Bio-Inspired Approaches for Autonomic Pervasive Computing Systems

Daniele Miorandi[1], Iacopo Carreras[1], Eitan Altman[2], Lidia Yamamoto[3], and Imrich Chlamtac[1]

[1] CREATE-NET, via alla Cascata, 56/C, 38100 – Povo, Trento Italy
name.surname@create-net.org
[2] INRIA, 2004 Route des Lucioles - BP 93, 06902 – Sophia Antipolis France
eitan.altman@sophia.inria.fr
[3] Computer Science Department, University of Basel, Bernoullistrasse 16, 4056 – Basel Switzerland
lidia.yamamoto@unibas.ch

Abstract. In this chapter, we present some of the biologically-inspired approaches, developed within the context of the European project BIONETS for enabling autonomic pervasive computing environments. The set of problems addressed include networking as well as service management issues. The approach pursued is based on the use of evolutionary techniques — properly embedded in the system components — as a means to achieve fully autonomic behaviour.

Keywords: pervasive computing, biologically-inspiread design paradigms, protocol evolution, chemical computing.

1 Introduction

The Future Internet will be characterized by scale, heterogeneity, complexity and dynamicity figures that call for novel approaches to the design and the management of computing and communication systems. This will enforce a shift from conventional "top-down" engineering approaches, in which systems' blueprints are designed, optimized and engineered to perform a well-defined task, to "bottom-up" approaches, in which systems will be provided with the necessary means for growing and evolving in an unsupervised manner. The final goal is to enable autonomic systems, which are able to self-manage themselves, requiring human intervention only in the definition of the high-level goals to be pursued.

The BIONETS project (Biologically-inspired Autonomic Networks and Services, www.bionets.eu), funded in the framework of the EC-FET initiative on Situated and Autonomic Communications, targets the introduction of biologically-inspired approaches for designing and managing pervasive computing and communication environments. The conceptual basis of the project is rooted in the observation that nature has been successfully tackling the aforementioned problems (scale, heterogeneity, complexity and dynamicity), leading to complex ecosystems which are able to self-sustain and to reach efficient equilibria in the absence of a central control.

P. Liò et al. (Eds.): BIOWIRE 2007, LNCS 5151, pp. 217–228, 2008.

In this chapter, we review some paradigms, inspired by natural (and in particular biological) systems' functioning, which could be successfully applied to architect pervasive computing/communications environments. We also discuss some of the issues to be faced when trying to engineer such paradigms into technological artifacts, presenting some examples drawn from the research activities carried out within the BIONETS project framework.

The remainder of the chapter is organized as follows. In Sec. 2 we discuss the challenges stemming from Future Internet scenarios, highlighting the need for embedding autonomic properties in computing/communication systems. In Sec. 3 we survey some potentially useful paradigms, inspired by the operations and functioning of various natural (mostly biological) systems. In Sec. 4 we discuss issues related to the use of such paradigms for designing computing/communication systems, presenting two examples taken from project's activities. Sec. 5 concludes the paper discussing promising applications of the presented paradigms.

2 Scenarios for Future Internet and the Need of Autonomicity

If in the 80s and 90s the Internet was still conceived as an "information highway", i.e., a set of physical links where myriads of data and packets were flowing, carrying the most disparate information, things are changing rapidly. The change concerns not just what users are doing with those data/packets, but the nature of the system itself. On the one hand, the Internet has become a vital ganglion of the globalised economy and society. On the other one, a lot is happening at the edges of the Internet as we have known it. Progresses in microelectronics and nanotechnologies are about to lead to situations in which electronics (including ability to communicate and to compute) gets embedded in a variety of common objects. The net results is expected to be an invisible digital halo, surrounding users and supporting them in their daily life. While this phenomenon is referred to in various ways, depending from the viewpoint (from pervasive computing environments [1] to smart spaces up to "Internet of things"), there is a general consensus that this is the direction we are moving to. This trend is bringing with it a set of challenges to current information and communication technologies and system architectures, requiring radical changes in the approaches conventionally pursued.

The first and most apparent problem is *scale*. Scale in the number of potentially connected devices as well as in the number of users and of services to be supported. Are currently solutions adequate? Can we rely on Moore laws for bandwidth/computing power/storage capacity to ensure we will be able to cope with scale? The answer is probably negative. The reason lies in the fundamentals of the current IP network architecture (which represents the backbone of almost all networked systems), which rely on an address-oriented end-to-end paradigm, assuming always-on connectivity. This incurs limitations in both the finiteness of the address space as well as problems related to the scalability of connected large-scale wireless networks.

The second problem is *heterogeneity*. Again, this is rooted in the hourglass model at the basis of the Internet TCP/IP protocol suite, where the IP acts as a "glue" between various subsystems. It is probably necessary now to rethink it, in such a way to accommodate heterogeneity and pluralism in the system, in terms of both nodes [2] and network architecture [3].

The third problem is *complexity*. Network and service management and maintenance is becoming a harder and harder job, requiring an extremely large amount of operators' time, with the consequent negative fallouts on the economic side. And with increase in scale and heterogeneity this is probably going to become the real ceiling limiting the ability to produce innovation in the ICT field.

The fourth problem is *dynamicity*. The ICT world is experiencing innovations at an extremely high rate. New technologies and services are created and disappear continuously, and new ways of profiting from the Internet are been envisaged and introduced seamlessly. There is an increasing need to design ICT systems which are able to be, in some sense, future-proof, i.e., able to plastically adapt to changing environments, services and users needs.

These four challenges motivated the research on methods and tools for building autonomic communication systems [4], in much the same way it had been proposed by IBM in the computing field [5]. The basic problem lies in the fact that current ICT systems are conceived as static ones. The ability to adapt is — in most cases — present, but in a limited form, and it is decided *a priori* in the design phase. In some sense, adaptability is "hardwired" into the system's blueprint, which is, however, unable to adapt and change. Traditional "direct engineering" approaches have the great advantage of engineering "by design" the desired system behaviour, but, at the same time, they limit the ability of systems to adapt or optimize to unforeseen scenarios.

The BIONETS project, on whose activities this chapter is mostly centred, aims at addressing such issues by looking at how nature (and biology in particular) has led to the arising of complex ecosystems, able to achieve the self-CHOP features of IBM's autonomic computing manifesto (self-configuration, self-healing, self-optimization, self-protection) [5] through open-ended evolution.

3 Nature-Inspired System Design Paradigms

Nature presents a variety of examples of systems that are able to successfully deal with scale, heterogeneity and complexity figures similar to those expected for pervasive computing environments. As far as dynamicity is concerned, things are slightly different, in that many natural phenomena (e.g., evolution) take effect over long time periods. On the other hand, in an artificial system the rate at which such phenomena can be emulated depends heavily on the available computing power.

In the early phase of the BIONETS project, a set of paradigms inspired by biological, physical and social phenomena were identified and studied [6]. We present in the following a short review of three of the most relevant paradigms

identified: chemical computing, embryology and evolutionary game theory. With respect to the aforementioned issues, chemical computing provides insight into the design of computing systems able to continuously change to self-optimize to current system conditions (dynamicity). Embryology may provide means for applying evolutionary computation methods to extremely complex systems (scalability, complexity), enabling at the same time run-time optimization (dynamicity). Evolutionary game theory provides means to analyze and design systems able to work unsupervised on the basis of local interactions (scalability, complexity).

3.1 Chemical Computing

The term *Chemical computing* [7,8] refers to two distinct areas: *(i) real chemical computing*: computing with real molecules, such as DNA computing; *(ii) artificial chemical computing*: hardware and software architectures inspired by chemistry. In the case of software, it refers to computation models following a chemical metaphor, which run on regular von Neumann computers. The scope of this section is restricted to the latter.

Chemical computing can be regarded as a branch of *Artificial Chemistry* [8], the subfield of Artificial Life devoted to modelling the dynamics of chemical phenomena in order to understand the origin and evolution of organizations in general, and life in particular. The term *artificial chemistry* also refers to the specific chemical model used, defined by the molecular species involved, the reaction rules, and the algorithm for the reaction vessel.

Numerous artificial chemistry models have been proposed. A model in which molecules are λ-calculus expressions is presented in [9], showing conditions for the emergence of self-maintaining organizations out of an initial "soup" of random molecules. The chemical reaction model is catalytic (i.e. reactants are conserved after the reaction), and mass conservation is ensured by a dilution flux. This line of research led to a theory of chemical organizations [10], with several applications in biology and computer science.

We believe that chemical models have a great potential for on-line evolution in autonomic systems. This can be illustrated by the following example: In [11] Genetic Programming (GP) is applied to an algorithmic chemistry in which instructions are drawn from a multiset and executed in random order. Starting from a nearly unpredictable system, a few generations later programs exhibit highly reproducible results. The system is therefore able to evolve programs that are robust to random execution order. The authors point out the importance of the concentration of instructions, rather than their sequence. Indeed, in the solutions evolved, the concentrations of the instructions that are the most crucial to the solution are higher, and instructions that are not relevant end up with low or no concentration at all.

Chemical models raise many new questions as well. Methodologies for engineering, programming or evolving chemical reaction networks are only now emerging, for natural or artificial systems. Most of the models have a stochastic nature, which makes them inherently non-deterministic, and also more complex

than traditional top-down, human-made solutions. On the other hand, we believe that making progress in this area will greatly improve our understanding of computational models close to biology, and of life in general.

3.2 Artificial Embryogenies

The application of ideas from embryology to artificial systems has been following two main research directions. The first one is *embryonics* (embryology plus electronics), an approach to improve fault tolerance in evolvable hardware by using a cellular architecture presenting dynamic self-repair and reproduction properties [12]. Approaches in this area have mostly focused on the use of *artificial stem cells*, i.e., cells which are able to differentiate into any specific kind of cell required for the organism to work. The systems devised in such way are based on the following two principles:

- Each cell contains the whole genome, i.e., the complete set of rules necessary for the organism to work and is totipotent, i.e., can differentiate into any specific function.
- The system presents self-organizing properties. Each cell monitors its neighborhood and may return to the stem cell state and differentiate into another type of cell to repair a fault detected.

The flexibility to switch functionality adds another level of robustness with respect to conventional approaches, as now not only cells with identical functionality can be used as backup or template to repair a failure, but also other cells with differentiated functionality can be used to recreate a lost one.

The second one is *artificial embryogeny* [13], which aims at extending evolutionary computing with a developmental process inspired by embryo growth and cell differentiation, such that relatively simple genotypes with a compact representation may express a wide range of phenotypes or behaviors. Indeed, researchers have recognized that "conventional" EC techniques (like GA, GP, Evolutionary Strategies, etc.) present scalability problems when dealing with problems of relevant complexity. In artificial embryogenies the genotype does not code the solution itself, but it codes recipes for building solutions (i.e., phenotypes). This can lead to a non-linear genotype-to-phenotype mapping, which may also be affected by environmental variables. In this way, a genotype change does not imply a direct change in the solution, but in the way solutions are decoded from the genotype and further grown from an initial "seed" (the embryo).

3.3 Evolutionary Games

In autonomic pervasive computing systems, users compete for resources; decisions related to flow control, to routing, to accessing common channels etc. are not under control of a central entity. Non-cooperative Game Theory [14] has naturally become a very popular paradigm for modeling the decentralized decision

making. The high complexity of dynamic computing, information and communication systems suggests that one should search for models and concepts that involve large populations, dynamics, adaptation and evolution.

Equilibria concepts (the Nash and the correlated equilibrium, the Wardrop equilibrium in road traffic engineering, and the Evolutionary Stable Strategy in evolutionary games) describe relatively "static" situations, in which those involved in the game are relatively satisfied: they cannot be better off by unilaterally deviating. We may expect however that many situations of competition in autonomous complex networks are dynamic ones. Tools are needed to understand the evolution of competition, the way one converges to, or diverges from, equilibria. Such understanding can then be very useful in designing mechanisms for evolution of services related to autonomic pervasive computing systems. The evolutionary game paradigm was created by J. Maynard Smith [15,16] in a context of conflicts and competition among populations in biological complex systems. It provides tools to describe the competition dynamics between populations through differential equations that are called replicator dynamics and which relate fitness of species with their growth rate [17].

The TCP congestion control protocol is an example of a distributed network mechanism that allows flows to adjust their transmission rate in a completely decentralized way. It has had many variants that differ from each other by the degree of aggressiveness. The first protocols were the most aggressive ones and have caused catastrophic events known as congestion collapse in the Internet. These protocols have disappeared, replaced by the very gentle, non-aggressive protocol, Tahoe, that has disappeared too. New TCP protocols keep appearing. In [18], an elementary model from Evolutionary game theory, called the "Hawk and Dove game" is used to predict both equilibrium behavior between aggressive and friendly TCP versions, as well as non equilibrium behavior. It is shown that depending on system's parameters, one of two types of equilibrium can emerge: (i) One in which the two types of TCP coexist (the actual percentage of each one are given there) or (ii) One in which only the aggressive TCP survives.

A stability condition is then derived. An oscillatory behavior is identified when the system is unstable. The oscillations are perceived as instability since they do not allow the system to attain the equilibrium point that may well exist under the same set of parameters. Oscillations in population sizes are also found in the context of competition between species in biology. The difference with biology is, however, that in networks we can use our understanding of evolutionary behaviors to achieve stability and suppress oscillatory behavior. Guidelines for achieving stable behavior have been proposed in [18] in terms of delays in the system as well as some gain parameter that controls the rate of adaptation. More sophisticated stability analysis can be found in [19,20].

4 Embedding Nature-Inspired Strategies

The paradigms presented in the previous section represent promising models for building autonomic computing/communications systems. Nonetheless, for some

of them (e.g., embryology), the current status of research does not allow direct application to real-world problems. On the other hand, some of the applications of nature-inspired techniques developed in BIONETS have relied on variants of tools and techniques from evolutionary computation [21].

In this section, we report two case studies related to evolution of network protocols. The aim of this section is to present some of the most relevant issues to be faced when moving from the paradigm to the application, together with a series of lessons learned from the experiments performed.

4.1 Case Study: Evolving Protocols with the Fraglets Language

In this section we report our experience with evolving network protocols using the Fraglets language [22], a programming inspired by chemical computing, and targeted at network protocols.

A *fraglet* [22] is a "virtual molecule", that represents a computation fragment as a string of symbols $[s_1\ s_2\ \ldots\ s_n]$. A fraglet may contain data, reaction rules involving two fraglets, or transformations of a single fraglet. Fraglets are injected for execution into a virtual reaction vessel which contains a multiset of fraglets.

The fraglets language has been designed for network protocols. For this purpose the rule processing engine is based on the "tag matching" principle: fraglets are processed according to their head symbol, which is consumed in the process, similar to protocol header processing in network packet streams.

The fraglet instruction set includes a few reaction rules and several transformations. As an example, the `match` reaction rule has the form `[match` $s\ tail_1$`]`. It reacts with any fraglet of the form `[`$s\ tail_2$`]` (i.e. which starts with the matching symbol s), and produces $[tail_1\ tail_2]$, i.e. the concatenation of the two tails. Other examples of transformation rules are: `dup` which duplicates a symbol, and `exch` which swaps two symbols.

The fraglets language had been originally proposed for the automatic synthesis and evolution of protocol implementations [22]. However, no actual results showing automatic evolution had been shown. We therefore performed some first GP experiments to evolve reliable transmission protocols in fraglets [23]. After several unsuccessful trials, feasible programs could finally be obtained via a primitive form of homologous recombination, which consisted in inserting pre-defined markers by hand at given points in the code, where recombination was allowed to occur. With this simple technique the system was able to find the optimum protocol for a given environment by combining existing marked modules.

The main weakness of [23] was its inability to actually produce any viable novelty: only individuals that were recombinations of existing successful ones were also successful. New individuals obtained by random mutation were either infeasible or had poor performance. We then started investigating the reasons for such difficulties in evolving fraglet programs, in contrast with the undeniable success of standard GP techniques based on trees or linear programs. Fraglets are also linear programs, therefore we had expected that it would be easy to evolve fraglet programs as it is to evolve linear ones. However, the header matching

pattern turns out to be extremely constraining to the evolution process: a random mutation in a program could easily lead to tags never being matched. And in many cases, one such tag would suffice to block the whole program.

Further obstacles to evolution were later discovered: for instance, when trying to evolve programs out of a "primordial soup" of randomly generated fraglets, the system most of the times ended up clogged by a mass of instructions that were not executable, because a corresponding matching tag could never be found: that was the case of a `[match exch ...]` or `[match dup ...]`. The fraglet interpreter does not allow matching on rule keywords, therefore such fraglets can neither match nor be eliminated (since a `[match match]` is not allowed either). So they pollute the system forever.

Moreover, the original fraglets reaction algorithm contained a non-standard "smallest" criterion: the probability of choosing a given reaction was proportional to the smallest concentration among each of its reactants. The side effect of this rule was that the balance of molecule concentrations did not have the desired effect on the reaction probabilities that would be expected from a chemical model. As a consequence, well-known techniques from systems biology (e.g., based on stoichiometric analysis or differential equations) could not be applied. As a result, it was difficult to implement an effective code regulation mechanism [24] aimed at enforcing good genotypes and eliminating bad ones, based on the control of molecule concentrations. Some modest results on code regulation were achieved [24], but the extension of that model to more complex cases clearly showed limitations.

In spite of all these difficulties, we still believe that there is a potential in chemical models for on-line program evolution, which remains unexplored: Chemical programs are inherently parallel, robust to random execution order, and can naturally support multiple alternative execution flows.

We are now in the process of fully redesigning the fraglets language: its molecule format, instruction set and reaction algorithm, with three aims. First, programs should be fully self-modifiable: it should only be possible to generate rules that may be later eliminated. Second, the language should become plainly suited to GP, by maximizing the chance of obtaining valid programs, both syntactically and semantically. Third, it should be possible to control the concentration of instructions using closed feedback loops which monitor the system, promote good programs and eliminate bad ones.

With respect to the third goal above, we have enhanced fraglets with variants of the Gillespie algorithm, a well-known algorithm for simulating the dynamics of a real-world chemical reactor tank. Experiments with the new algorithm confirm that concentration dynamics now match the expected chemistry patterns.

Concerning the first and second goals above (self-modification and GP orientation), the modifications needed are so radical that the outcome might be an entirely new language. The basic principles however stay the same: A programming language based on the chemical reaction metaphor, in which programs are expressed as a set of virtual molecules in the form of strings that can react with each other. Virtual molecules express code and data in a uniform way. Since

molecules may be long strings, it is important to choose reactants based on short keys that can be quickly looked up in a hash table, making the implementation of the chemical reaction algorithm feasible in terms of computation time. Moreover, each chemical reaction should take a short time to execute, like a thread that is scheduled for execution for a short time and then preempted.

In the new language [25], random access to any position in a fraglet will be allowed, and stack-based operations will be possible: one fraglet will be able to act as a data repository for another, accessible as a vector or as a stack, upon reaction. Since code and data are represented in a uniform way, the program can produce further code by writing on the data structure, which can then be executed by simply removing the head symbol. Rules will be able to act on other rules, such that they can always be created and eliminated, leading to fully self-modifiable code. A semantic will be assigned to each possible reaction rule, such that it can always be executed in a valid way, no matter the amount of parameters available and their types. This should enable smoother GP runs.

Given the key length restriction described above, the tag matching problem will somehow remain: the alternative would be to consider every different string as a separate molecular species, which would not be scalable. This problem is intrinsic to any artificial polymer chemistry, in which chemical species may be arbitrarily long molecular chains, and which assumes a well-stirred solution (such that any molecule can potentially collide with any other). However the problem might be significantly reduced if: first, any symbol may form a valid key, including reserved keywords of the language, numbers, etc.: this enhances the space of valid programs; second, the same key may be reused several times during sequential operations: it is easier to automatically generate a program sequence based on a single key, rather than a chain of interconnected keys as currently required in fraglets.

4.2 Case Study: Evolutionary Epidemic Dissemination Mechanisms

Epidemic-style forwarding [26] has been proposed as an approach for achieving message delivery in intermittently connected wireless ad hoc networks, also referred to as Delay-Tolerant Networks (DTNs) [27]. Such environments are characterized by a high degree of dynamism and by the unpredictable nature of the contact patterns, which make standard ad hoc wireless networks routing protocols unsuitable. Epidemic-style forwarding in DTNs is based on a "store-carry-forward" paradigm: a node receiving a message buffers it and carries it around, passing it on to new nodes upon encounter. Each time a node encounters a peer not having a copy of one message it carries, the carrier may decide to *infect* this new node by passing on a message copy. The message gets delivered when the destination first meets an infected node.

In a DTN scenario, the choice of a forwarding scheme and of its set of running parameters depend on a set of factors (mobility, traffic patterns etc.) which are — in general — not known at design time, and may significantly change over time and space. Standard adaptive techniques are limited in that they require an *a priori* definition of the actions to be taken in response to some external

stimuli. The mechanism proposed in [28], on the other hand, is based on the use of concepts and tools from the Genetic Algorithms field.

In the proposed implementation, each node in the system employs its own forwarding policy, determining the actions to be taken upon the reception of a message destined to another node. The genotype in such system is represented by an array of parameters, defining the system's behavior. In the considered case-study, we considered as relevant parameters the probability of forwarding a message upon a contact with a susceptible node, as well as the maximum number of hops traversed by a message. (In general, however, there is no limitation on the number of parameters that can encoded.) Each genotype is associated with a fitness level which describes its ability to contribute to the general system's functioning. Upon encountering, two nodes may exchange information on the genotype current in use and the respective fitness level. Each node maintains a pool which contains all available information on genotypes and associated fitness levels. Such set of genotypes is used to generate new ones periodically, applying standard GA operators (crossover and mutation) to two genotypes selected with probability proportional to their fitness level.

Message delivery can be regarded as a *distributed* service, which require multiple entities to cooperate in order to achieve the desired goal. As a direct consequence, there are two main difficulties to be handled. The first one is related to the estimation of the fitness level associated to each genotype in use, which needs to be performed locally relying on partial information only. The second one is related to the fact that, in general, there is no way of controlling the behavior of the (other) nodes contributing to the delivery process. There is also a further subtlety which is worth being considered. In this case indeed, the evolutionary process is deeply intertwined with the mechanisms used for achieving communications (which represent a key component of a distributed evolutionary framework). This coupling introduces a bias in the fitness estimation process, which turns out to have a notable influence on the mechanism behaviour.

While the proposed mechanism has been shown, through extensive numerical simulations, to perform well over a wide range of operating conditions, some problems were encountered which represent as many lessons learned on the engineering of distributed evolutionary mechanisms. The main issue to be faced is the problem of mapping a global optimization problem to a distributed localized one, in which many local entities perform each its own optimization process, based on information available locally only. While in the proposed solution this was done in a rather ad hoc way, there are good chances that results from game theory can be used to provide a framework for this mapping from global optimization to local decisions. Another issue of considerable impact is the definition of suitable mechanisms for cascade fitness estimation in distributed services. In the considered service, indeed, all intermediate nodes along the path followed by the message from the source to the destination contribute to the performance of the global service (end-to-end delivery). The fitness of the overall service is computed by the two end-points, but the problem is then to decide how to reward nodes along the delivery path. A general framework for such problem is

still missing. Last, the proposed solution suffer from a bias in the estimation of the delay. While this is a general sampling problem (due to the use of a finite observation window), it turns out to have a rather considerable impact on the behavior of the mechanism.

5 Conclusions

In this chapter, we have presented an overview of the bio-inspired lines of research developed within the framework of the BIONETS project for building autonomic pervasive communications/computing environments. Three of the most promising paradigms identified have been briefly presented, together with two examples reporting the difficulties to be faced when moving to the application phase.

In general, much remains to be done in order to engineer autonomic pervasive computing systems by applying bio-inspired approaches. The main limitations appear to be related to the problems encountered when trying to reproduce in a computable medium natural phenomena. While living eco-systems can indeed be regarded as a special form of distributed computing, our ability to reproduce such processes (with all their positive features) in a computing system is still limited. Attempts to bridge research communities working on biology, ecology and computer sciences appear to bring great potential for architecting the key technologies to build the Future Internet.

Acknowledgments

This work has been partially supported by the European Commission within the framework of the BIONETS project, IST-FET-SAC-FP6-027748, `www.bionets.eu`. The authors are grateful to S. Alouf, G. Neglia and A. Fialho for the discussions on the evolutionary forwarding case-study and to T. Meyer for the joint work on reaction algorithms for fraglets.

References

1. Weiser, M.: The computer for the 21st century. SIGMOBILE Mob. Comput. Commun. Rev. 3(3), 3–11 (1999)
2. Carreras, I., Chlamtac, I., Pellegrini, F.D., Miorandi, D.: BIONETS: Bio-inspired networking for pervasive communication environments. IEEE Trans. Veh. Tech. 56, 218–229 (2007)
3. Crowcroft, J., Hand, S., Mortier, R., Roscoe, T., Warfield, A.: Plutarch: an argument for network pluralism. In: Proc. of ACM SIGCOMM, Karlsruhe, DE (2003)
4. Dobson, S., Denazis, S., Fernandez, A., Gaiti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., Zambonelli, F.: A survey of autonomic communications. ACM Trans. Aut. Adapt. Syst. 1, 223–259 (2006)
5. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. IEEE Comp. Mag. 36(1), 41–50 (2003)
6. Altman, E., Dini, P., Miorandi, D., Schreckling, D.: Paradigms and foundations of BIONETS research, `http://www.bionets.eu/docs/BIONETS_D2_1_1.pdf`

7. Dittrich, P.: Chemical Computing. In: Banâtre, J.-P., Fradet, P., Giavitto, J.-L., Michel, O. (eds.) UPP 2004. LNCS, vol. 3566, pp. 19–32. Springer, Heidelberg (2005)
8. Dittrich, P., Ziegler, J., Banzhaf, W.: Artificial Chemistries – A Review. Artificial Life 7(3), 225–275 (2001)
9. Fontana, W., Buss, L.W.: The Arrival of the Fittest: Toward a Theory of Biological Organization. Bulletin of Mathematical Biology 56, 1–64 (1994)
10. Dittrich, P., di Fenizio, P.S.: Chemical organization theory: towards a theory of constructive dynamical systems. Bulletin of Mathematical Biology 69(4), 1199–1231 (2005)
11. Banzhaf, W., Lasarczyk, C.: Genetic Programming of an Algorithmic Chemistry. In: O'Reilly, et al. (eds.) Genetic Programming Theory and Practice II, vol. 8, pp. 175–190. Kluwer/Springer (2004)
12. Prodan, L., Tempesti, G., Mange, D., Stauffer, A.: Embryonics: artificial stem cells. In: Proc. of ALife VIII, pp. 101–105 (2002)
13. Stanley, K.O., Miikkulainen, R.: A taxonomy for artificial embryogeny. Artif. Life 9, 93–130 (2003)
14. Fudenberg, D., Tirole, J.: Game Theory. MIT Press, Cambridge (1991)
15. Maynard Smith, J.: Game theory and the evolution of fighting. In: Maynard Smith, J. (ed.) On Evolution, pp. 8–28. Edinburgh University Press (1972)
16. Maynard Smith, J.: Evolution and the Theory of Games. Cambridge University Press, Cambridge (1982)
17. Hofbauer, J., Sigmund, K.: Evolutionary Games andvPopulation Dynamics. Cambridge University Press, Cambridge (1998)
18. Tembine, H., Altman, E., El-Azouzi, R., Hayel, Y.: Evolutionary games for predicting the evolution and adaptation of wireless protocols submitted
19. Tembine, H., Altman, E., El-Azouzi, R.: Delayed evolutionary game dynamics applied to the medium access control. In: Proc. of IEEE BioNetworks, Pisa, IT (2007)
20. Tembine, H., Altman, E., El-Azouzi, R.: Asymmetric delay in evolutionary games. In: Proc. of ValueTools, Nantes, FR (October 2007)
21. Foster, J.A.: Evolutionary computation. Nature 2, 428–436 (2001)
22. Tschudin, C.: Fraglets - a metabolistic execution model for communication protocols. In: Proc. of AINS, Menlo Park, USA (July 2003)
23. Yamamoto, L., Tschudin, C.: Experiments on the Automatic Evolution of Protocols using Genetic Programming. In: Stavrakakis, I., Smirnov, M. (eds.) WAC 2005. LNCS, vol. 3854, pp. 13–28. Springer, Heidelberg (2006)
24. Yamamoto, L.: Code Regulation in Open Ended Evolution. In: Ebner, M., O'Neill, M., Ekárt, A., Vanneschi, L., Esparcia-Alcázar, A.I. (eds.) EuroGP 2007. LNCS, vol. 4445, pp. 271–280. Springer, Heidelberg (2007)
25. Yamamoto, L.: PlasmidPL: A plasmid-inspired language for genetic programming. In: Proc. of EuroGP, Napoli, IT (2008)
26. Vahdat, A., Becker, D.: Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University (April 2000)
27. Fall, K.: A delay-tolerant network architecture for challenged Internets. In: Proc. of ACM SIGCOMM, Karlsruhe, Germany, pp. 27–34 (2003)
28. Alouf, S., Carreras, I., Miorandi, D., Neglia, G.: Embedding evolution in epidemic-style forwarding. In: Proc. of IEEE BioNetworks, Pisa, IT (2007)