Chapter 1
Introduction

Abstract. Self-evolvability is advanced here as the key method to successfully manage the evergrowing complexity of systems.

The necessary transition from adaptable, to evolvable and finally to self-evolvable systems is highlighted.

Self-properties as self-organization, self-configuration, self-repairing and so on have been briefly introduced.

Challenges and limitations of the self-evolvable engineering systems are evaluated.

1.1 Self-Evolvable Systems to Manage Complexity

Technical systems as industrial equipments, telecommunication networks, and socio-technical systems as manufactures, companies and societies became more and more complex. This is the result of natural increasing complexity for products and markets, of the embedding of hardware, software, logical tools and necessary knowledge into such systems. The evergrowing complexity of modern industrial systems will lead to unsustainable increases in their management and operation costs.

Problems of design and organization become pressing as products increase in complexity regarding both hardware and software. It is becoming inevitable to shift much of the problem of organization into the machines themselves. This brings up the problem of keeping the self-organization under control. Product, installations, hardware and software engineers aim to relieve growing complexity problems by conceiving adaptable, evolvable and finally self-evolvable systems.

A self-evolvable system should be capable to adapt dynamically to the current conditions and future requests of its environment.

Conventional engineering or scientific methods have been conceived to deal with systems reducible to simpler parts which exhibit controlled behavior.

A significant step in high complexity management was and continues to be the pursuit of autonomy for systems. However, the autonomous robotic systems will inevitably fail to meet some tasks or obligations when they are operated in different field conditions, if they are not self-evolvable.

Self-evolvability is advanced here as the key method to confront evergrowing complexity and to successfully run in the higher complexity domains.

The self-evolvability paradigm is the proposed response to the shift to faster and stronger hardware integrated computer systems, to the need for less external management of high complexity systems.

To explain the self-evolvability need, we start from the observation that there are significant differences between adaptable, evolvable and self-evolvable systems.

Fig. 1.1 illustrates the trends and steps in understanding and system modeling as complexity grows. Adaptability, based on learning, implies optimization or adjustment on the time scale of the existence of a system as for instance an industrial product or organization. Adaptability may refer to animate and its inanimate environment. It offers a preliminary and low dimensional perspective for complexity running.

In a simplified form, the learning models describing adaptability considered two spaces, the space of states, S and the space of conditions, K and their interaction (Fig. 1.1a). The learning system is presented with a series of conditions $k \in K$, on each of which it changes states $s \in S$, allowing sequential adaptability (Bush and Mosteller 1955, Iosifescu and Grigorescu 1990).

The evolvability requires more than learning, specifically the capacity for change to autonomous march into new life cycles, for instance new type of products, new market niches, new organizations and new levels. This entails a higher dimensional perspective and also the systems closure.

The evolvability refers to several levels, for instance, to a transformation from physical states S, toward biological-like, K1, cognitive-like, K2, intelligent-like, K3 and finally closed evolvable systems. Four levels systems in which the states S are related to a hierarchy of conditions K1, K2, K3 have been studied in different engineering domains where evolvability was implemented (Iordache 2010).

Fig 1.1b shows the basic cognitive frame for evolvability in which the states S interact with the conditions K1, meta-conditions K2, and meta-meta conditions K3.

Critical to achieve evolvability is the embodiment that is the connection between the top level of conditions K3 and the states S. Artificially or naturally evolvable systems are supposed to cross this critical gap between these two extreme levels.

The continuous growth of complexity imposes a new transition from evolvability to self-evolvability, that is to systems that self-configure, self-optimize, self-control, self-heal and so on, based on a set of higher-level intrinsic capabilities and meeting of the user-specified variable objectives.

The polytope shown in Fig. 1.1c illustrates the architecture of such self-evolvable systems. It is a representation known as hypercube or 4-cube and consists of a cube inside another cube. The 4-cube is obtained by joining all corners of the inner cube with the corresponding corners of the outer cube (Ziegler 1995).

As shown in Fig. 1.1c, complementing the direct way of integration and convergence $S \rightarrow K1 \rightarrow K2 \rightarrow K3$ we need to look at the reverse way of differentiation and divergence $K3' \rightarrow K2' \rightarrow K1' \rightarrow S'$.

The convergence way does not quite grasp the essence of creativity required by self-evolvability. That is because the boundaries where creativity flourishes and new information and new solutions are created consist of synchronized tendencies. Tendencies to converge should coexist with tendencies to diverge and it is the metastable blend of both that matters (Kelso 2002).

The Self-cube centers and correlates the four-level evolvable frames shown on different faces of the outer cube (Fig. 1.1c). Swinging, mediated by the Self between the two complementary way's is crucial for self-evolvability.

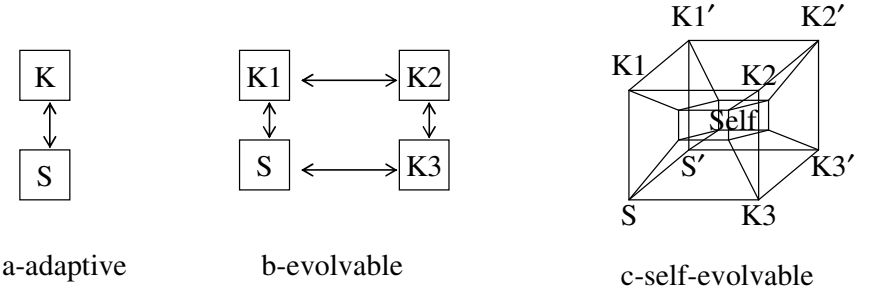

Fig. 1.1 Modeling architectures for growing complexity

Evergrowing complexity management needs conveying, as shown in Fig. 1.1, from adaptive, to evolvable and lastly to self-evolvable systems. The self-evolvability subsumes and challenges the adaptability and the evolvability stages.

For the design and control of forthcoming self-evolvable systems a natural query is: why not use traditional methods, based on modeling and extrinsic implementation of the models in the usual computer based adaptive design and control?

A two-level relation between states S and conditions K, as shown in Fig. 1.1a, may accomplish some adaptive control tasks. But the imagined strategy is not to reduce several interconnection steps to the two levels architecture, as an attempt to achieve a degree of control in the speediest manner. Contrary to this kind of reductionism, the evolvability schemas shown in Fig. 1.1b look to the basic four interactions steps to allow the study and to take into account the necessary ingredients, as shown by the evolvable systems and organisms existing in nature.

From the computer science and engineering points of view, the apparent ease with which living systems solve computationally difficult organizational problems makes it inevitable to adopt the strategies observed in nature for creating information processing architectures.

For systems of higher complexity, the envisaged computing and control tasks are impossible to be extrinsically operated. For conventional design and control the majority or non-linear interactions that could contribute to the problem are inherently excluded. The properties characterizing high complexity systems should be the consequences of their own dynamic of the computational environment, not of the decision of the external designer or program that is anyway unable to predict the evolution of its construction.

Self-evolvable systems are intended to work for their building and evolution more efficiently than any external computer-aided operator can do. Only self-evolvable systems will have the potential to survive in unforeseen situations.

Additionally, it was observed that the more an adaptive system is optimized to perform a task the less capability it has to deal with unexpected or unprogrammed major changes in an uncertain environment. This implies that for highly complex environments, self-evolvability rather than optimization or adaptability may be the appropriate measure of the system potential to carry out high complexity tasks and to survive.

Building on the adaptable and evolvable systems, we see the potential for the development of sophisticated methods, devices and controllers for use on autonomous systems. But it is impossible for engineers to anticipate all possible events and non-linear combinations of events that may be faced by systems in hazardous, hostile, or increasingly complex environments. Providing the industrial system with a representation of human-like cognition and a knowledge base from which decisions may be developed will greatly expand the applicability of autonomous systems as solutions to high complexity problems (Strunk and Ganger 2003).

It is acknowledged that as systems continue to grow in complexity, they will reach a point where administrators will not only be unable to understand the behavior but will lack the ability to control that system by conventional methodologies. A breakthrough is necessary since research is at the crossroads and the conventional methods will no longer work (Ritchey 2011).

An embedded human-like cognitive control capability may provide the means for establishing reliable control of these systems. This is the perspective visualized by the self-evolvable architectures from Fig. 1.1c. This outlines the Self, correlating the two ways of integration and differentiation.

Since there is no fixed limit for growing complexity, higher dimensional polytopes as 5-cubes and other polytopes will be considered as cognitive architecture instead of the 4-cube shown in Fig. 1.1c. Self-evolvability is the growth toward greater understanding and control not its attainment.

At the today technological level, a project grouping in a system the major faculties of self-evolvability becomes realistic.

Undoubtedly the agenda for development of self-evolvable systems requires a transdisciplinary and selfdisciplinary effort. It is expected to implement self-evolvability as an increasing capability in steps. Gradually it will be possible to make a system more close to self-evolvable ones as the complexity understanding and resources to prevail over high complexity will grow.

Self-evolvability will emerge as a natural consequence of step-by-step implementations of the self-properties.

1.2 Self-Properties

The term self-property generally refers to the acquirement of the indicated property in the absence of external intervention or control.

The self-evolvable systems are based on integrated hardware, software and selfware infrastructure**.** Selfware concept defines the growing set of self-properties that are emerging in self-managing computing systems major initiatives as for instance autonomic computing or organic computing (Sterritt and Hintchey 2005).

The initial set of self-properties, namely self-configuration, self-healing, self-optimization and self-protection as objectives to be attained through self-awareness, self-monitoring and self-adjusting attributes, has been expanded, and further properties are expected to be added to this still growing list.

Diverse interpretations of concepts as self-properties and emergence can be found in the literature (Banzhaf 2002a, 2002b, Frei and Di Marzo Serugendo 2011a, 2011b).

A taxonomy of self-properties which focuses on decentralized autonomic computing and discusses characteristics of self-properties and implications for their engineering was proposed (De Wolf and Holvoet 2007). This taxonomy takes into account if a self-property is achieved on macroscopic or microscopic level, if it is on-going or one-shot, if it is time dependent or independent, if it evolves in a continuous or abrupt way, and it is adaptation-related or not. The taxonomy gives examples of mechanisms leading to self-properties and classifies application examples according to the considered characteristics.

The self-properties of interest, mainly for industrial engineering systems, will be briefly characterized in the following.

• Self-organization

Self-organization is defined as a process where systems acquire and maintain structures themselves, without external control. Self-organization can be broadly understood as the ability of a system to change its internal structure and its function in response to external circumstances.

A system creates or adapts its own structure to reach a goal. Components may form and break coalitions to provide the requested capabilities. A self-organizing system can assemble, construct and stabilize itself, with the help of outside matter, energy or information.

Self-organization would not be possible without nonlinear interactions between components of the system. The self-organizing systems are open-ended and not in static equilibrium state with their environment. Self-organization includes the increase in structure, autonomy and robustness with reference to changes and far-from-equilibrium conditions.

• Self-adaptation

A self-adaptive system adjusts itself to changing conditions without major physical modifications. For instance, in the case of an industrial production system when more urgent requests arrive, an automaton can modify its working parameters (Frei 2010).

Self-adaptation is one of the processes by which a system can self-organize. Whether it is a system of brain cells that adapt to firing patterns, or a system of street signals that adapt to traffic, or a system of product manufacturing cells that

adapt to new product requests, similar formal models might be able to describe those systems.

• Self-configuration and Self-reconfiguration

These refer to systems that automatically configure and reconfigure components to adapt them to different environments. Self-configuring is based on feedback from environment.

A self-configuring system prepares itself for functioning, including the adjustment of parameters and calibration. Automata adjust their geometry and movement accuracy to the desired range of requests (Frei and Di Marzo Serugendo 2011a, 2011b).

Self-reconfiguration encompasses self-adaptation, but also basic change, including the software and hardware. For instance, when a component or a cell fails, and there is an alternative component or cell path to reach the affected destination, the parts adapt their behavior and use the alternative path until the part has recovered from the failure. A new device or cell is required and it should be integrated into the existing system.

A self-reconfigurable modular approach was imagined to build responsive industrial systems to satisfy customers, various demands (Mun et al. 2004, Hu and Efstathiou 2007).

• Self-assembly and Self-disassembly

The self-assembly refers to sub-systems or parts that connect with each other, to form the totality.

By disassembly, system decomposes itself into subsystems or parts. An association which is not necessary any more may disassembles or may be disconnected.

The disassembly process allowing the formulation of new systems is as important as the self-assembly itself.

There are some differences between self-assembly, self-formation, and self-organization concepts (Banzhaf 2002a).

Self-assembly comprises the assembly of parts into a whole, directed by the assembling parts and their interactions. A self-assembling process is usually not recursive, that is, it cannot move through successive stages of first assembling some elementary parts into more complex parts, which in turn self-assemble into the whole.

Self-formation has a clear feature of sequence. Processes of self-formation can be used to generate more complicated entirety. This requires the developing system to change state repeatedly, with each state determining subsequent states and the sequence of events to follow. More complex wholes can be constructed by such a mechanism, and it is possible to generate patterns of higher complexity. Self-formation finds its limits in the problem of repairing and maintenance of the structures formed. Because of the very specific sequence of events that lead to the original result, these needs are difficult to achieve with self-formation.

We may consider self-organization to be the most general term in this order, including both self-assembly and self-formation, but also self-maintenance and self-development.

Self-organization does not require a specific sequence to arrive at the end result since it has a multitude of paths toward the desired goal. Self-organization allows phenomena on different time scales, and a hierarchy of levels, which in turn allows a recursive consideration of its mechanisms.

• Self-diagnosis, Self-repairing and Self-healing

Components can find out and state what is not right with their functioning. A device that cannot provide products may check if there are no ready products inside, or if there is a blockage or any other problem preventing normal performance (Barata et al. 2007).

Self-repairing refers to systems that automatically discover, diagnose and correct faults.

A system can treat its problems and maintain or re-establish functionality. A blocked path or device will restart its software, execute calibration movements, check and if still blocked, may ask the user for debugging. Self-healing implies a level of tolerance to errors.

• Self-reproduction, Self-replication and Self-fabrication

These properties refer to systems that can create copies. A modular assemblage supports suitable modules to form the same type of assemblage. Self-similarity may be a feature of the resulting system.

The self-fabrication refers to fabrication process that happens spontaneously without assistance from a fabricator. Hofmeyr identifies unassisted self-assembly as the process that ultimately makes the system self-fabricating (Hofmeyr 2007).

Hofmeyr characterizes living systems as featuring persistence despite a continual decay of their molecular components and machinery and despite changes in context. This led to present a living system as a chemical factory of which the output is the factory itself.

The concept of autonomous self-fabrication was developed in the theory of autopoietic systems (Maturana and Varela 1980) and of self-reproducing automata based on universal constructors (Von Neumann 1966).

• Self-protection and Self-security

These refer to systems that anticipate, identify and protect against arbitrary attacks.

A self-evolvable system should protect itself from interferences or attacks. In case a system was open enough for external agents to gain access to it, it would need to protect itself from possible damages. A self-protecting system observes, constructs, knocks down and modifies its boundaries.

• Self-control, Self-determination and Self-management

These refer to a system that guides itself. The modules control their own behavior, for instance that directed by rules and policies. Self-management should characterize a system that can take care of itself. This may include self-protection, self-healing, self-configuration, self-optimization, and self-adaptation. Self-optimization, for instance, means that the system automatically monitors and adapts resources to ensure optimal functioning regarding the defined requirements.

For example, at production time, the processing components should maintain themselves as well as their neighbors in safe conditions. They should manage their multi-lateral interactions providing the requested services and schedules maintenance.

• Self-awareness

This refers to the capability of the system or its individual components to identify by themselves, internally, any new condition, failure, or problem, without specifically being instructed, from outside, by any administrator. A system is self-aware in that it can observe itself and improve its behavior to meets its goals.

Self-awareness requires sensing capabilities and triggers reasoning and acting.

Self-aware systems are currently thought of as equipped with monitoring, planning and plan execution capabilities at the level of autonomic managing. Self-aware systems sense their environment in different ways as configurations or neighbors, and take decisions accordingly, changing directions, roles, goals or links (Di Marzo Serugendo et al. 2010). Self-awareness concept may be related to the recent attempts to create artifacts that have some characteristics typically associated with consciousness (Haikonen 2007, Sanz et al. 2011)

One important differentiation to be made for all the self-properties is the direction.

Self-adaptation and self-management are considered as top-down, whereas self-organization and self-healing are considered as bottom-up properties.

A self-evolvable system will swing between the two directions and benefits from both directions. This kind of swinging was described for architectures shown in Fig. 1.1c.

1.3 Challenges and Limitations

Self-properties represent an important part of complexity engineering. They allow systems to play active and increasingly autonomous roles in accomplishing their tasks, but there are also challenges and limitations to the possibilities of self-properties.

Self-evolvability offers potential solutions to evergrowing complexity but also may be a source of new problems.

Some issues and warnings in implementing self-properties, some solutions and perspectives have been discussed in the literature and will be presented in the following (Herrmann et al. 2005, Frei and Di Marzo Serugendo 2011a, 2011b).

• Inherent Differences between Engineered and Natural Systems

A difference between systems that are engineered and systems that result from natural processes is that engineered systems may be optimized for one or more particular properties with inadequate search for self-evolvability. When the environment in which an engineered system changes or when the uses to which one wants to put an engineered system changes, it is often difficult to change the system to accommodate the new needs.

It was observed that natural systems tend to be much more adaptable than engineered systems (Abbott 2006, 2007). An open problem is to understand the root of this difference and what can be done to surpass such differences.

The increasingly complex systems that self-evolve are not to be re-designed from scratch with every evolvability step. Each step is an evolutionary neighbor of something less complex. The evolutionary sequence provides both a specific feature needed at each step and a framework that can support the evolutionary process itself. The architectures that survive are those that support and facilitate self-evolvable change. Thus the evolutionary process produces not only fitness for changing environments but also itself self-evolvability.

New versions of many engineered systems may be built from scratch, that is, they are not evolved as extensions of previous versions. Those that are extensions of previous versions are often uncoordinated. In the long run, self-evolution rewrites and reduces the structures that impede self-evolution leaving evolvable core frameworks.

It should be emphasized that self-evolvability is not required for any engineering systems. One reason for this is that designing for self-evolvability is very difficult. Another reason is that we do not have a satisfactory way either of specifying or measuring self-evolvability. Given the current state of the knowledge about self-evolvability, the most effective way of requiring self-evolvability is economic. This implies that the developers should evaluate and absorb parts of the cost of post-delivery enhancements.

• Sensitivity to Specific Conditions

Industrial systems often exhibit sensitivity to specific conditions and to disturbances. Certain factors, like energy disruptions or an abnormal increase of temperature and humidity, may lead to system breakdown, while others, as for instance the extreme noise, may have insignificant effects. Some disturbances may have consequences in some cases, lack any effect in others, and may be beneficial in some cases. An automaton using thermal sensors reacts sensitively to changing temperature conditions, whereas an automaton working with optical sensors remains unaffected by temperature.

A degree of randomness may be beneficial for systems structuration in levels.

Systems may efficiently find a way to accomplish their task under certain initial conditions, but may not be able to do so when the conditions are to some extent different. Engineers have to consider their systems' sensitivity to initial conditions and attempt to find solutions to alleviate the effects.

• Time-Scales and Impossibility to Find New Stable States or Convergent Solutions

Most self-evolvable systems can eventually find stable states or stable solutions to achieve their tasks, but this takes time. This means that the designer and the user of self-evolvable systems must be able to accept postponement.

In certain cases, a self-evolvable system may not be able to solve the task given or its calculations may converge too slowly. The engineer has to preview this and arrange for a way out, such as alerting the user and settling for a solution which requires the relaxation of certain constraints or imposing new ones.

The delay in self-evolvability achievement may be due to the inherent difficulty to analyze self-properties. Moreover, the system may find ways to fulfill tasks which the engineer did not plan or preview. The engineer may intend the system to act in a certain way, and in reality, it is all different. Also the interplay between various self-properties may be difficult to analyze and further research efforts may be necessary.

• Reliability and Resilience

Reliability is defined as the probability that an item will perform its intended function for a specified time interval under stated conditions. Reliability analysis addresses the changes in quality of a system over time.

Resilience is the property of a system that enables it to resume its original capabilities after being perturbed. Resilience is a dynamic process that systems exhibit positive behavioral adaptation when they encounter significant sources of stress.

The engineer should be assured that the system does what it is supposed to do, independent from the actual situation and circumstances, and this may be challenging, for some high complexity systems.

Thanks to their redundancy, the systems are often inherently robust to certain types of failures. They may, however, be fragile when facing a different type of fault. An open problem is the lack of widely accepted of what self-evolvability actually requires for reliability and of appropriate standards for unifying the self-evolvability processes.

• Control Issues and Predictability

A common warning that opposes the deployment of complex systems knowledge in engineering is the idea that all control over the system will be lost. However, the issue about self-evolvability is not that the developed systems will be unpredictable, nondeterministic or uncontrolled (Buchli and Santini 2005).

It is expected that the guarantees about the functioning of the system will be of statistical nature. In all engineering works, the guarantees that can be made about a system are limited and essentially of statistical nature. Furthermore, it shows an erroneous understanding of how the development of technology works if a complete understanding of the functioning of a system is demanded before it is accepted. Technology and engineering have worked with systems which are not

fully understood but nevertheless they are accommodating and brought good or acceptable services.

Often the understanding is deepened after which it can lead to a yet better or more efficient use. Self-evolvability for a complex system is the way to arrange itself, to find the balance between controllability, predictability, impredicative behavior and a letting loose of some fuzzy aspects of the system (Frei and Di Marzo Serugendo 2011a, 2011b).

Self-evolvability inherently implies loops and circularity. It is known that a graph will contain no cycles or loops if and only if it is well founded (Aczel 1988). This means that a graph describing a complex system that contains loops or cycles is a picture of a non-well-founded set. The presence of cycles and loops would indicate that certain set has itself as a member or that the concept system or definition it models is impredicative.

Impredicative behavior does not mean that the system can't be computable, constructed or engineered (Mossio et al. 2009). For instance, the graphs of a proof may be allowed to contain a balance of cycles and hierarchical trees (Santocanale 2002).

• Relevance Limits

It should be emphasized that self-evolvability is not meant to replace traditional engineering approaches, but is merely a natural development complementing, reactivating and enhancing conventional domains.

Self-evolvability has to be applied to appropriate and critical problems. Even if the engineer will take inspiration and learn from physical and living systems, his main goal is not restricted to understanding the nature. The goal to formulate a model which should serve for understanding the real-world phenomena is the primary concern of the scientist.

Thus if an engineer finds that a few modifications to a system may serve him well, even if these modifications are not justified by complete observations in the real systems, there is no reason why he should not use the modified system and no other justification for the change is needed.

Numerous traditional systems are still operated in a regime where the complexity properties are observed but may be neglected. The reductionistic approach is a powerful method for gaining knowledge about certain aspects of the natural world and for supporting the development of technology precisely because it selects phenomena that have a simple explanation. But the cost of restricting to simplicity only is to be unable to represent the full range of possibilities offered by engineering systems. As a mandatory way of success of thinking and a firm base for undertaking, the reductionism should be complemented by the way of systemic or integrative thinking. The subtle blend of both ways, the reductionistic and the complexity ways, the differential and integrative ways, proved to be the main sources of efficiency.

While, in some cases, traditional engineering studied the ways of taking the complexity out of the systems, we now have to allow complexity to come back in, to complement the reductionism and learn to exploit both epistemological ways for our own good.

Surely, the complexity science and engineering with a self-evolvability perspective in mind will be more goal-directed and successful than conventional reductionistic and distributed attempts (Ottino 2004).

References

Abbott, R.: Complex systems + systems engineering = complex systems engineering. In: Conf. on Systems Engineering Research, California State University, Los Angeles and The Aerospace Corporation. Tech. Rep. (2006)

Abbott, R.: Putting complex systems to work. Complexity 13, 30–49 (2007)

Aczel, P.: Non-well-founded Sets. Stanford, CSLI (1988)

Banzhaf, W.: Self-organizing Systems. Encyclopedia of Physical Science and Technology, pp. 589–598. Academic Press, New York (2002a)

Banzhaf, W.: Artificial Chemistries–Towards constructive dynamical systems. Nonlinear Phenom. Complex Syst. 5, 318–324 (2002b)

Barata, J., Ribeiro, L., Onori, M.: Diagnosis on evolvable production systems. In: IEEE International Symposium on Industrial Electronics (ISIE), Vigo, Spain, pp. 3221–3226 (2007)

Buchli, J., Santini, C.: Complexity engineering, harnessing emergent phenomena as opportunities for engineering. Tech. Rep., Santa Fe Institute Complex Systems Summer School, NM, USA (2005)

Bush, R., Mosteller, F.: Stochastic Models for Learning. Wiley, New York (1995)

De Wolf, T., Holvoet, T.: A taxonomy for self-* properties in decentralized autonomic computing. In: Parashar, M., Hariri, S. (eds.) Autonomic Computing: Concepts, Infrastructure, and Applications, pp. 101–120. CRC Press, Taylor and Francis Group (2007)

Di Marzo Serugendo, G.: Robustness and Dependability of Self-Organizing Systems - A Safety Engineering Perspective. In: Guerraoui, R., Petit, F. (eds.) SSS 2009. LNCS, vol. 5873, pp. 254–268. Springer, Heidelberg (2009)

Di Marzo Serugendo, G., Fitzgerald, J., Romanovsky, A.: Metaself - an architecture and development method for dependable self-* systems. In: Symp.on Applied Computing (SAC), Sion, Switzerland, pp. 457–461 (2010)

Frei, R.: Self-Organization in Evolvable Assembly Systems. PhD. Thesis, Department of Electrical Engineering, Faculty of Science and Technology, Universidade Nova de Lisboa, Portugal (2010)

Frei, R., Di Marzo Serugendo, G.: Advances in Complexity Engineering. Int. J. of Bio-Inspired Computation 1(1), 11–22 (2011a)

Frei, R., Di Marzo Serugendo, G.: Concepts in Complexity Engineering. Int. J. of Bio-Inspired Computation 3(2), 123–139 (2011b)

Haikonen, P.O.: Robot Brains: Circuits and Systems for Conscious Machines. Wiley & Sons, Chichester (2007)

Herrmann, K., Mühl, G., Geihs, K.: Self-Management: The Solution to Complexity or Just Another Problem? IEEE Distributed Systems Online 6(1), 1–17 (2005)

Hofmeyr, J.-H.: The biochemical factory that autonomously fabricates itself: a systems biological view of the living cell. In: Boogard, F., Bruggeman, F., Hofmeyr, J.-H., Westerhoff, H. (eds.) Systems Biology: Philosophical Foundations, pp. 217–242. Elsevier (2007)

Hu, B., Efstathiou, J.: Responsive system based on a reconfigurable structure. In: Pham, D.T., Eldukhri, E.E., Soroka, A.J. (eds.) 2nd I*PROMS Virtual Conference on Intelligent Production Machines and Systems, pp. 517–522 (2007)

Iordache, O.: Polystochastic Models for Complexity. Springer, Heidelberg (2010)

Iosifescu, M., Grigorescu, S.: Dependence with complete connections and applications. Cambridge Univ. Press, Cambridge (1990)

Kelso, J.: The complementary nature of coordination dynamics: Self-organization and the origins of agency. Journal of Nonlinear Phenomena in Complex Systems 5, 364–371 (2002)

Maturana, H., Varela, E.: Autopoiesis and Cognition. The Realization of the Living. Reidel, Dordrecht (1980)

Mossio, M., Longo, G., Stewart, J.: An expression of closure to efficient causation in terms of λ-calculus. J. Theor. Biol. 257, 498–498 (2009)

Mun, J., Ryu, K., Jung, M.: Self-reconfigurable software architecture: design and implementation. In: Proc. 33rd Int. Conf. Computers Industrial Engineering, CIE569, Jeju, Korea, March 25–27, pp.1–6 (2004)

Ottino, J.M.: Engineering Complex Systems. Nature 427, 399 (2004)

Ritchey, T.: Wicked Problems – Social Messes: Decision support Modeling with Morphological Analysis. Springer, Berlin (2011)

Santocanale, L.: A Calculus of Circular Proofs and its Categorical Semantics. In: Nielsen, M., Engberg, U. (eds.) FOSSACS 2002. LNCS, vol. 2303, pp. 357–371. Springer, Heidelberg (2002)

Sanz, R., Hernández, C., Sánchez, G.: Consciousness, meaning and the future phenomenology. In: Machine Consciousness; Self, Integration and Explanation, York, UK, AISB (2011)

Sterritt, R., Hinchey, M.: Autonomic Computing – Panacea or Poppycock? In: The 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems, ECBS, pp. 535–539 (2005)

Strunk, J.D., Ganger, G.R.: A human organization analogy for self-* systems. In: Algorithms and Architectures for Self-Managing Systems, San Diego, CA, June 11, pp. 1–6. ACM (2003)

Von Neumann, J.: Theory of Self-Reproducing Automata. University of Illinois Press, Urbana (1966)

Ziegler, G.M.: Lectures on Polytopes. Graduate Texts in Mathematics, vol. 152. Springer, New York (1995)