness [1]), sometimes they could be damaging and would call for proper forms of control [19].

4 Relevant Research Projects

Several projects around the world are starting to recognize the above needs and are facing issues related, to different extents, to the engineering and programming of spray computers systems and applications.

Focusing on micro-scale scenarios, the Amorphous Computing project at MIT addresses the problem of identifying suitable models for programming applications over amorphous networks of simple computational particles [11]. However, the project so far has mostly focused on direct engineering approaches, and the problem of network dynamics has not been faced. The other main thread of researches in the area of micro-scale spray computers concerns sensor networks [7]. There, the key issues being investigated relate to the identification of effective algorithms and tools to perform distributed monitoring activities in a physical environment. These researches are providing good insights on the theme of self-organization and are leading to some very interesting results. However, most approaches rely on direct self-organization, while reverse engineering approach are only occasionally exploited. In addition, the accent on "sensing" – disregarding the "actuating" factor – and the key attention paid to resource and power constraints, tend to limit the research focus and undermine the identification of a wider range of additional applications and research issues.

Coming the medium scale, most of the researches are focusing either on routing algorithms for mobile ad-hoc networks of handheld computers [6] or on the definition of effective user-level ubiquitous environments [16]. Researches on routing algorithms for mobile networks share several common issues with researches on algorithms for data distribution on sensor networks. In our opinion, these works are too often focused on power and resources limitation problems and mostly disregard higher-level self-organization issues such as coordination of distributed behaviors. Researches on ubiquitous computing environments mostly focus on achieving dynamic interoperability of existing application-level components and of smart-artifact and pervasive computing devices [16]. In any case, these approaches have little to say on the issue of developing and controlling self-organizing distributed applications.

As far as the global scale is involved, most research on adaptive and unsupervised computing focus, as we have already stated, on the key idea of self-organizing

overlay networks for P2P computing, which can be considered as a typical example of a direct engineering approach to self-organization. However, we do not think this is necessarily the best and only approach. In fact, building and maintaining globally coherent overlay networks at a worldwide scale may be very costly. Thus, although simulations on small-scale systems with moderate dynamics show the feasibility of the approach, it is still unclear how this could scale to millions of nodes with very high dynamics. Next generation P2P networks should rely on flexible and light-weight approaches exploiting reverse engineering of self-organization phenomena. For instance, recent approaches based on artificial ants [2,4,5,10] – relying on overlay networks produced by the emergent collective behaviors in ants randomly roaming in the network – and computational fields [9] – enabling a sort of self-organization of components into virtual spatial shapes, i.e., mimicking in a virtual space what we have shown in this paper for mobile cooperative robots – appear both very promising.

In summary, whether the micro, medium, or global scale are involved, little is currently known about the possibility of extensively exploiting self-organization phenomena, and about the methodologies that can be devised to effectively program and control them.

5 Research Agenda

To conclude, we sketch what we consider key challenges to be faced in the area of self-organization for the design, development and control of spray computer applications.

First, researches should rely on a deeper understanding of the global behavior of a wider class of spatially distributed systems of autonomous and interacting components, other than ants and social systems. This could be used to gain a better knowledge about both the offensive exploitation of self-organization phenomena and the defense from possibly dangerous phenomena. Both cases also require the study of mechanisms and tools to somehow direct and engineering such systems in a decentralized way, so as to enforce some sorts of control over these systems despite the impossibility of controlling them in full.

Once the above understanding will be quite assessed, there will be need to define a general purpose programming model for designing and deploying applications in dynamic networks of spray computers, together with the development of associated middleware infrastructure and tools. Such a model should enable to program, deploy, and control self-organizing applications (exploiting both direct and reverse engineering approaches) with a minimal knowledge and independently of the specific application scenario, sensor networks rather than wide-area distributed applications. Approaches based on computational fields [9,17] appears very promising, enabling to uniformly model a wide variety of distributed self-organizing behaviors [10].

Eventually, all the above researches will definitely increase our understanding on the potentials of spray computers at any scale, and will likely cause a range of new application areas to come to the fore. For instance, systems such as worldwide file sharing and artifacts like the spray of invisibility could have simply never been conceived a few years ago. The new software and hardware technology will call for visionary thinkers, to unfold the newly achieved application potentials.

References

1. R. Albert, H. Jeong, A. Barabasi, “Error and Attack Tolerance of Complex Networks”, Nature, 406:378-382, 2000.
2. O. Babaoglu, H. Meling, A. Montresor, “Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems”, 22nd Conference on Distributed Computing Systems, Vienna (A), 2002.
3. C. Bernon, M.P. Gleizes, S. Peyruqueou, G. Picard, “ADELFE: a Methodology for Adaptive Multi-Agent Systems Engineering”, 3rd International Workshop Engineering Societies in the Agents World, LNAI No. 2577, 2002.
4. E. Bonabeau, M. Dorigo, G. Theraulaz, “Swarm Intelligence”, Oxford University Press, 1999.
5. D. Braginsky, D. Estrin, “Rumor Routing Algorithm For Sensor Networks”, 1ST Workshop on Sensor Networks and Applications (WSNA), 2002.
6. J. Broch, et al., “A Perfomance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols”, ACM/IEEE MOBICOM Conference, 1998.
7. D. Estrin, et al., “Connecting the Physical World with Pervasive Networks”, *IEEE Pervasive Computing*, 1(1):59-69, 2002.
8. J. Kephart, D. M. Chess, "The Vision of Autonomic Computing", IEEE Computer, 36(1):41-50, 2003.
9. M. Mamei, F. Zambonelli, “Co-Fields: a Physically Inspired Approach to Motion Coordination”, IEEE Pervasive Computing, 3(2):51-51, 2004.
10. R. Menezes, R. Tolksdorf, “SwarmLinda: a New Approach to Scalable Linda Systems based on Swarms“,ACM Symposium on Applied Computing, Melbourne (FL), 2003.
11. R. Nagpal, A. Kondacs, C. Chang, “Programming Methodology for Biologically-Inspired Self-Assembling Systems”, AAAI Spring Symposium on Computational Synthesis, 2003.
12. V. Parunak, S. Bruekner, J. Sauter, "ERIM’s Approach to Fine-Grained Agents", NASA/JPL Workshop on Radical Agent Concepts, Greenbelt (MD), 2002.
13. K. Pister, Invited Plenary Talk, 23rd International Conference on Distributed Computing Systems, Providence (RI), 2003.
14. S. Ratsanamy,, et al., "A Scalable Content-Addressable Network", ACM SIGCOMM Conference, ACM Press, 2001.
15. M. Ripeani, A. Iamnitchi, I. Foster, “Mapping the Gnutella Network”, IEEE Internet Computing, 6(1):50-57, 2002.
16. M. Roman et al., “ Gaia : A Middleware Infrastructure for Active Spaces”, IEEE Pervasive Computing, 1(4):74-83, 2002.
17. M. Vasirani, M. Mamei, F. Zambonelli, “Experiments in Morphogenesis of Simple Mobile Robots”, Applied Artificial Intelligence, 18(9-10), 2004.
18. F. Zambonelli, M. Mamei, “The Cloak of Invisibility: Challenges and Applications”, IEEE Pervasive Computing, 1(4):62-70, 2002.
19. F. Zambonelli, M. Mamei, A. Roli, “What Can Cellular Automata Tell Us About the Behavior of Large Multi-Agent Systems?”, in Software Engineering for Large Agent Systems, LNCS No. 2603, 2003.