Engineering Emergent Behaviour: A Vision

Giovanna Di Marzo Serugendo

Centre Universitaire d'Informatique, University of Geneva
24, rue Général-Dufour
CH-1211 Geneva 4, Switzerland
Giovanna.Dimarzo@cui.unige.ch

Abstract. Today's application tend to be more and more decentralised, pervasive, made of autonomous entities or agents, and have to run in dynamic environments. Applications tend to be social in the sense that they enter into communication as human people, and engage into discovery, negotiation, and transactions processes; autonomous programs run their own process, interact with other programs when necessary, but each program lives its life, and a global behaviour emerges from their interactions, similarly to what can be observed in natural life (physical, biological or social systems). Tomorrow's applications are more and more driven by social interactions, autonomy, and emergence, therefore tomorrow's engineering methods have to take into account these new dimensions. Traditional software engineering will not be adapted to this new kind of applications: they do not scale, they do not enable the definition of local behaviours and drawing of conclusions about global behaviours. The scope of this paper is to determine today's and tomorrow's application domains, where such a sociological behaviour can be observed. Starting from the observation of natural life (natural mechanisms used for self-organisation, for anonymous communication, etc), we then discuss how these natural mechanisms can be translated (or have an artificial counterpart) into electronic applications. We also consider software engineering issues, and discuss some preliminary solutions to the engineering of emergent behaviour.
Keywords: Self-organisation, emergent behaviour, swarm intelligence, software engineering.

1 Introduction

The applications of today, such as the WWW, P2P systems, or those based on spontaneous or wireless networks, have the characteristic to be decentralized, pervasive, and composed of a large number of autonomous entities such as personal assistants, and agents. They run in highly dynamic environment, where content, network topologies and loads are continuously changing. In addition, they comprise a social dimension, i.e., the entities engage interactions, discover themselves, negotiate, and perform transactions.

These characteristics are also those which one finds in the self-organising systems we can see in nature, such as physical, biological and social systems. Indeed, self-organising systems have the characteristic to function without central

control, and through contextual local interactions. Each component carries out a simple task, but as a whole they are able to carry out complex tasks emerging in a coherent way through the local interactions of the various components. These systems are particularly robust, because they adapt to the environmental changes, and are able to ensure their own maintenance or repair.

The majority of the applications of today then have certain characteristics of the self-organisation, to begin with the WWW itself, but also the Grids, P2P storage systems, e-purses, or ad-hoc routing. In certain cases, the complexity of the application is such, e.g. world scale, that no centralized or hierarchical control is possible. In other cases, it is the unforeseeable context, in which the application evolves or moves, which makes any supervision difficult. Among the applications of tomorrow, much of them will be biologically inspired: self-organising sensors networks, allowing the control of aerospace vehicles, or of dangerous zones; but also storage facilities, or operating systems facilities, which, like the human nervous system, controls in a transparent way significant functionalities [8].

There is currently an awakening that modern applications can gain (in robustness, and simplicity) if they are developed by following the principles of self-organisation which one finds in nature. To simulate and imitate nature in the electronic world constitute a first step. However, it is necessary to go beyond a simple translation of the natural paradigms. Mechanisms of interaction specific to the (electronic) applications have to be defined, as well as development methods making it possible to design components having their own local goal and whose interaction will make emerge the desired global result.

The challenges to take up in this field relate to the design, and the development of applications which "work by themselves": how to define a global goal; how to design the components and their local functionality knowing that the global goal is not a local sum of functionality; which will be the interactions between components, and how to check that the desired result will emerge during the execution. The traditional software engineering techniques are insufficient, since they are based on interfaces fixed at design time, or well established ontology. As for current methodologies, they only make it possible to define a global behaviour when it is a function of the behaviour of the various parts.

We present first self-organising systems taken from natural life, then we review some emerging self-organising electronic systems, finally we give an insight on how such applications can be engineered.

2 Self-Organising Systems

> *Self-organising systems are made of* ***many interconnected parts*** *whose* ***local interactions****, within a given* ***environment****, give rise to* ***emergent*** *properties, or behaviour, observable at the level of the global system only.*

The particularity is that the emergent properties do *not* arise out of the description of an individual component; or that the emergent global behaviour

is *not* encoded in the local behaviour of entities. Other characteristics of self-organisation include: *no central control*, the system is composed of several parts acting as peers, i.e., there is no top-down control, or top-down description; the system evolves *dynamically* with time; the system *interacts with its environment*, it modifies it and is consequently affected by its modification. More generally, the field of complex systems studies emergent phenomenon, and self-organisation [2].

Domains of natural life where we can find emerging behaviour include physical, biological and social systems.

2.1 Physical Systems

A thermodynamic system such as a gas of particles has emergent properties, temperature and pressure, that do not derive from the description of an individual particle, defined by its position and velocity. Similarly, chemical reactions create new molecules that have properties that none of the atoms exhibit before the reaction takes place [2].

2.2 Biological Systems

In biology, the human nervous system, or the immune system transparently manages vital functions, such as blood pressure, digestion, or antibodies creation. The immune system defends the body from attacks by undesired (foreign) organisms. It is made of many different kinds of cells that circulate the body looking for foreign substances. The immune system cells recognise and respond to substances called antigens: "self" antigens are part of the body, while infectious agents, recognised as "non-self" have to be eliminated[1].

2.3 Social Systems

Social insects organise themselves to perform activities such as food foraging or nests building. Cooperation among insects is realised through an indirect communication mechanisms, called stigmergy, and by interacting through their environment. Insects, such as ants, termites, or bees, mark their environment using a chemical volatile substance, called the pheromone, e.g., as do ants to mark a food trail. Insects have a simple behaviour, and none of them alone "knows" how to find food, but their interaction gives rise to an organised society able to explore their environment, find food, and efficiently inform the rest of the colony. The pheromonal information deposited by insects constitutes an indirect communication means through their environment.

Human societies use direct communication, they engage negotiation, build whole economies, and organise stock markets. Another interesting point is related to the structure of connectivity between individual human beings, also called social networks, where one can reach anyone else through a very small number of connections [12].

[1] National Institute of Allergy and Infectious Diseases, http://www.niaid.nih.gov/publications/

3 Self-Organising Applications

Nature provides examples of emergence, and self-organisation. Likewise, applications of a near future, as well as current distributed applications, by their heterogeneity, scale, dynamism, absence of central control, gain to be designed so that they organise themselves autonomously.

3.1 Self-Organising Sensor Networks

Self-organising wireless sensor networks are used for civil and military applications, such as volcanoes, earthquakes monitoring, or chemical pollution checking. Sensor networks consist of self-organised nodes, which dynamically need to set up (maybe several times) an ad-hoc P2P network, once they are deployed in a given area. They need as well to calibrate themselves in order to adapt to their environment [13]. Sensor networks benefit of recent technology enabling integration of a complete sensor system into small-size packages, as for instance the millimeter-scaled motes provided by the SmartDust project[2].

3.2 Smart Materials

Maintenance or security systems can now be part of clothes, walls, or carpets. Such electronic textile contain intertwined sensor chips or LEDs that form a self-learning and self-organising network. Applications of such smart materials include intruders detection (by pressure on a carpet); visitors guidance through a trail of LEDs; or identification of escape routes in emergency situations[3].

3.3 Autonomic Computing

Based on the human nervous system metaphor, IBM Autonomic computing initiative considers systems that manage themselves transparently wrt the applications. Such systems will then be able to self-configure, self-optimize, self-repair, and protect themselves against malicious attacks [8].

3.4 Ambient Intelligence

Ambient intelligence envisions seamless delivery of services and applications, based on ubiquitous computing and communication. Invisible intelligent technology will be made available in clothes, walls, or cars, and people can freely use it for virtual shopping, social learning, micro-payment using e-purses, electronic visas, or traffic guidance system [4]. Ambient intelligence requires low-cost and low-power designs for computation running in embedded devices or chips, as well as self-testing and self-organising software components for robustness and dependability.

[2] http://robotics.eecs.berkeley.edu/ pister/SmartDust/
[3] http://www.infineon.com

4 Engineering Emergent Behaviour

The central question related to the software engineering of self-organising applications is: how to program single agents so that, when taken as a whole, they self-organise. The engineering of self-organising applications needs means to define a global goal, and to design local behaviours so that the global behaviour emerges. This is difficult, because the global goal is *not* predictable as the sum or a function of the local goals. Consequently, the verification task turns out to be an arduous exercise, if not realised through simulation.

Traditional practice in multi-agent systems introduce basic techniques for autonomously interacting or retrieving information, such as agents coordination, service description, or ontology. However, these techniques rely on preprogammed interaction patterns, preventing adaptation to unexpected environmental changes. Current engineering practices, which directly address self-organisation, consist in designing distributed algorithms according to the social insect metaphor (e.g., digital pheromone) [3]. More recently, specific electronic interaction mechanisms are being defined, and middleware technology developed, that will help the development of self-organising applications. However, verification and whole engineering methods remain open issues.

4.1 Coordination and Control using Stigmergy

Analogies with natural life have been used and direct translation of natural mechanisms into the electronic world have already been implemented. For intrusion detection and response in computer networks [5], the immune system serves as a metaphor for detecting intruders, and the stigmery paradigm is used for responding to the attack. Mobile agents permanently roam the network in order to locate abnormal patterns of recognition. Once an attack is detected, a digital pheromone is released so that the source of attack can be located, and a response to the attack can be given.

The stigmergy paradigm serves also for manufacturing control [7]. Agents coordinate their behaviour through a digital pheromone. In order to fulfill manufacturing orders, they use mobile agents that roam the environment, and lay down pheromonal information.

4.2 Interaction Models

In addition to the digital pheromone, which is the artificial counterpart of the natural pheromone used by the ants, it is necessary to establish new electronic mechanisms directly adapted to the applications. They can be based on *tags*, a mechanism from simulation models. Tags are markings attached to each entity composing the self-organising application [6]. These markings comprise certain information on the entity (functionality, behaviour, etc.) and are observed by the other entities.

Alternatively, the *Co-Fields* model drives agents behaviour as would do abstract force fields. Agents and their environment create and spread such fields

in the environment. A field is a data structure composed of a value (magnitude of field), and a propagation rule. An agent then moves by following the coordination field, which is the combination of all fields perceived by the agent. The agents moves modify the fields, which in turn modify the agents behaviour [10].

4.3 Middleware Approaches to Self-Organisation

Acting as middleware layers, coordination spaces provide uncoupled interaction mechanisms among autonomous entities, which input data into a common tuple space, and may retrieve data provided by other entities. On top of the basic coordination environment, several enhancements have been realised in order to support self-organisation. The TOTA environment (Tuples On The Air) propagates tuples, according to a propagation rule, expressing the scope of propagation, and possible content change [9].

Anthill is a framework for P2P systems development based on agents, evolutionary programming, and derived from the ant colony metaphor. An Anthill distributed system is composed of several interconnected nests (a peer entity). Communication among nests is assured by ants, i.e., mobile agents travel among nests to satisfy requests. Ants observe their environment, and are able to perform simple computations [1].

4.4 Verification

Non-linear systems are difficult to understand because they cannot be described analytically, using equations that can help predict the system behaviour. Generally, the system is simulated through a model, and some results are obtained from the execution of the simulation.

Simulation is an essential tool to anticipate the results and to determine parameters. However, it is not sufficient to guarantee a correct result when a whole system is built using self-organising principles. Self-organisation itself ensures robustness and adaptability. In addition, one can give to the components the means of avoiding situations who could harm their correct execution. For instance, the concept of tags, explained higher, could vehicle a proof [11] and a specification, of the correct operation of a peer entity, that could be checked before interactions take place.

5 Conclusion

This paper advocates that modern and future distributed applications gain to be considered and engineered as self-organising applications. Traditional software engineering methodologies are no longer adapted to this new kind of software, which is running in highly dynamic environments, pervasive, large-scale, resource-constrained, and heterogeneous. In addition to interaction techniques, or middleware favoring self-organising behaviour, we need software engineering techniques for design, test, and verification, based on mathematical theory enabling the establishment of local goals, given the expected global behaviour [8].

6 Acknowledgements

This research is supported by Swiss NSF grant 21-68026.02.

References

1. O. Babaoglu, H. Meling, and A. Montresor. Anthill: A framework for the development of agent-based peer-to-peer systems. In *Proceedings of the 22th International Conference on Distributed Computing Systems (ICDCS '02)*, July 2002.
2. Y. Bar-Yam. *Dynamics of Complex Systems*. Perseus Books, Cambridge, MA, 1997.
3. E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Santa Fe Institute Studies on the Sciences of Complexity. Oxford University Press, 1999.
4. K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J.-C. Burgelman. Scenarios for Ambient Intelligence in 2010. Technical report, Institute for Prospective Technological Studies, 2001.
5. N. Foukia, S. Hassas, S. Fenet, and P. Albuquerque. Combining immune systems and social insect metaphors: a paradigm for distributed intrusion detection and response systems. In *5th International Workshop on Mobile Agents for Telecommunication Applications (MATA'03)*, LNCS, 2003. to appear.
6. D. Hales and B. Edmonds. Evolving Social Rationality for MAS using "Tags". In J. S. Rosenschein, T. Sandholm, M. Wooldridge, and M. Yokoo, editors, *Second International Joint Conference on Autonomous Agents and MultiAgent Systems*, pages 495–503. ACM Press, 2003.
7. H. Karuna, P. Valckenaers, C. B. Zamfirescu, H. Van Brussel, B. Saint Germain, T. Holvoet, and E. Steegmans. Self-organising in multi-agent coordination and control using stigmergy. In G. Di Marzo Serugendo, A. Karageorgos, O. F. Rana, and F. Zambonelli, editors, *First Workshop on Engineering Self-Organising Applications (ESOA'03)*, 2003.
8. J. O. Kephart and D. M. Chess. The Vision of Autonomic Computing. *Computer*, 36(1):41–50, January 2003.
9. M. Mamei and F. Zambonelli. Self-Organization in MultiAgent Systems: a Middleware approach. In G. Di Marzo Serugendo, A. Karageorgos, O. F. Rana, and F. Zambonelli, editors, *First Workshop on Engineering Self-Organising Applications (ESOA'03)*, 2003.
10. M. Mamei, F. Zambonelli, and L. Leonardi. Co-fields: Towards a unifying approach to the engineering of swarm intelligent systems. In *3rd International Workshop on Engineering Societies in the Agents World (ESAW)*, number 2577 in LNCS, pages 68–81. Springer-Verlag, 2003.
11. G. C. Necula and P. Lee. Safe, Untrusted Agents using Proof-Carrying Code. In G. Vigna, editor, *Mobile Agents and Security*, volume 1419 of *LNCS*, pages 61–91. Springer-Verlag, 1998.
12. D. J. Watts and S. H. Strogatz. Collective dynamics of small worlds networks. *Nature*, 393(6):440–442, 1998.
13. I. Wokoma, L. Sacks, and I. Marshall. Biologically inspired models for sensor network design. In *London Communications Symposium*, 2002.