Using self-organized living structures principles in the design of Complex Adaptive Systems

Radu Dobrescu, Matei Dobrescu, Dan Popescu
POLITEHNICA University of Bucharest, Faculty of Control and Computers

Abstract - We present a review of complex systems research which focuses on their similarities with self-organizing living systems. We classify the types of complex systems that relate to self-organisation and we discuss the overall requirements for self-organising modeling. As a novelty, the paper proposes a methodology to aid engineers in the design and control of adaptive complex systems. The methodology offers a conceptual framework and a series of steps to follow to find proper mechanisms that will promote elements which by actively interacting among themselves lead to better performance.

Keywords - complexity modeling, self-organisation, coevolutive structures, adaptive control systems, heterarchical structure, design methodology.

I. INTRODUCTION

The term *self-organization* has been used in different areas with different meanings, as in cybernetics (Heylighen, 2003), thermodynamics (Nicolis and Prigogine, 1977), biology (Feltz *et al.*, 2006), computer science (Kohonen, 2000), complexity (Schweitzer, 1997), information theory (Haken, 1988), robotics (Steels, 2003), synergetics (Haken, 1981). Many people use the term “self-organization”, but it has no generally accepted meaning, as the abundance of definitions suggests. Also, proposing such a definition faces the philosophical problem of defining “self”, the cybernetic problem of defining “system”, and the universal problem of defining “organization”. We will not attempt to propose yet another definition of self-organizing systems. Nevertheless, in order to try to understand these systems better, we will combine insights from different contexts where self-organizing systems have been studied.

In a complex system it is often the case that the utility of a structure or process is expressed at the next higher level of organization relative to the process itself. Unlike entropy and the related concept of information, complexity is not extensive, nor is it entirely intensive. What is clear though is that complexity concerns a specific description, which is of course dependent on the technology and subjective capabilities of the observer. Anyway, we can consider that a complex system is a system with a large number of elements, building blocks or agents, capable of interacting with each other and with their environment. The interaction between elements may occur only with immediate neighbors or with distant ones; the agents can be all identical or different; they may move in space or occupy fixed positions, and can be in one of two states or of multiple states. The main characteristic of all complex systems is that they display organization without any external organizing principle being applied. The paper proposes a methodology that use these feature for a improved design of adaptive complex systems.

II. TYPES OF COMPLEXITY

Previous work has identified four classes of complexity (Lucas, 1999), of which only the last is directly relevant to our focus here. In this more general treatment we will extend these concepts to cover high-dimensional complexity, where in the limit the system is assumed to possess infinite components. These four nested complexity types (the later including the former) are: i) type 1: Static Complexity - Fixed structures, frozen in time, (for example the visual complexity of a computer chip or a picture). ii) type 2: Dynamic Complexity - Systems with time regularities (this includes such states as planetary orbits, heartbeats, seasons; they have cyclic attractors). iii) type 3: Evolving Complexity - Open ended mutation, innovation (these are open, non-equilibrium systems and can be regarded as existing on a permanent non-repeatable transient; also related are diffusion-aggregration and similar branching tree structures). iv) type 4: Self-Organising Complexity - Self-maintaining systems.

Operating at the edge of chaos, the systems of the type 4 loop back on themselves in nonlinear ways and generate the rich structure and complex mix of the above attractors. This is the advent of autopoiesis, the creation of adaptive self-stabilising organic systems that can swap between the available attractors depending upon external influences and also modify and create the attractors coevolutionarily (by learning). They differ from the purely evolving category in that state space is canalized by the self-organising nature (downward causation) of their internal emergent processes, thus possible functions are self-limiting. These systems occupy dissipative, semi-stable, far-from-equilibrium positions exhibiting the typical power law distribution of events familiar from critical systems at the phase transition, they are structurally and organisationally both open and closed, with semi-permeable material and informational membranes allowing the passage of operational triggers driving their attractor modes.

We can summarise the structure of these complex self-organizing systems in an overall heterarchical view (Fig.1) where successively higher levels show a many to many (*N:M*) structure (metasystem).

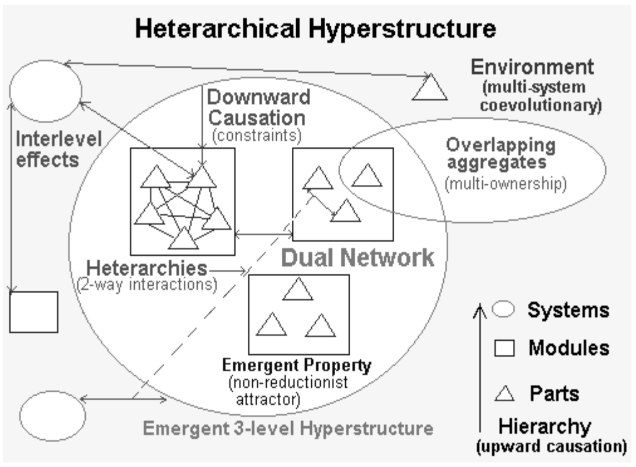
Heterarchical Hyperstructure
Environment (multi-system coevolutionary)
Downward Causation (constraints)
Interlevel effects
Overlapping aggregates (multi-ownership)
Dual Network
Heterarchies (2-way interactions)
Emergent Property (non-reductionist attractor)
Emergent 3-level Hyperstructure
Systems
Modules
Parts
Hierarchy (upward causation)

Fig. 1. A typical hyperstructure for self-organizing systems

Part interactions create emergent modules with new properties. These modules themselves interact as parts at an higher level and this process leads to the creation of an emergent *Hierarchical system* (the upward causation). The components at each level connect horizontally to form an *Heterarchy* - an evolving web like network of associations. Additionally systems can have overlapping members at each level (e.g. individuals can belong to many social groups, molecules to many substances, a situation to many models and a model to many situations). These large scale interacting emergent systems are called Hyperstructures, groups of interlaced dual networks constrained by downward causality (Baas, 1994). Here we extend these ideas slightly by allowing explicit cross level interference between systems (e.g. an individual affecting another country overall, a cell affecting an external part). This extended design we call here an *Heterarchical Hyperstructure*. Given that a metasystem has such a set of structures, then the overall fitness will relate to the interdependent properties at all levels, in other words to the full contextual environment. Now, putting some of the main points together we can arrive at a significant definition of the interraction process in self-organizing systems. *Definition*. Critically interacting components self-organize to form potentially evolving structures exhibiting a hierarchy of emergent system properties.

The elements of this definition relate to the following features: *Critically Interacting* (System is information rich, neither static nor chaotic); *Components* (Modularity and autonomy of part behavior implied); *Self-Organize* (Attractor structure is generated by local contextual interactions); *Potentially Evolving* (Environmental variation selects and mutates attractors); *Hierarchy* (Multiple levels of structure and responses lead to a hyperstructure); *Emergent System Properties* (New features are evident). Therefore, to better understand self-organizing systems we must concentrate less on the building blocks and more on their internal interactions.

III. COMPLEX ADAPTIVE SYSTEMS AS SELF - ORGANIZING STRUCTURES

Over the last half a century, much research in different areas has employed self-organizing systems to solve complex problems. Particular methodologies using the concepts of self-organization have been proposed in different areas, such as software engineering (Zambonelli and Rana, 2005), electrical engineering (Ramamoorthy *et al.*, 1993) and collaborative support (Jones *et al.*, 1994). However, there is as yet no general framework for constructing self-organizing systems. Different vocabularies are used in different areas, and with different goals. The methodology proposed in this section, useful for designing and controlling *complex* systems, provides a *conceptual framework* to assist the solution of problems. What this methodology suggests is a way of introducing the expectation of change into the development process.

Many of the key processes in coevolution - adaptation on multiple levels, dynamic feedback loops, mutually causal flows of knowledge across boundaries - are at the core of several complexity disciplines. More importantly, the essential goal of coevolution - studying the adaptive changes within and between all levels of organizational and environmental interactions - can be operationalized in terms of emergence. In this sense a complex process can be modeled as a complex attractor, which, like strange attractors in *deterministic chaos theory,* provides a method for mapping the dynamics of interactive systems. According to this view, exploitation and exploration processes are complementary means for optimizing organizational resources and design features in the face of multiple environmental constraints.

Complex adaptive systems (CAS) offers an alternative approach for studying the emergent behaviors of agents or populations adapting and coevolving in a computational context. In complex adaptive systems, agents adapt by changing their rules as experience accumulates. In addition, each change of strategy of an agent alters the context in which the next change will be tried and evaluated. When multiple populations of agents are adapting to each other, the result is a coevolutionary process. Studying this emergence process can generate insights about the mutual, simultaneous and nested effects of coevolution. Perhaps more important, CAS as a discipline can help define interaction process that hold across levels, which may allow researchers to identify similar patterns acting in macroevolution and in microevolution. This search for similar patterns across scale can be aided by the mathematics of *fractals.* The fractal notion of “self-similarity across scales,” and the resulting topological mapping techniques used to uncover those often unseen patterns, has been utilized by complexity scholars. Although the operationalization of “fractal dimensions” may not yet be obvious in coevolutionary contexts, the mathematics is a unique way to reveal whole-part relations that are a key to understanding mutual adaptation processes.

A critical part of explaining interactions between and across levels is the feedback loops that are involved. The bi-directional influencing processes are a central property of coevolution research, and *system dynamics* provides a powerful means for modeling the non-linearities of these positive feedback systems. System dynamics forces researchers to carefully identify each feedback process within an entire system; the rule-based computational model can reveal hidden interdependencies and emergent characteristics that are not tractable using linear thinking.

At a more micro level, *cellular automata* modeling could augment coevolution research by examining the relationship between individual agent moves (e.g. strategic adaptations) and the moves of that agent's immediate neighbors. Like other computation-based disciplines of complexity, the algorithmic tractability of the model makes it easy to test many different configurations in a short period of time, thus speeding up the theory-building process.

Whereas many of these computational models are grounded in structured rules that mediate flows of behavior, deep structures and resource flows are also at the heart of the qualitative theories of *autogenesis/autopoiesis.* Autogenesis is a theory of identity-making, in which an agent's core values and schemas define the rules that formulate emergent structures. The value of autogenesis/autopoiesis is its conceptualization of the mutual causality of resource flows and environmental potentials. According to the theory, entities (agents) are constituted by flows of tangible and intangible resources; these flows provide the capability for accessing further regimes of resources, for example in the form of knowledge, opportunity and competitive advantage. According to the theory of *dissipative structures,* when this resource flow moves to a far-from-equilibrium dynamic, whole-system structures can emerge through the process of self-organization. In coevolutionary terms, environmental changes can spark major organizational transitions.

Finally, *emergent evolution* provides a broad theoretical foundation for coevolution, by explaining the contingent differences in internal factors and external environments in terms of a continuous expansion of developmental capacities conditioned by localized constraints. Coevolutionary variation is represented by the emergence of new levels of self-organized order, which then undergo selection and retention according to the well-known processes of organizational evolution. As such, co-evolutionary change can be understood as a coherence of factors at multiple levels (individual, organizational, institutional), with an overall direction of increased information, communication, trust, interdependence, and managerial development.

IV. A CONCEPTUAL FRAMEWORK FOR CAS DESIGN

Elements of a complex system interact with each other. The actions of one element therefore affect other elements, directly or indirectly. These interactions can have negative, neutral, or positive effects on the system. Now, intuitively thinking, it may be that the "smoothing" of local interactions, i.e. the minimization of "interferences" or "friction" will lead to global improvement. To solve this problem, the design principles of multi-agent systems (Wooldridge, 2002) can be used. We can start with a basic definition: *An agent is a description of an entity that* acts *on its environment.* Examples of this can be a trader acting on a market, a school of fish acting on a coral reef, or a computer acting on a network. Thus, every element, and every system, can be seen as agents with *goals* and behaviors aiming to reach those goals. The behavior of agents can affect (positively, negatively, or neutrally) the fulfillment of the goals of other agents, thereby establishing a relation. The *satisfaction* or fulfillment of the goals of an agent can be represented using a variable $\sigma \in [0,1]$ (in some cases, σ could be seen as a fitness function). Relating this to the higher level, the satisfaction of a system σ_{sys} can be recursively represented as a function $f : R \rightarrow [0, 1]$ of the satisfaction of the n elements constituting it:

$$\sigma_{sys} = f(\sigma_1, \sigma_2, ..., \sigma_n, w_0, w_1, w_2, ..., w_n) \quad (1)$$

where w_0 is a bias and the other weights w_i determine the importance given to each σ_i. If the system is homogeneous and the components have linear interactions, then f will be the weighted sum of σ_i, with $w_i = 1/n, \forall\ i \neq 0$ and $w_0 = 0$. Note that this would be very similar to the activation function used in many artificial neural networks (Rojas, 1996). For heterogeneous systems, f may be a nonlinear function. Nevertheless, the weights w_i's are determined *tautologically* by the importance of the σ_i of each element to the satisfaction of the system σ_{sys}. If several elements decrease σ_{sys} as they increase their σ_i, we would not consider them as part of the system. It is important to note that this is independent of the potential nonlinearity of f. An example can be seen with the immune system (Dobrescu *et al.*, 2006). It categorizes molecules and micro-organisms as akin or alien. If they are considered as alien, they are attacked. Auto-immune diseases arise when this categorization is erroneous, and the immune system attacks vital elements of the organism. On the other hand, if pathogens are considered as part of the body, they are not attacked.

A reductionist approach would assume that maximizing the satisfaction of the elements of a system would also maximize the satisfaction of the system. However, this is not always the case, since some elements can "take advantage" of other elements. Thus, we need to concentrate *also* on the interactions of the elements. If the model of a system considers more than two levels, then the σ of higher levels will be recursively determined by the σ's of lower levels. However, the f's most probably will be very different on each level. Certainly, an important question remains: how do we determine the function f and the weights w_i's? To this question there is no complete answer. One option would be to approximate f numerically. An explicit f may be difficult to find, but an approximation can be very useful. Another method consists of *lesioning* the system: removing or altering elements of the system, and observing the effect on σ_{sys}. Dealing with complex systems, it is not feasible to tell each element what to do or how to do it, but their behaviors need to be constrained or modified so that their goals will be reached, blocking the goals of other elements as little as possible. These constraints can be called *mediators* (Michod, 2003). They can be imposed from the top down, developed from the bottom up, be part of the environment, or be embedded as an *aspect* of the system. Mediators are determined by an *observer*, and can be internal or external to the system (depending on where the observer sets the boundaries of the system). The efficiency of the mediator can be measured directly using σ_{sys}. Individually, we can measure the "friction" $\varphi_i \in [-1,1]$ that agent i causes in the rest of the system, relating the change in satisfaction $\Delta\sigma_i$ of element i and the change in satisfaction of the system $\Delta\sigma_{sys}$:

$$\varphi_i = (-\Delta\sigma_i - \Delta\sigma_{sys}(n-1))/n \qquad (2)$$

Friction occurs when the increase of satisfaction of one element causes a decrease in the satisfaction of some other elements that is greater than the increase. If $\varphi_i > 0$, it is because there was a noticeable decrease in σ_{sys}, or a disproportional decrease in σ_i. If $\varphi_i < 0$, then most probably it is due to an increase of σ_{sys}, or at least a noticeable in increase in σ_i with a negligible cost to the system. Negative friction would imply *synergy*, e.g. when $\Delta\sigma_i \geq 0$ while other elements also increase their σ. The role of a *mediator* would be to maximize σ_{sys} by minimizing φ_i's.

Concurrently, the *dependence* $\delta \in [-1,1]$ of an element to the system can be measured by calculating the difference of the satisfaction σ_i when the element interacts within the system and its satisfaction σ_i' when the element is isolated: $\delta=\sigma_i-\sigma_i'$. In this way, full dependence is given when the satisfaction of the element within the system σ_i is maximal and its satisfaction σ_i' is minimal when the element is isolated. A negative δ would imply that the element would be more satisfied on its own and is actually "enslaved" by the system.

Now we can use the dependencies of the elements to a system to measure the *integration* $\tau \in [-1,1]$ of a system, which can be seen also as a gradual measure of a meta-system transition (MST) (Turchin, 1977). A MST is a gradual process, but it will be complete when elements are not able to reach their goals on their own, i.e. $\sigma_i' \to 0$. In the next section, the steps suggested for developing a self-organizing system are presented, using the concepts described in this section.

V. A METHODOLOGY FOR CAS DESIGN

The proposed methodology receives the requirements of a system, i.e. what the system should do, and enables the designer to produce a system that fulfills the requirements. The methodology includes the following five steps: Representation, Modeling, Simulation, Application and Evaluation, These steps should not necessarily be followed one by one, since the stages merge with each other. The stages proposed are not new, and similar to those proposed by iterative and incremental development methodologies. The novelty of the methodology lies in the *taxonomy* used to describe self-organizing systems.

Representation. The goal of this step is to develop a *specification* (which might be tentative) of the components of the system. There are many possible representations of a system. According to the *constraints* and *requirements*, which may be incomplete, the designer should choose an appropriate vocabulary (metaphors to speak about the system), abstraction levels, granularity, variables, and interactions that need to be taken into account. The designer should try to divide a system into elements by identifying semi-independent modules, with internal goals and dynamics, and with few interactions with their environment. Since interactions in a model will increase the complexity of the model, we should group "clusters" of interacting variables into elements, and then study a minimal number of interactions between elements. The first constraints that help us are space and time. It is useful to group variables that are close to each other (i.e. interacting constantly) and consider them as elements that relate to other elements in occasional interactions. Multiscale analysis (Bar-Yam, 2005) is a promising method for identifying levels and variables useful in a Representation. Since the proposed methodology considers elements as agents, another useful criterion for delimiting them is the identification of goals. Dividing the system into *modules* also divides the problem it needs to solve, so a complex task will be able to be processed in parallel by different modules. Nevertheless, modularity in a system also increases its robustness and adaptability.

Modeling. In science, models should ideally be as simple as possible, and predict as much as possible. These models will provide a better understanding of a phenomenon than complicated models. The Modeling should specify a Control mechanism that will ensure that the system does what it is required to do. For example, since we are interested in self-organizing systems, the Control will be *internal* and *distributed*. But there are a lot of other attributes a Control must have, as we can see in the following.

The Control mechanism can be seen as a *mediator*, ensuring the proper interaction between elements of a system, and one that should produce the desired performance. However, one cannot have a strict control over a self-organizing system. To develop a Control, the designer should find aspect systems, subsystems, or constraints that will prevent the negative interferences between elements (friction) and promote positive interferences (synergy). In other words, the designer should search for ways of minimizing frictions φ_i's that will result in maximization of the global satisfaction σ_{sys}.

The Control mechanism should be *adaptive*. Since the system is dynamic and there are several interactions within the system and with its environment, the Control mechanism should be able to cope with the changes within and outside the system, in other words, be *robust*. An adaptive Control will be efficient in more contexts than a static one. In other words, the Control should be *active* in the search of solutions. A static Control will not be able to cope with the complexity of the system.

For a system to self-organize, its elements need to *communicate*: they need to "understand" what other elements, or mediators, "want" to tell them. This is easy if the interactions are simple: sensors can give *meaning* to the behaviors of other elements. But as interactions become more complex, the *cognition* required by the elements should also be increased, since they need to process more information. New meanings can be learned to adapt to the changing conditions.

The problem of *cooperation* has been widely studied. This will certainly reduce friction and therefore increase σ_{sys}. Elements of a system should *coordinate* while reducing friction, not to obstruct each other. An important aspect of coordination is the *division of labor*. This can promote synergy, since different elements can specialize in what they are good at and *trust* others to do what they are good at. A good Control will promote division of labor by mediating a balance between *specialization* and *integration*: elements should devote more time doing what they are best at, but should still take into account the rest of the system. Another aspect of coordination is the *workflow*: if some tasks are prerequisites of other tasks, a mediator should synchronize the agents to minimize waiting times.

As conclusions for the importance of Modeling, let consider several trade-offs that a system needs to reach a balance and cope with the complexity of its domain. We must keep in mind during the development of the system the following ratios between: Quality and Quantity, Economy and Redundancy, Homogeneity and Heterogeneity, Ability and Clarity, Generality and Particularity. There are only very relative ways of measuring the above mentioned trade-offs. However, they should be. While developing a particular system, the trade-offs will become clearer once the Simulation is underway. They can then be reconsidered and the Modeling updated.

Simulation. The aim of this stage is to build computer simulation that implement the model developed in the Modeling stage, and test different scenarios and mediator strategies. This is a key stage, since the precise behaviors of a complex system cannot be easily deduced from the Modeling, i.e. they are not reducible. The Simulation development should proceed in stages: from abstract to particular. First, an abstract scenario should be used to test the main concepts developed during the Modeling. Only when these are tested and refined, should details be included in the Simulation. This is because particular details take time to develop, and there is no sense in investing before knowing whether the Modeling is on the right track. Ideally, the designer should develop more than one Control to test in the simulation. The designer can then adjust or combine the Controls, and then compare again in the Simulation. A Simulation should be mature before taking the implementation into the real world.

Application. The role of this stage is basically to use the developed and tested models in a real system. If this is a software system, the transition will not be so difficult, since the software would have been already developed in the Simulation stage. Good theoretical solutions can be very difficult or too expensive to implement. The feasibility of the Application should be taken into account during the whole design process. In other words, the designer should have an *implementation bias* in all the Methodology stages. The *legacy* of previous systems should also be considered for the design: if the current implementation is to be modified but not completely replaced, the designer is limited by the capabilities of the old system. Usually, a pilot study should be made before engaging in a full Application, to detect incongruence and unexpected issues between the Simulation or Modeling stages and the Application.

Evaluation. Once the Application is underway, the performance of the new system should be measured and compared with the performances of the previous systems. Constraints permitting, efforts should be continued to try to improve the system, since the requirements it has to meet will certainly change with time (e.g. changes of demand, capacity, etc.). The system will be more adaptive if it does not consider its solution as the best once and for all and is able to change itself according to its performance and the changing requirements.

VI. CONCLUSIONS

The proposed Methodology will be useful for unpredictable and/or dynamic problem domains, where all the possible system's situations cannot be considered beforehand. It could also be useful for creative tasks, where a self-organizing system can explore the design space in an alternative way. The Methodology in principle is applicable to describe any system, but it would be redundant in simple or static problem domains, i.e. with a fixed solution, where adaptation is not required.

The proposed Methodology is not quite a spiral design model, because the last stage does not need to be reached to return to the first. That is, there is no need to deploy a working version (finish a cycle/iteration) before revisiting a previous stage, as for example in extreme programming. Rather, the Methodology is a "*backtracking* design model", where the designer can always return to previous stages. It is not necessary to understand a solution before testing it. In many cases understanding can come only after testing, i.e., the global behavior of the system is irreducible. Certainly, understanding the causes of a phenomenon will allow the modelers to have a greater control over it.

The backtracking between different steps in the Methodology is necessary because the behavior of the system cannot be predicted from the Modeling, i.e. it is not reducible. It might be possible to reason about all possible outcomes of simple systems, and then to implement the solution. But when complexity needs to be dealt with, a mutual feedback between experience and reasoning needs to be established, since reasoning alone cannot process all the information required to predict the behavior of a complex system. For this same reason, it would be preferable for the Control to be distributed.

Even when a central supercomputer could possibly solve a problem in real time, the information delay caused by data transmission and integration can reduce the efficiency of the system. Also, a distributed Control will be more robust, in as much as if a module malfunctions, the rest of the system can still provide reliable solutions. If a central Control fails, the whole system will stop working.

Since the behavior of a complex system in a complex environment cannot be predicted completely, the models need to be contrasted with experimentation before they can be validated. This Methodology suggests one possible path for finding solutions. The lack of predictability does not come only from chaotic processes. It might come also from new information generated by the interactions, so that the system behavior cannot be predicted from the behavior of the elements.

Finally, let answer if the proposed Methodology is a *top-down* or a *bottom-up* approach. Well, it is both and neither, since (at least) higher and lower levels of abstraction need to be considered simultaneously. In fact, the approach tests different local behaviors, and observes local and global performances, for local and global requirements. Thus, the Methodology can be seen as a *multi-level* approach and its architecture corresponds to the heterarchical structure shown in fig. 1. And also let notice that the Methodology strives to build artificial systems. Nevertheless, these could be used to understand natural systems using the synthetic method.

Therefore, the ideas presented here are potentially useful not only for engineering, but also for science. The backtracking ideology is also applicable to this Methodology: it will be improved once applied, through learning from experience.

ACKNOWLEDGEMENTS

This work is partially supported by Romanian Ministry of Education and Research under grants PN2_61031/2007 and ID_1032 no.125/2007

REFERENCES

[1] Baas, N. A. (1994) *Emergence, Hierarchies and Hyperstructures.* Artificial Life. Addison-Wesley

[2] Bar-Yam, Y. (2005). About engineering complex systems: Multiscale analysis and evolutionary engineering. In *Engineering Self Organising Sytems: Methodologies and Applications*, S. Brueckner, G. Serugendo-Di Marzo, A. Karageorgos, and R. Nagpal, (Eds.). Lecture Notes in Artificial Intelligence, vol. 3464. Springer, pp. 16–31.

[3] Dobrescu, R., Andone, D., Dobrescu, M. and Mocanu, S. (2006) Differential Equations Systems versus Scale Free Networks in Sepsis Modeling, *Proceedings of the 3rd IEEE Conference On Intelligent Systems (IEEE IS 06)*, London, pp.173-178

[4] Feltz, B., Crommelinck, M. and Goujon, P., Eds. (2006). *Self-organization and Emergence in Life Sciences*. Synthese Library, vol. 331. Springer.

[5] Haken, H. (1981). Synergetics and the problem of self organization. In *Self- Organizing Systems: An Interdisciplinary Approach*, G. Roth and H. Schwegler, (Eds.). Campus Verlag, pp. 9–13.

[6] Haken, H. (1988). *Information and Self-organization: A Macroscopic Approach to Complex Systems*. Springer-Verlag, Berlin.

[7] Heylighen, F. (2003). The science of self-organization and adaptivity. In *The Encyclopedia of Life Support Systems*, L. D. Kiel, (Ed.). EOLSS Publishers, Oxford.

[8] Jones, P. M., Contractor, N., O'keefe, B. and Lu, S. C.-Y. (1994). Competence models and self-organizing systems: Towards intelligent, evolvable, collaborative support. In *Systems,Man, and Cybernetics, 1994. 'Humans, Information and Technology'., 1994 IEEE International Conference on*. Vol. 1. pp. 367–372.

[9] Kohonen, T. (2000). *Self-Organizing Maps*, 3rd ed. Springer

[10] Lucas, C. (1999) *Quantifying Complexity Theory*. CALResCo, www.calresco.org

[11] Michod, R. E. (2003). Cooperation and conflict mediation during the origin of multicellularity. In *Genetic and Cultural Evolution of Cooperation*, P. Hammerstein, (Ed.). MIT Press, Cambridge, Chapter 16, pp. 261–307.

[12] Nicolis, G. and Prigogine, I. (1977). *Self-Organization in Non-Equilibrium Systems: From Dissipative Structures to Order Through Fluctuations*. Wiley.

[13] Ramamoorthy, P., Zhang, S., Fubao, C. and Ramachandran, D. (1993). A new paradigm for the design of nonlinear dynamical systems and self-organizing systems. In *Proceedings of the 1993 IEEE International Symposium on Intelligent Control,* pp. 571–576.

[14] Rojas, R. (1996). *Neural Networks: A Systematic Introduction.* Springer, Berlin.

[15] Schweitzer, F., Ed. (1997). *Self-Organization of Complex Structures: From Individual to Collective Dynamics*. Gordon and Breach.

[16] Steels, L. (2003). Evolving grounded communication for robots. *Trends in Cognitive Science* **7** (7), pp.308–312.

[17] Turchin, V. (1977). *The Phenomenon of Science. A Cybernetic Approach to Human Evolution*. Columbia University Press, New York.

[18] Wooldridge, M. (2002). *An Introduction to MultiAgent Systems*. John Wiley and Sons, Chichester, England.

[19] Zambonelli, F. and Rana, O. F. (2005). Self-organization in distributed systems engineering: Introduction to the special issue. *Systems, Man and Cybernetics, Part A, IEEE Transactions on* **35** (3) (May), pp. 313–315.