Exploring Adaptation & Self-Adaptation in Autonomic Computing Systems

M.T.Ibrahim[1], R.J.Anthony[1], T. Eymann[2],
A.Taleb-Bendiab[3] and L.Gruenwald [4] **

[1]*University of Greenwich UK,* [2]*University of Bayreuth Germany,*
[3]*University of Liverpool John Moore's UK,* [4]*NSF USA*
m.t.ibrahim@gre.ac.uk, r.j.anthony@gre.ac.uk, torsten.eymann@uni-bayreuth.de,
A.Talebbendiab@livjm.ac.uk, lgruenwa@nsf.gov

Abstract

This panel paper sets out to discuss what self-adaptation means, and to explore the extent to which current autonomic systems exhibit truly self-adaptive behaviour. Many of the currently cited examples are clearly adaptive, but debate remains as to what extent they are simply following prescribed adaptation rules within pre-set bounds, and to what extent they have the ability to truly learn new behaviour. Is there a standard test that can be applied to differentiate? Is adaptive behaviour sufficient anyway? Other autonomic computing issues are also discussed.

** *Acknowledgement: "Le Gruenwald's work is partially supported by (while serving at) National Science Foundation."*

1. Introduction and background

As the autonomic concepts and terminology are starting to reach critical mass, having pervaded into many corners of computer science research and application development, this paper attempts to consider some fundamental questions concerning the nature and state of the art of some aspects of adaptivity in autonomic computing systems (ACS).

Autonomics certainly is a popular buzzword nowadays. There are a large number of autonomics conferences, several autonomics research groups, and a large number of self-* applications under development. However, to what extent is the autonomics label a marketing 'hype' for adaptive behaviour that pre-dates the latest jargon, and to what extent have applications become truly self-adaptive? In answering this very general question, in the next paragraph, we define a number of more clearly scoped questions.

How adaptive are current Autonomic Computing Systems? Do ACS adapt or self-adapt? To what extent are they simply following prescribed adaptation rules within pre-set bounds or do they have the ability to truly learn new behaviour?

2. Panel contributions

A panel of active academics/practitioners debate these and other related questions. The workshop audience (and readers) are expected to actively participate in the debate and bring their own experiences to bear on the panel discussions.

2.1 Ibrahim believes that this paper, as all means of communication do, aims primarily to communicate a message to its readers. It's hoped that its message, or rather perhaps many messages, are understood. Unfortunately, it is not always true that communicated messages are understood for obvious reasons [1].

Over the recent past, many conferences addressed topics relating to communication and understanding. More recently, much research work and standardisation efforts has been directed towards ontology, semantics and metadata [6, 7, 8]. No wonder then that, in common with many other disciplines and domains, the Information/IT community including Autonomic Computing fraternity are badly in need of open standards [9] to enable concepts and terms such as those appearing in the title and section 1 as well many others to be properly communicated and understood.

Having made the above observations, and given that there is no universal agreement on autonomic ontology, I further believe that whether a computing system is adaptive or self-adaptive is, in this author's view, not one of primary concern at a high level of abstraction and/or architectural layers.

Adaptive features, as well as many other self-* ones associated with autonomic computing, are solutions to one or more real world problems in one or more domains spanning intra or inter-domain applications. It all depends on the different stakeholders' roles in the development process and their own perspective. The issue of communicating messages and understanding between parts of a system (user-user, user-computer or computer-computer) need to be carefully considered if we are to achieve the ultimate aim of the fully mature autonomic level 5, [9], see Figure 1 below. This part, from now on, focuses mainly on computing systems and not natural systems unless specifically stated.

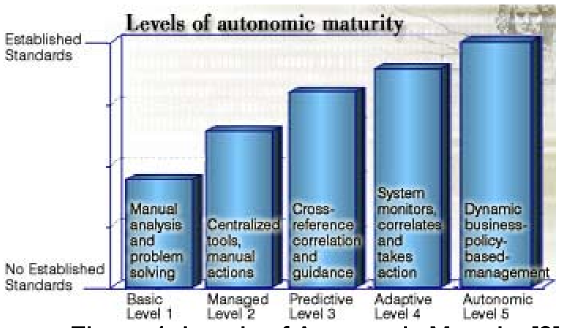

Figure 1: Levels of Autonomic Maturity [9]

Nowadays, the overwhelming majority of systems, if not all, have many 'sub'systems including computing (sub)systems as well as communication (sub)systems and so on. For clarity, it should be pointed out that I am using the term 'computing system' in this part, to mean an integrated system inclusive of computing and communication systems and not just the former. If a computing system can be designed to be adaptive then it is possible, in most if not all cases, to be designed to be self-adaptive. A question we are addressing is where are we now? Well, we are not at level 5 yet, but and to the best of my knowledge, I posit that we are beyond maturity level 2. This may not be the agreed position by all authors.

In figure 2 we show a simplified layered architectural of an ACS with only one Autonomic Manager (AM) and one Managed Resource (MR) are shown. Collectively, all autonomic managers and their managed components constitute an ACS. To be truly self-adaptive, both managers and managed must exhibit such behaviour. So, a question arises as to where is the system boundary delineating an adaptive system is in this case?

Following on from [2], we consider an illustrative example from aviation, a flight control system. Such a system includes a computer system as a subsystem. It also has a human in the loop. How adaptive or self-adaptive do 'stakeholders' wish that system to be?

NASA recently [3] proposed advanced concepts for the airspace system as a complex, highly integrated system of systems. This necessitates deploying autonomic computing system technologies. It is claimed that the autonomic vehicle concept is similar to the autonomic computing paradigm initiated by IBM to make future computing systems self-managing and self-optimizing, to eliminate the expensive management services needed today. The computing systems considered in that activity consist of large collections of computing engines, storage devices, visualization facilities, operating systems, middleware, and application software.

An autonomic air vehicle [3] can be piloted or uninhabited, and will exhibit a number of advanced characteristics. The vehicle will be self-defining, in that it will have detailed knowledge of its components, current status, internal constraints, ultimate performance, and its relation to other vehicles and to the airspace system. It will be able to reconfigure itself under varying and unpredictable conditions. For example, it will reconfigure wing and airframe geometry to satisfy requirements for a wide range of flight speeds and manoeuvres.

The concept of autonomic vehicles can be extended to hierarchical autonomic transportation systems, with the autonomic vehicle being the first level. The second level is the airspace system—a complex collection of networked subsystems, including facilities, vehicles, and ground support. The third level in the hierarchy is an integrated intermodal system, covering space, air, land, and water transportation. It will function as one seamless whole, maximizing options for convenience, efficiency, and reduced cost. [3, 4]

Flight Control Systems (FCS) software [3,4,5] will incorporate *self-learning* concepts to enable it to discover problems and to reconfigure the system to keep functioning smoothly. The vehicle will collect, analyze, and share information about itself and its local environment with other crafts in the air and with supervisors on the ground to enable a coordinated and optimized airspace system.

Other examples of current systems are taken from IBM's ETTK (Emerging Technologies Toolkit), e.g. Policy Management for Autonomic Computing (PMAC), due to space limitations this is best demonstrated at the workshop.

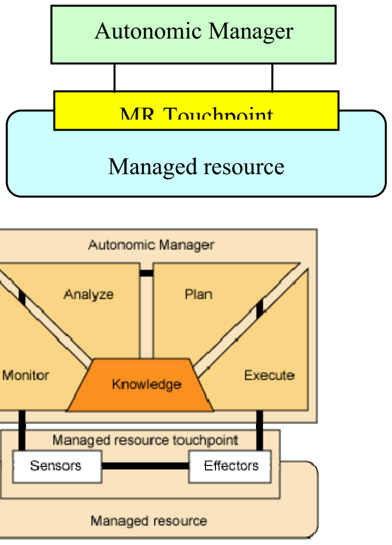

Figure 2: Layered Architecture of ACS

2.2 Anthony finds that the current use of terminology is inconsistent. A very broad spectrum of systems are being described as having self-properties. Such claims can be ambiguous or misleading; terms such as self-configuring (which is part of the autonomics terminology) and self-adaptation are being used to describe widely diverse levels of behavioural sophistication.

There is a need for more-precise classification, based on criteria which include:-

1. The aspect of a system that is adapted:
   - Externally visible behaviour;
   - Internal *configuration*, such as threshold values;
   - Internal *logic* or *semantics*, such as using a meta-policy to dynamically select the most appropriate rule or business policy;
   - Internal *structure* or *architecture*, such as dynamically creating new rules and/or policies.
2. Over what time-frame adaptation is effected:
   - Immediate, having a one-off effect, such as dealing with an unexpected anomaly;
   - Short term, changes remain in force until further changes occur, or the policy instance terminates;
   - Long-term, changes are persisted to future policy instances.
3. Whether adaptation has local or global effects:
   - Local changes affect a single node or agent;
   - Global changes are propagated to other nodes.

Based on these criteria, the author proposes a new 'adaptation taxonomy', comprising four levels of sophistication as follows:

Adaptive: Immediate action effect, as a reaction to environmental or contextual change. Here the externally visible behaviour is modified, but there is no internal change of policy or of stored state that will be used to inform future decisions. Thus the system cannot learn in even the most rudimentary way. The system adapts its instantaneous behaviour, but not itself (i.e. it does not change its configuration or logic).

Self-Configuring: Internal *configuration* is changed. Such as changing a threshold value which subsequently impacts on the way in which rule(s) are evaluated. This type of change affects subsequent decisions. Longer-term adaptation may be achieved if the new configuration is persisted over several instantiations of the policy's container application.

Self-Adapting: In addition to the self-configuring capabilities as described above, internal *logic* or *semantics* are changed permanently, for example changing the priority, and/or execution order of rules. The extent of reorganisation capability is limited by the flexibility of the adaptation mechanisms themselves, and the foresight of the system developer. The adaptation is effectively pre-programmed at a meta-level. For example consider the situation where two rules work together to change both the managed system and their own future behaviour: (within the meta-policy) "*If external event X is sensed at a faster rate than event Y, over a period P, make PolicyZ the active policy and increment the count of PolicyZ substitutions (N_Z)*" and (within PolicyZ) "*If N_Z exceeds a threshold N_T, increase P by 10%*".

Evolvable: New behaviour is 'learnt'. For example a completely new rule or policy is devised, tested and found to be superior to the current setup (at least in the context of the ambient conditions), and is thus incorporated automatically. The inclusion of this category avoids making the self-adaptation category too broad and open-ended. The term 'self-adaptation' does not encompass the full range of possible ways in which a system may change its own behaviour, some desirable behaviours in which systems learn truly new behaviour, are clearly beyond reorganising existing logic.

Several research projects are targeting evolvable systems; see for example [10]. However, current state-of-practice spans the first three categories of the taxonomy. Consider contemporary 'autonomic' systems: a manager sub-component adapts or configures a managed sub-component (which performs the actual business logic) but does not change itself (at the level of its control program). It endlessly runs the same adaptation logic (so *it* is typically *adaptive*). Whether an autonomic component (when viewed externally, i.e. the manager and managed sub-components are treated as a single entity) exhibits *self-adaptive* behaviour depends on the internal sophistication. The autonomics model supports true *self-adaptation*, but many current systems seem to be fundamentally *self-configuring*.

Anthony [11] is concerned with policy-based systems in which policies can modify their *own* behaviour as well as adapting the controlled system. There are several ways in which a policy can change its own behaviour. For example the action carried out as a result of evaluating a rule is permitted to include policy-updating statements that change the way in which the same, or another, rule behaves in the future. This is *self-adaptive* because *semantic* changes can be written into persisted policy files and templates and thus can have long-term and global effects. Significant re-organisation is possible; the system can make changes to its own configuration and internal logic, but not to its own structure or architecture.

A self-adaptive system is restricted to making changes that were designed in (implicitly, at least), and thus is not capable of evolving in a true sense, although the cumulative effect of many individually predictable changes can be sufficiently complex as to be 'surprising'. A key question is: after many self-adaptation iterations, do systems remain *fundamentally* the same or is it possible for completely new behaviour to emerge?

Evolvable systems offer the promise of effectively automatically re-inventing themselves as a result of optimisation to their execution context. This may lead to superior systems, but it also represents significant risk.

2.3 Torsten asserts that Autonomic Computing uses a biological paradigm as a design and control metaphor, the autonomic nervous system. The core CHOP properties of the Autonomic Computing concept are intended to be an electronic realization of the respective mechanisms of the human body. Self-organization can be found in other parts of our natural environment as well, e.g. biological evolution, social group behaviour, market dynamics phenomena and other complex adaptive systems. It is not surprising that projects labelled Autonomic Computing are thus manifold, coming from diverse backgrounds and academic habitats, and aiming at a variety of technological and scientific knowledge increase.

To get a clearer look at the prospects and hurdles, chances and risks of Autonomic Computing, let me divide the complex concept into three spheres: the technological infrastructure, the services infrastructure and the policies infrastructure. The computing technology infrastructure describes the technological progress, the software and hardware modules and the engineering processes to build these. Having the technology in place is a prerequisite for creating new products and services which benefit from self-organizing computing. In their entirety, these new products and services build up the services infrastructure of potential Autonomic computing business.

Business however needs rules, for protecting legitimate rights and properties of the participants. The policies infrastructure describes a joint understanding acceptance of rules, norms and laws as well as agreed-on measures to regulate and enforce compliance.
F.A. von Hayek's claim was that such a coordination mechanism already exists in economic markets, and we may only have to realize that and transfer concepts like money and price signals into the realm of information systems [T1].

Figure 3 [13] shows a framework on how the self-coordination aspect of the Catallaxy leads to a spontaneous order. To achieve such an ordered whole, the software designers of the original system adapt the element's behaviour to changing environments and participants. By way of a cultural evolution, rules of acceptable behaviour get refined and give way to the next version of system inhabitants, who will be released in the information system and shape it to their needs.

The final open question is, whether the spontaneous order provides "acceptable behaviour" of the system – in principle, spontaneous order has no conscience.
It is, however, similarly possible to disrupt the self-coordination through targeted violation of the "regulatory framework". The automated pursuit of the individual goals does not alone produce an acceptable behaviour of the entire system, e.g. in terms of robustness, controllability and the adherence of security criteria for the individual participant.

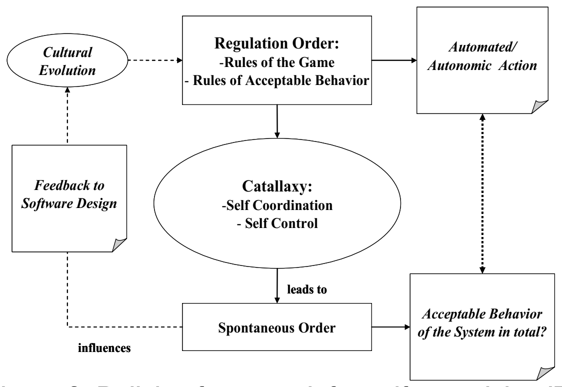

Figure 3: Policies framework for self-organizing IT

The question is raised as to whether these perceptions can be generalized and used for the design of decentralized information systems (or information systems in a decentralized environment) and lead to more efficient paradigms for the implementation of computers. The question remains open as to which is the most effective framework to achieve spontaneous order. The necessary regulation framework required can ex ante only be specified as trial and error. However, we hope that more cost-effective design processes and general conditions befitting the new technology can be found. Self-organization will, in our opinion, then become the main principle for decentralized coordination of multi-component information systems.

2.4 Taleb-Bendiab's view that in nature; self-adaptation is often cited as a principal mechanism for evolution. The differences in architecture between software systems and natural systems can lead to confusion over the terms adaptive and self-adaptive, with respect to the micro and macro level actions of a system and its participants. For example natural organisms are largely dependent on individuality, where only their own self determines their response to environmental signals. This is in contrast to traditional software systems where direct actions of an overseeing administrator and thus obfuscate the exact definition of *self*-adaptive behaviour.

Much established research focuses on the relationship between evolution and self-adaptation. σSA, a self-adaptation model developed by Schwefel & Rudolph [14], in a generalized form, relates individual components to evolvable strategies. To be self-adaptive the system is required to gain knowledge of previously unknown events or results that have an impact on the system at run time. What is required is a method whereby signals are grounded by the system in order to have intrinsic meaning to the system [15].

Signal Grounding: Adaptive vs Self-Adaptive: What's the difference,

For a comparison to be made between adaptive and self-adaptive behaviours, in software systems, a clean distinction is needed. Adaptive behaviour can be thought of as encompassing any action that results in a change to the operation of the problem domain being viewed. This can include external influences that, whilst not directly referencing the domain, adapt its behaviour, and therefore are critical to its continued operation.

Self-adaptive behaviour, however, is limited to actions taken by the software itself, without direct human involvement. Indeed, whilst these actions must take inspiration from human input (in terms of configuration, execution and human adaptation of the domain), the distinction is made because self-adaptive behaviour cannot have been pre-planned or pre-configured: it is emergent, based on perceived signals within the system, in its current configuration. Emergent behaviour, by its very nature, has the property that it cannot be foreseen or pre-planned, and as such, quantifying its construction can be difficult. The signals, emanating from the system, possess an underlying and intrinsic meaning within the context of the functioning system. The difficulty arises in endowing the system with the necessary cognitive facilities to have knowledge of the meaning of these emergent signals, as it is not sufficient to simply prescribe reactions to signals [16]. Again these signals need a ground within the system so that they acquire an intrinsic meaning for the system.

Signal Grounding

In outline the signal grounding problem has three strands: 1) What signals ought to be monitored, 2) When and how to adapt existing thresholds for action for a given signal and, 3) How to determine, completely and automatically within the system, new signals of interest in supplying an autonomic response.

Each of these is a major research topic both in the area of autonomic systems and in the engineering of artificial immune systems in allowing the mapping of sensing to actions and providing anticipatory functions to a system. The first two are equally applicable to adaptive and self-adaptive systems. However the third is a necessary condition for a system to be self-adaptive. The precise formulation of how an autonomic system grounds its own symbols is still dependent on human level systems. However various methods have been proposed for automated cognitive facilities, within the system, to be used to provide a notion of signal grounding. That is the system may be able to use existing grounded signals to infer or deduce novel, previously unrecognisable signals so that the meaning (effect) of these signals to the system becomes known.

Using the grounded signals a logical approach allows the evolution of a dynamic self through deliberation on the results of actions. Reasoning, using deduction, abduction, induction or inference, can then be performed, on the logical representation, to supply receptors for perceived autonomic trigger signals, whether these are for system gain or its protection [Ref]. For example, in the situation calculus [17], the action history represented by:

$$do(a,do(a_1,do(a,s))) \text{ with } SR(a,s) \neq SR(a, do(a_1,do(a,s)))$$

(where $a=sense_f$ for some fluent f, SR(a, s) is the result of performing sensing action a in situation s and a_1 is some deterministic action) can be used to provide a new prediction for the results of action a_1 where the values of other fluents in situation s form the action precondition axioms for a_1 as a context. In this way the signal for f is grounded by occurring in the context of situation s when action a_1 happens. So a signal may be grounded by the system, or a trigger for autonomic response, can be adapted at runtime through a suitable implementation system.

This implementation consists of a representation of the state in which a system must find itself before the transition can occur, and a defined action ontology may then be applied to yield the transition. In essence, an unplanned transition can be thought of as an evolution of a state, triggered by a change in the operating environment, and can be modelled in the same form as a planned transition, using concepts to provide the abstracted action ontology needed for the evolution to occur.

The CA-SPA (Concept-Aided Situated Prediction Action) policy format [18] provides just such behaviour. By providing a situation and a prediction of the required behaviour (or the state to move to), CA-SPA uses introspective functions to determine the actions that need to take place to provide the transition. Modelling the reactions to the system after the application of the action, and sensing whether this moves the system towards the predicted situation, formalises the required ontology. Representing these within the CA-SPA therefore allows the actions required for the transition to complete to be computed, with the direct consequence being that the system moves to the state required.

Signal grounding is still an open question in Information Theory. For a truly self-adaptive system it is necessary that the system has an intrinsic knowledge of the signals known at design time and of those arising at run time. It is for this reason that further research is required to provide means and methods to ascertain signal epistemologies, for autonomic response, throughout the life cycle of the system.

2.5 Gruenwald suggests that **t**he term self-Adaptive has been used quite loosely to mean different things by different computing research communities. The most common definition of this term is that it is the ability of a system to adapt itself to the new environment. With this in mind, we can now examine how it is related to Autonomic Computing Systems. There are eight characteristics that make a computing system Autonomic: self-defining, self-protecting, self-optimizing, self-healing, self-configuring, contextually aware, open, and anticipatory [19, 20]. One can see that a combination of many of these characteristics forms the definition of Self-Adaptive. For example, for a system to be able to adapt itself to a new environment/situation, it must be self-defining so that it can realize its components inside out with respect to the new conditions. Similarly, the system must be self-configuring, self-optimizing and contextually aware. This means that one cannot really separate self-adaptive from autonomic computing systems. With that understanding, the question of "how adaptive are the current autonomic computing systems?" may be rephrased as *"how autonomic are current computing systems?" or "what autonomic feature do the current computing systems have*?"

Ideally, an autonomic computing system should be designed in such a way that the system administrator does not need to interfere with the system's operation in order for it to adjust or react to the new environment. However, current systems have not yet reached this ideal state; most of them are semi-autonomic with only some of the required characteristics implemented. This is partly because it would need to involve research from many different fields, such as data management, distributed computing, and artificial intelligence, which have yet to produce mature results in the autonomic computing arena.

For example, in the data management system area, Microsoft with its auto-admin project has developed auto-indexing techniques that allow its database system, SQL-Server, to tune and recommend indexing structures based on workloads. Similarly, IBM's SMART DB2 system has features providing autonomic index determination and continuous monitoring and alerting the database system administrators about the system's status. In addition, a number of research projects on autonomic optimization using data mining have recently been proposed [22]. However, existing commercial as well as research systems require certain parameters/rules to be predefined and are not easy to use/modify.

This is not to say that not much progress has been made; in fact, the contrary is true. As reviewed in [21] which examines the state of the art research in autonomic computing, good results have been achieved in many areas, including prediction and optimization, knowledge capture and representation, monitoring and root-cause analysis, and policy-based management. In addition, a number of commercial/prototype products have been developed with inclusion of some autonomic features. Besides SMART DB2 and SQL-Server, one can name systems, such as Q-Fabric, OceanStore, and Oceano [23]. In Q-Fabric, self-configuration with continuous online quality management of applications is implemented. In OceanStore, all four issues of self-healing, self-optimization, self-configuration and self-protection are addressed, while in Oceano, only self-optimization and self-defining (or self-awareness) are available. As classified in [23], there are also systems that, even though are not autonomic computing systems themselves, actually provide supporting environments for the development of such systems. Examples include Kinesthetics, eXtreme, Autonomia, and AutoMate.

Although much progress has been made, an observation one can make is that not only that existing systems do not address all eight autonomic characteristics satisfactorily, but also, as pointed out in [21] much of autonomic computing research focuses more on self-optimization than anything else. Even though performance improvement is important, it is only one of the eight elements constituting autonomic computing systems. This leads us to the conclusions that current systems are not really self-adaptive, and research on the remaining seven properties needs be conducted with the same focus and earnest as we have witnessed with that on self-optimization.

Summary

The paper's authors set out to express their views on questions relating to current autonomic computing systems work in general and in particular their adaptive or self-adaptive features. Ibrahim gave examples from NASA's Aeronautics and IBM's ETTK to illustrate his views and position. Anthony proposed a novel taxonomy of adaptation and discussed projects currently exploring evolvable and emergent behavioural systems. Torsten took the view that we need to examine three overarching spheres, viz technological infrastructure, the services infrastructure and the policies infrastructure. He used a system called Catallaxy developed with collaborators to illustrate. Taleb-Bendiab pointed out distinctions between natural and software system when it comes to discussing issues relating adaptive behaviour. He suggests that signal grounding is a means for distinguishing adaptive behaviour. Finally, Gruenwald takes the position that adaptive behaviour pervades many autonomic features and thus the question, in her view, should be how autonomic are current ACS. The author then goes on to explore autonomic features present in some current autonomic computing products and research work e.g. MS SQL Server, Smart DB2, Q-Fabric, OceanStore, and Oceano, Kinesthetics eXtreme, Autonomia, and AutoMate.

Conclusions & Future work

It is clear from perspectives the authors outlined above that there is a long way to go to reach level 5 of autonomic maturity as defined by IBM.

This paper has made some contributions to ACS work. Anthony has proposed a novel taxonomy for adaptive systems and outlined a new higher level in the hierarchy – evolvable systems. Taleb-Bendiab proposed 'adapting' signal grounding as a means for clarity in communicating between kinds of users. The question of how autonomic are current ACS was also addressed with example given from industry and academia. Gruenwald showed that most current autonomic systems focus mainly on self-optimisation than anything else. It is concluded therefore that current systems are not self-adaptive, but using whose definition of that term is a matter for debate.

There is much future work that needs to be done not only in ACS but in many related domains including e.g. to establish many open standards, to use common ontology, metadata and semantics in order to enable communication and understanding between the many parts of a complex autonomic system. Much autonomic and other related research work (computing and otherwise) in many domains and along many dimensions still needs to be done.

References

[1] Marcelo Dascal, Interpretation and Understanding, John Benjamin's Publishing Company, 2003.

[2] Ibrahim, M.T, Telford, R, Dini, P, Lorenz, P, Vidovic, N, Anthony, R, "Self-Adaptability and Man-in-the-Loop: A Dilemma in Autonomic Computing Systems", 15th International Workshops, DEXA'04, 2004, pp. 722-729

[3] Samuel L. Venneri and Ahmed K. Noor "plenty of room in the air", ASME, 2002, http://www.asme.org/ & http://www.memagazine.org/backissues/nov02/features/feat_toc.html

[4] NASA, **"**Designing the 21st Century Aerospace Vehicle" 2003, http://www.nasa.gov/vision/earth/improvingflight/morphing.html

[5] NCARAI, Intelligent Systems "Adaptive & Autonomous-Systems",2003, http://www.nrl.navy.mil/aic/iss/aas/index.php

[6] W3C, "SemanticWeb", 2001, http://www.w3.org/2001/sw/

[7] IEEE P1600.1 , Working Group (SUO WG) , "SUMO,Standard Upper Ontology", 2003, *http://suo.ieee.org/*

[8] W3C, "Metadata and RDF", 2001 http://www.w3.org/Metadata/

[9] IBM, "Open standards driving the development of autonomic technologies", 200? http://www-03.ibm.com/autonomic/open-standards.shtml and http://www.alphaworks.ibm.com/demo/flash/display/pmac0

[10] H. Liu, M. Parashar and S. Hariri, "A Component Based Programming Framework for Autonomic Applications", *ICAC'04*, IEEE, New York, 2004 pp. 10-17.

[11] R. Anthony, "Generic Support for Policy-Based Self-Adaptive Systems", *SAACS '06* (*DEXA*), IEEE, Krakow, Poland, 2006 [this publication]

[12] T. Eymann, M. Reinicke, O. Ardaiz, P. Artigas, F. Freitag, L. Navarro, "Self-Organizing Resource Allocation for Autonomic Networks", DEXA 2003, Prague, Czech Republic. IEEE Computer Society Proceedings, 2003, pp. 656–660.

[13] T. Eymann, S. Sackmann, G. Müller, I. Pippow, "Hayek's Catallaxy - A Forward-looking Concept for Information Systems?" Proc. of America's Conference on Information Systems (AMCIS-2003), Tampa, USA August 2003,. http://aisel.isworld.org/proceedings/amcis/2003/article.asp?Author=4704]

[14] H. Schwefel,, G. Rudolph, "Contemporary Evolution Strategies". Advances in Artificial Life. Third ECAL Proceedings, 1995(1): p. 893-907.

[15] L. Floridi, "Open Problems in the Philosophy of Information", Metap hilosophy vol 35, 2004, pp:554-582,

[16] J.R. Searle, "Mind, Brains and Programs", Behavioral and Brain Sciences, vol.3, 1980, pp:417-457,

[17] H. J. Levesque, F. Pirri and R. Reiter , 'Foundations for the Situation Calculus'. Linköping Electronic Articles in Computer and Information Science, Vol. 3, 1998 #18. http://www.ep.liu.se/ea/cis/1998/018/

[18] P. Miseldine, A. Taleb-Bendiab, "CA-SPA: Balancing the Crosscutting Concerns of Governance Autonomy in Trusted Software", in the Proceeding of the IEEE International Workshop on Trusted and Autonomic Computing Systems within AINA 2006. Vienna, Austria. April 2006.

[19] Paul Horn, "Autonomic Computing: IBM Perspective on the State of Information Technology," 2001, http://www.research.ibm/com/autonomic/

[20] Xiangdong Dong; Hariri, S.; Lizhi Xue; Huoping Chen; Ming Zhang; Pavuluri, S.; Rao, S., "Autonomia: an autonomic computing environment," IEEE International Conference on Performance, Computing, and Communications, April 2003, pp. 61-68.

[21] Sterritt, R., "Autonomic Computing," Innovations Systems Software Engineering, 2005, pp. 79-88.

[22] Sylvain Guinepain and Le Gruenwald, , "Research Issues in Automatic Database Clustering," SIGMOD RECORD, 2005, Vol. 34, No. 1, pp. 33-38.

[23] Manish Parashar and Salim Hariri, "Autonomic Computing: An Overview," Title: Unconventional Programming Paradigms: International Workshop UPP, 2004, pp. 257-269