



This is an author manuscript post-peer-reviewing (accepted version) of the original publication. The layout of the published version may differ .

Engineering Adaptivity, Universal Autonomous Systems Ethics and Compliance Issues

Di Marzo Serugendo, Giovanna

How to cite

DI MARZO SERUGENDO, Giovanna. Engineering Adaptivity, Universal Autonomous Systems Ethics and Compliance Issues. In: Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques. Corfu, Greece. Cham : Springer International Publishing, 2016. doi: 10.1007/978-3-319-47166-2_50

This publication URL: <https://archive-ouverte.unige.ch/unige:91743>

Publication DOI: [10.1007/978-3-319-47166-2_50](https://doi.org/10.1007/978-3-319-47166-2_50)

Engineering adaptivity, Universal autonomous systems Ethics and compliance issues

ISOLA'2016 - Panel Discussion
Position Paper

Giovanna Di Marzo Serugendo

Centre Universitaire d'Informatique,
Institute of Information Services Science,
University of Geneva, Switzerland
`giovanna.dimarzo@unige.ch`

Abstract This paper summarises some of the discussion held during the panel of the ISOLA'2016 conference on whether artificial systems actually adapt to unforeseen situations and whether we master autonomous adaptive systems. We focus here on three questions: (1) What is a collective adaptive system and what are the elements to consider when engineering a collective adaptive system? (2) What type of universal autonomous system can we envision and what for? and finally (3) How are we considering and integrating ethics, trust, privacy and compliance to laws and regulations in adaptive systems?

1 What is adaptivity and how to engineer it?

Figure 1 shows a mind-map describing the different elements participating to the engineering of (artificial) collective adaptive systems [5,6]. First it requires the use of *software agents*, autonomous in their behaviour, having a common or personal goal, able to sense and act upon their environment. They may be (among others) intelligent, reactive and/or mobile [20]. They can also be embodied into physical devices such as robots, autonomous cars, or purely sitting in electronic/computing environments, e.g. auction agents, soft-bots, or personal digital assistants.

Second, for a collective adaptive system to work as a collective of agents, we need to define an *interaction mechanism*, usually coded locally in each agent. An interaction mechanism is typically a set of rules, that the agents follow and apply according to their local perceptions. By locally applying their rules, the agents as a collective entity display some emergent behaviour. These rules allow the agents to continuously adapt their behaviour to the sensed conditions and perceived changes in their environment. Here we can identify a spectrum of rules that vary in their capability (or not) to change or adapt. Most of the engineered self-organising systems today adopt interaction mechanisms based on *fixed rules*. This is typically the case for bio-inspired systems using mechanisms such as evaporation, gradient, flocking, ant foraging, etc. For instance, agents locally apply

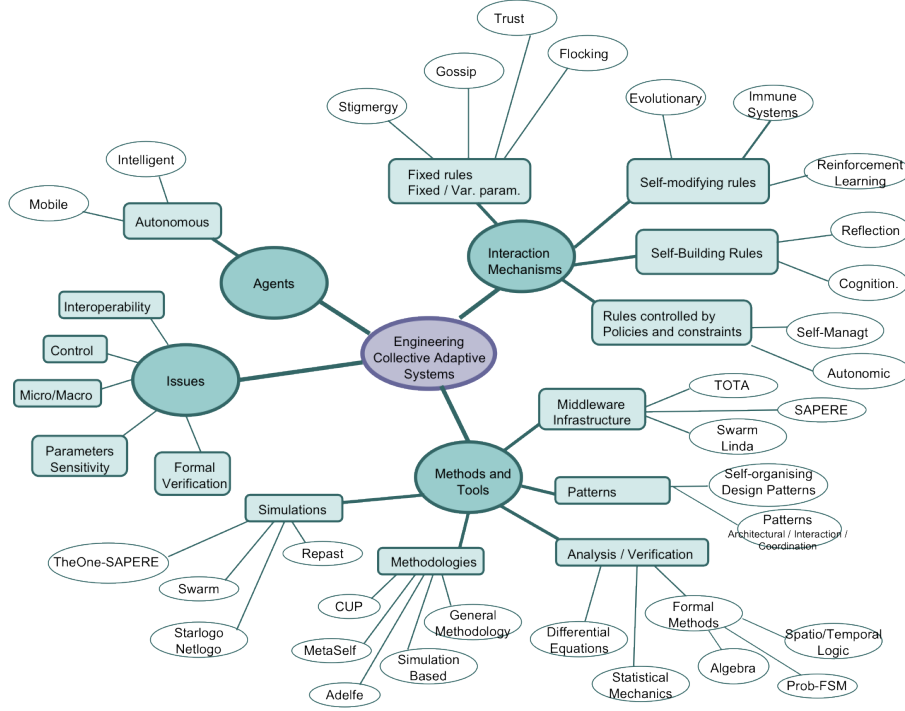


Figure 1: Engineering Collective Adaptive Systems

flocking rules, and as a collective are able to move in a coherent manner as flocks of birds. A common characteristic of collective adaptive systems is their sensitivity to parameters. Parameters are either set up (and fixed) in advance or may adapt on-the-fly, bringing increased adaptivity [8]. In the spectrum of adaptivity, we further distinguish interaction mechanisms that employ *self-modifying rules*: starting from a set of rules established at design-time, through evolutionary algorithms, learning and memory (e.g. immune systems), or reinforcement learning, the rules of each agents progressively change and modify themselves to better adapt to the agent's own observation and goal. If we move further along an adaptivity axis, we consider the next level, the case of interaction mechanisms provided by *self-building rules*. Here, agents are not provided with rules at design-time, but progressively build their own rules from scratch based on their own experiences [14,12]. Finally, we also consider the whole body of work on autonomic, self-managed systems involving explicit feedback loops [2,13] and revolving around four key activities: Monitor, Analyse, Plan, and Execute functions, also known as the MAPE architecture [4], decoupling the component that is adapted from the one that reasons and enforces the adaptation. Advanced versions of the MAPE architecture involve distribution and different variants supporting decentralised control [19]. Systems are generally designed in a top-down

manner. This is in contrast to self-organising systems that employ multiple implicit feedback loops and decentralised control, and are generally designed in a bottom-up manner.

Moving along Figure 1, the third element of the engineering of collective adaptive systems, necessary to make them trustworthy and possibly deployed on an industrial large-scale basis in everyday life, concerns the *Methods and Tools* we can use to help develop those systems, like *Middleware Infrastructures*, most of them based on blackboard deriving from the early Linda system [11], such as SAPERE or those using nature-inspired coordination [21]; *Patterns* facilitating the understanding and use of self-organising mechanisms [10]. A large body of work is provided by *Analysis and Verification* efforts, in particular the use of *Formal methods* of different kinds and recent works on spatio-temporal logics [3]. There are no actual software engineering methodologies that emerge, even though efforts are provided in this direction since several years [18]. Most of these methods are heavily based on simulations, either purely simulation tools [16] or hybrid prototyping tools [17,9].

Finally, even though recent research provided some answers to some pending issues, we still have no definite solution for formal verification of collective adaptive systems properties, in particular emergent ones; clear techniques for addressing parameters sensitivity of collective adaptive systems, or how to still remain in control of a fully self-* decentralised system once it has been deployed, and how to solve the macro to micro issue, i.e. how to engineer the local agents so that collectively they actually behave as intended.

2 What about a universal autonomous system?

There exists problems of very high complexity, such as *hyper-complex* or *wicked problems*, defined by [15] as “those that defy conventional approaches to understanding, planning, design, implementation and execution because: (1) The stakeholder interests are so diverse and divisive; (2) Interdependencies are so complex and so little understood; (3) Behaviours are so dynamic and chaotic (unpredictable)”. Wicked problems have no purely algorithmic solution and need a combination of machine processing and human-based experience and heuristics to be solved. These are problems where stakeholders have different views and understanding of the problem, and the problem itself is subject to changing constraints. This is typically the case with computational or societal problems, where human intelligence, dynamically changing data, the Internet services, networks of sensors and machines need to be combined to address them. These are problems for which we often do not know if they have a good solution, or even less how to reach a reasonable solution if it exists.

A possible vision is to develop a new type of computer, a *Social Computer* [7] - a “machine-enhanced society”. An instance of a Social Computer is a network of humans (individuals, groups) and machines (computers, data, services, sensors) able together and together only to assist experts in solving a specific large-scale (scientific or societal) problem that cannot be tackled by either computers

or humans alone. It innately integrates human abilities based on intelligence, competences and skills with machine processing power so that they complement each other. A Social Computer is a Computer because it accepts input data, can store and process it, and can produce output results. It is, however, also Social since it is based on collaboration between humans and machines. In addition and by design, it must operate in an ethical, law-abiding, correct and trustable way.

Examples of primitive social computers encompass groups of coworkers supported by computing resources, people playing massively multiplayer online games, or a single person whose activity would be supported by a network of machines. In these cases, however, the matter at hand is often not presented under the form of a problem to be solved, or the decomposition of problems into subtasks; furthermore the links between humans and/or machines are not established in any principled, problem-solving way. A lot of burden is still placed on humans to identify problems and their solution.

The types of problems we envision a Social Computer should be able to solve, and the environments in which it should exist, are of a much higher complexity. We anticipate that people and society, by using and interacting in principled ways through a Social Computer, will be able to solve hyper-complex problems. Such issues can be computational (e.g. how to solve a scientific problem that cannot be completely formalised), consensual (e.g. how to reduce the costs of health insurance) or controversial (e.g. how to reduce our carbon footprint, more generally how to reach UN defined sustainable goals). Addressing them will require collecting partial solutions from diverse human and machine clusters, assessing opinion from experts and from the public, predicting the outcome and consequences of individual subproblems, and other similar tasks impossible to achieve by humans or machines independently. A Social Computer is an integration of humans and machines collaborating together on-demand to solve problems and answer questions. It frees users from organisational burden, helping them in breaking down problems into manageable tasks; it allows deep and exhaustive search of information and data mining in order to obtain partial solutions; and it exploits at best the different human and computational resources to obtain effective solutions. Social Computers are not fixed, pre-defined entities like today's computers, but are dynamic, evolving collaborations of humans and/or machines, adapting themselves to the problem at hand.

Social Computers can also be seen as tools supporting decisions during the process of establishing public policies. They help gather, understand and create evidence in support of policy-making processes.

3 What about privacy, trust, ethics and compliance to laws and regulations?

Central to Social Computers above, and central to any adaptive system are the notions of ethics, privacy, trust and legal aspects. Most of the ICT developments so far were however primarily guided by the market, leaving behind ethical, legal,

and psychological considerations. Today's services offer no ethical warranty. Privacy is becoming a very fragile matter, with sensitive data often stored, used and aggregated unbeknownst to their owners, sometimes with malicious intent. For instance, people have not fully grasped the impact of the reputation of their online persona and their online actions, with undesired social or professional consequences. Society regularly define, revisit and enforces laws. Autonomous systems should not only adapt to unforeseen circumstances in their environment, but also be fully compliant with current regulations when first deployed *and* able to adapt - on their own - to any law or regulation change. The vision and proposal here is double: (1) we need to address ethical, trust, privacy and law-abiding considerations from the start, i.e. providing those consideration *by design* (ethics by design, privacy by design, compliance by design [1]); (2) we need also to develop research for engineering autonomous systems able to adapt to changes of laws and regulations *on-the-fly*. For autonomous systems, this could be provided by an ethical middleware or an ethical operating system ensuring in a built-in manner all those considerations; and/or ethical principles to be included into individual agents and global systems developed with them.

4 Conclusion

The considerations above have engineering concerns in mind. Including ethics, privacy and trust by design renders autonomous systems apparently more acceptable. Some questions still need to be considered: (1) How do we define the ethics to integrate into those systems? this shouldn't be left to individuals but thought at some universal world-wide level; (2) What about legal issues arising from situations involving autonomous systems? who is responsible when an accident, a failure or any harm happens: the user, the programmer, the designer, the company's manager? (2) To which extent is it a good thing to develop the "ultimate" adaptive system so intelligent that it may decide to question the usefulness of humans or even take over? Fortunately, ethicians, lawyers and philosophers are already busy thinking this through.

References

1. Guillaume Aucher, Guido Boella, and Leendert van der Torre. A dynamic logic for privacy compliance. *Artificial Intelligence and Law*, 19(2-3):187–231, 2011.
2. Betty Cheng, Rogério de Lemos, Holger Giese, Paola Inverardi, Jeff Magee, Jesper Andersson, Basil Becker, Nelly Bencomo, Yuriy Brun, Bojan Cukic, Giovanna Di Marzo Serugendo, Schahram Dustdar, Anthony Finkelstein, Cristina Gacek, Kurt Geihs, Vincenzo Grassi, Gabor Karsai, Holger Kienle, Jeff Kramer, Marin Litoiu, Sam Malek, Raffaella Mirandola, Hausi Müller, Sooyong Park, Mary Shaw, Matthias Tichy, Massimo Tivoli, Danny Weyns, and Jon Whittle. Software engineering for self-adaptive systems: A research roadmap. In Betty Cheng, Rogério de Lemos, Holger Giese, Paola Inverardi, and Jeff Magee, editors, *Software Engineering for Self-Adaptive Systems*, volume 5525 of *Lecture Notes in Computer Science*, pages 1–26. Springer Berlin / Heidelberg, 2009.

3. Vincenzo Ciancia, Diego Latella, Michele Loreti, and Mieke Massink. Specifying and verifying properties of space. In *IFIP TCS'14*, pages 222–235, 2014.
4. IBM Corporation. An architectural blueprint for autonomic computing, 2006.
5. G. Di Marzo Serugendo. Robustness and dependability of self-organising systems - a safety engineering perspective. In *Int. Symp. on Stabilization, Safety, and Security of Distributed Systems (SSS)*, volume 5873 of *LNCS*, pages 254–268, Lyon, France, 2009. Springer, Berlin Heidelberg.
6. Giovanna Di Marzo Serugendo, Marie-Pierre. Gleizes, and Anthony Karageorgos, editors. *Self-Organising Software - From Natural to Artificial Adaptation*. Natural Computing Series. Springer, 1st edition edition, 2011.
7. Giovanna Di Marzo Serugendo, Matteo Risoldi, and Mohammad Solemayni. The social computer. In J. Pitt, editor, *The computer after me*, pages 159–172. World Scientific, 2014.
8. A. E. Eiben, Zbigniew Michalewicz, M. Schoenauer, and J. E. Smith. *Parameter Setting in Evolutionary Algorithms*, chapter Parameter Control in Evolutionary Algorithms, pages 19–46. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
9. Jose Luis Fernandez-Marquez, Francesco De Angelis, Giovanna Di Marzo Serugendo, Graeme Stevenson, and Gabriella Castelli. The one-sapere simulator: A prototyping tool for engineering self-organisation in pervasive environments. In *SASO*, pages 201–202. IEEE Computer Society, 2014.
10. Jose Luis Fernandez-Marquez, Giovanna Di Marzo Serugendo, Sara Montagna, Mirko Viroli, and Josep Lluís Arcos. Description and composition of bio-inspired design patterns: a complete overview. *Natural Computing*, 12(1):43–67, 2013.
11. David Gelernter and Nicholas Carriero. Coordination languages and their significance. *Commun. ACM*, 35(2):97–107, February 1992.
12. L. Lana de carvalho, S. Hassas, E. J. Lopes, and A. Cordier. Four kinds of models of emergent representations resulting from the decomposition individual/collective and internal/external. In *Proceedings of 5th European Conference on Complex Systems (ECCS'08)*, 2008.
13. Rogério Lemos, Holger Giese, Hausi A. Müller, Mary Shaw, Jesper Andersson, Marin Litoiu, Bradley Schmerl, Gabriel Tamura, Norha M. Villegas, Thomas Vogel, Danny Weyns, Luciano Baresi, Basil Becker, Nelly Bencomo, Yuriy Brun, Bojan Cukic, Ron Desmarais, Schahram Dustdar, Gregor Engels, Kurt Geihs, Karl M. Göschka, Alessandra Gorla, Vincenzo Grassi, Paola Inverardi, Gabor Karsai, Jeff Kramer, Antônia Lopes, Jeff Magee, Sam Malek, Serge Mankovskii, Raffaella Mirandola, John Mylopoulos, Oscar Nierstrasz, Mauro Pezzè, Christian Prehofer, Wilhelm Schäfer, Rick Schlichting, Dennis B. Smith, João Pedro Sousa, Ladan Tahvildari, Kenny Wong, and Jochen Wuttke. *Software Engineering for Self-Adaptive Systems II: International Seminar, Dagstuhl Castle, Germany, October 24-29, 2010 Revised Selected and Invited Papers*, chapter Software Engineering for Self-Adaptive Systems: A Second Research Roadmap, pages 1–32. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
14. S. Mazac, F. Armetta, and S. Hassas. Bootstrapping sensori-motor patterns for a constructivist learning system in continuous environments. In *14th International Conference on the Synthesis and Simulation of Living Systems (Alife'14)*, New York, NY, USA, 2014.
15. D. Newman and N. Gall. *Gain a Foundation in Design Thinking to Apply Gartner's Hybrid Thinking Research*. Gartner Analysis, 2010.
16. Danilo Pianini, Sara Montagna, and Mirko Viroli. Chemical-oriented simulation of computational systems with Alchemist. *Journal of Simulation*, 2013.

17. Danilo Pianini, Mirko Viroli, and Jacob Beal. Protelis: practical aggregate programming. In Roger L. Wainwright, Juan Manuel Corchado, Alessio Bechini, and Jiman Hong, editors, *Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, April 13-17, 2015*, pages 1846–1853. ACM, 2015.
18. Mariachiara Puviani, Giovanna Di Marzo Serugendo, Regina Frei, and Giacomo Cabri. A method fragments approach to methodologies for engineering self-organizing systems. *ACM Transactions on Autonomous Adaptive Systems*, 7(3):33:1–33:25, October 2012.
19. Danny Weyns, Bradley Schmerl, Vincenzo Grassi, Sam Malek, Raffaella Mirandola, Christian Prehofer, Jochen Wuttke, Jesper Andersson, Holger Giese, and Karl. Goschka. On patterns for decentralized control in self-adaptive systems. In Rogerio de Lemos, Holger Giese, Hausi Mueller, and Mary Shaw, editors, *Software Engineering for Self-Adaptive Systems II*, volume 7475 of *Lecture Notes in Computer Science*, pages 76–107. Springer Berlin Heidelberg, 2013.
20. Michael Wooldridge and Michael J. Wooldridge. *Introduction to Multiagent Systems*. John Wiley & Sons, Inc., New York, NY, USA, 2001.
21. Franco Zambonelli, Andrea Omicini, Bernhard Anzengruber, Gabriella Castelli, Francesco Luca De Angelis, Giovanna Di Marzo Serugendo, Simon Dobson, Jose Luis Fernandez-Marquez, Alois Ferscha, Marco Mamei, Stefano Mariani, Ambra Molesini, Sara Montagna, Jussi Nieminen, Danilo Pianini, Matteo Risoldi, Alberto Rosi, Graeme Stevenson, Mirko Viroli, and Juan Ye. Developing pervasive multi-agent systems with nature-inspired coordination. *Pervasive and Mobile Computing*, 17:236—252, February 2015. Special Issue “10 years of Pervasive Computing” In Honor of Chatschik Bisdikian.