Integrated Design and Process Technology, **IDPT-2005**
Printed in the United States of America, June, 2005

From Here to Autonomicity: Self-Managing Agents and the Biological Metaphors that Inspire Them

Roy Sterritt and Mike Hinchey

University of Ulster
School of Computing and Mathematics
Jordanstown Campus
Northern Ireland
r.sterritt@ulster.ac.uk

NASA Goddard Space Flight Center
Software Engineering Laboratory
Greenbelt, MD 20771
USA
michael.g.hinchey@nasa.gov

ABSTRACT

We seek inspiration for self-managing systems from (obviously, pre-existing) biological mechanisms. Autonomic Computing (AC), a self-managing systems initiative based on the biological metaphor of the autonomic nervous system, is increasingly gaining momentum as the way forward for integrating and designing reliable systems, while agent technologies have been identified as a key enabler for engineering autonomicity in systems. This paper looks at other biological metaphors such as reflex and healing, heart-beat monitors, pulse monitors and apoptosis for assisting in the realization of autonomicity.

INTRODUCTION

The case has been well made for the need to create self-managing systems, whether due to the complexity problem, the total cost of ownership, or to provide the way forward to enable pervasive and ubiquitous computation and communications [1][2][3].

To enable self-management (*autonomicity*) a system requires many *self* properties (*self-** or *selfware*), such as self-awareness. As such, agent technologies have been identified as a key enabler for engineering autonomicity in systems, both in terms of retrofitting autonomicity into legacy systems and in designing new systems, in part due to their capability to provide self-governance (*autonomy*).

This paper initially looks again at the definitions and properties of agents to set the scene for how they may relate to self-managing initiatives. Autonomic Computing is then introduced in terms of its biological metaphor. We then consider other biological metaphors and how they may contribute to system integrated design and processes, and conclude with some ongoing biologically-inspired applications research.

AGENTS

The definition of what qualifies as an agent and the question of what constitutes its properties is always debatable. A loose definition is 'programs that assist people and act on their behalf by delegating work to them' (end-user perspective) [4]. A systems perspective would highlight that the agent is situated within an execution environment, and have mandatory properties such as being [4]:

- Reactive (adapt to changes in the environment)
- Autonomous (self-governing – control over own actions)
- Goal Driven (proactive)
- Temporally Continuous (continually executing).

Agents may also possess any, or all, of the following orthogonal properties [4]:

- Communicative (with other agents)
- Mobile (travel from one host to another)
- Learning (adapt in accordance with past experience)
- Believable (appear believable to the end user).

Mobility has been highlighted as an orthogonal property; as such, not all agents need be mobile. Stationary agents will communicate with the environment by conventional means such as event messages and remote procedure calls, whereas a mobile agent is not bound to the system where it begins execution, having the unique ability to transport itself from one host to another. Reasons for utilizing mobile agents are typically to [4];

- reduce network load
- overcome network latency
- encapsulate protocols
- execute asynchronously
- execute autonomously
- adapt dynamically
- reflect natural heterogeneity
- maintain robustness and fault-tolerance

Of course mobile agents may also suffer from some of these issues – network latency, for instance – but tend to do so to a lesser extent than relying on a series of event messages and RPC to perform the work between hosts. As such, they may make a contribution to an autonomic environment where it is envisaged substantial communications and work effort will take place between the autonomic managers on different hosts to provide the envisaged system-wide self-managing environment.

BIOLOGICAL SYSTEMS INSPIRATION

Biological systems have inspired system design in many ways – Artificial Intelligence, Artificial Neural Networks, Genetic Algorithms, and Genetic Programming, to name a few. The most recent is inspiration for creating self-managing systems.

Autonomic Nervous System (ANS)

The human body's Autonomic Nervous System is the part of the nervous system that controls the vegetative functions of the body such as circulation of the blood, intestinal activity and secretion and the production of chemical 'messengers', hormones, that circulate in the blood [5]. The system is subdivided into the *sympathetic* (SyNS) and *parasympathetic* (PaNS) nervous systems, the outflow of the autonomic centres is not under direct conscious control. The separate systems tend to have opposite effects: parasympathetic slows the heart rate whereas the sympathetic speeds it up. The sympathetic nervous activity increases in response to fear, i.e., the 'fight or flight' response, while the parasympathetic nervous activity acts to calm, the 'rest and digest' response, for instance [5]. This biological autonomicity is influencing a new paradigm for computing to create similar self-management within systems (Autonomic Computing, Autonomic Communications and Autonomic Systems).

Autonomic Computing

IBM introduced the Autonomic Computing initiative in 2001, with the aim of developing self-managing systems [1],[6]. With the growth of the computer industry, notable examples being highly efficient networking hardware and powerful CPUs, autonomic computing is an evolution to cope with the rapidly growing complexity of integrating, managing, and operating computer-based systems. Computing systems should be effective [**7**], they should serve a useful purpose when they are first launched and continue to be useful as conditions change. The realization of autonomic computing will result in a significant improvement in system management efficiency. The disparate technologies that manage the environment work together to deliver best performance results [8].

As has been mentioned, the Autonomic Computing initiative is inspired by the human body's autonomic nervous system [8]. The autonomic nervous system monitors heartbeat, checks blood sugar levels and keeps the body temperature normal without any conscious effort from the human. There is an important distinction between autonomic activity in the human body and autonomic responses in computer systems. Many of the decisions made by autonomic elements in the body are involuntary, whereas autonomic elements in computer systems make decisions based on tasks chosen to delegate to the technology [8].

Upon launching the Autonomic Computing initiative, IBM defined four key self properties: *self-configuring*, *self-healing*, *self-optimizing* and *self-protecting* [8]. In the few years since, the *self-x* list has grown as research expands, bringing about the general term *selfware* or *self-**, yet these four initial self-managing properties along with the four enabling properties; *self-aware*, *environment aware*, *self-monitor* and *self-adjust*, cover the general goal of self management [9].

To meet this autonomic *selfware* vision, systems should be designed with components being allocated an autonomic manager.

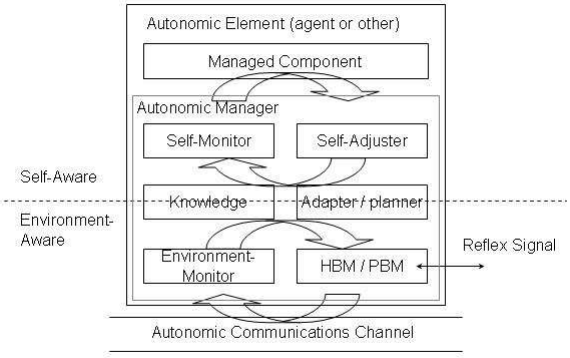

Figure 1 Autonomic Element: Managed Component plus Autonomic Manager

Figure 1 represents an Autonomic Element (AE) which consists of a managed component (MC) and an autonomic manager (AM). Control loops with sensors (self-monitor) and effectors (self-adjuster) together with system knowledge and planning/adapting policies allow the autonomic element to be self-aware and to self-manage. A similar scheme facilitates environment awareness (allowing self-managing if necessary, but without the immediate control to change the environment – this is effected through communication with other autonomic managers that have the relevant influence, through reflex or event messages). In this scheme, every component in a system, and every system within *systems of systems* are self-managing with management communications between autonomic managers.

The influence of the autonomic nervous system (ANS) may imply that the Autonomic Computing initiative is concerned only with low level self-managing capabilities such as reflex reactions. Yet the vision behind the initiative is an overarching goal of system-wide policy-based self-management where a human manager will state a business-critical success factor and the ICT systems will take care of it, self-configuring and self-optimising to meet the policies, and self-protecting and self-healing to ensure the policies are maintained in light of changing circumstances. It may be reasoned that due to our ANS we are freed (non-conscious activity) from the low-level complexity of managing our bodies to perform high-level complex tasks. Similarly, for Computing to develop further and provide equivalent high-level system-wide tasks, necessitates a corresponding low-level 'non-conscious' architecture. As such, increasing this initiative will converge and cross-influence the fields of ubiquitous and pervasive computing.

Autonomic Computing and Agents

Autonomic Computing is dependent on many disciplines for its success; not least of these is research in agent technologies. At this stage, there are no assumptions that agents have to be used in an autonomic architecture, but as in complex systems there are arguments for designing the system with agents [10], as well as providing inbuilt redundancy and greater robustness [11], through to retrofitting legacy systems with autonomic capabilities that may benefit from an agent approach [12].

Emerging research suggests that the autonomic manager may be an agent itself; for instance, an agent termed a self-managing cell (SMC) [13], containing functionality for measurement and event correlation and support for policy-based control.

In Figure 1 the autonomic manager may be considered to be a stationary agent managing a component. The autonomic communications channel implies that AMs communicate through such means as event messages. Yet it is feasible for mobile agents to play a role here. The mobile agent advantages highlighted in the introduction would facilitate autonomic managers within different systems cooperating via agents (Figure 2) as well as self-* event messages.

Reflex and Healing

Reflex and healing is a dual strategy approach concept inspired by biological systems [14],[15]. Animals have a reflex system, where the nerve pathways enable rapid response to pain. Reflexes cause a rapid, involuntary motion, such as when a hot object is touched. The effect is that the system reconfigures itself, moving away from the danger to keep the component functioning.

The body will heal itself on a much longer timescale. Resources from one part of the system are redirected to rebuild the injured body part, including repair of the reflex response network. While this cannot help in the real-time response, directly after an event, it can prepare the system for the next event. In addition, it can readjust the system for operation with a reduced set of resources [14].

An example of the dual approach is being developed for high energy physics experiments, with the use of massive facilities to delve into the basic composition of matter [14]. In this case, the data is so extensive that it is practically impossible to collect all data – decisions must be made in real-time as to whether or not an interesting event has occurred. During its several year lifespan only a small number of novel events are expected. Downtime for the computing system is not an option since this may be when a novel event occurs! Due to the expensive nature of the overall experiment, dual or triple mode redundancy is precluded.

Essentially the design allows for non-critical applications to be overwritten upon fault conditions where the reflex reaction will cause a re-configuration to ensure the matter experiments are still being adequately (less than optimally) monitored. The healing approach then attempts to re-optimize the system with the remaining resources.

Heart Beat Monitoring (HBM)

The typical approach for system management is based on events which are generated and sent under fault or problem conditions.

In the embedded system space, the opposite is typically the case. A system management action occurs when something does not occur. An example is the fault tolerant mechanism of a heartbeat monitor (HBM), through a combination of the hardware (the timer) and software (the heartbeat generator) an 'I am alive' signal is generated periodically to indicate that all is well [16]. The absence of this signal indicates a fault or problem. Some embedded processors have a hardware timer which, if not periodically reset by software, causes a reset/restart. This allows a particularly blunt, though effective, recovery from a software hang.

This approach offers the advantage that, through continuous monitoring, problem determination becomes a proactive rather than a reactive process [15].

An example is in the telecommunications fault management architecture: managers and their vital processes are guarded through the use of heartbeat monitors (HBM); for instance, an SDH (Synchronous Digital Hierarchy) fault manager will send periodic heartbeats up to the cross technology network fault manager. This fault tolerant approach is vital in safeguarding the fault management processes and ensuring the continued operation of the manager. The absence of this 'I am alive' signal enables the remote manager to take protective action such as switching to

the back-up manager while investigating the absence of the beat. This automated failover may be considered a reflex reaction to safeguard the system while the slower healing process is to determine the root cause of the problem in the failed manager [17].

Pulse Monitoring

Essentially, the HBM provides a vital construct. Without it, the system is relying on another process to notice that the process has died, with no guarantee of how much time will have lapsed before this occurs, if it occurs at all.

Yet, vital as it is, essentially the HBM only informs if a process is alive or dead (assuming communications are working) – not the process's actual health or state of well-being. Taking the biological analogy, the rate of the heartbeat indicates the current conditions within which the biological 'system' is operating and determines strategies for 'components' within the system.

The monitoring of the component's health by the autonomic managers, informing the environment through event messaging, and the sub-system monitoring of the managers' health through HBM, are all vital activities carried out independently. Yet, the autonomic managers themselves are in a key position to give an indication of the health of their component or the environment from their perspective as they see it at that moment in time. For instance, if a PC's autonomic manager has noticed a sudden dramatic increase in the traffic emitting from the local area network, this may indicate spam email.

The HBM and event correlation (or event messages to determine what is occurring in the environment) have distinct purposes yet they may be used together to offer new options. Since the autonomic managers are in a position to offer a view of the vital health signs this may be piggybacked into the periodic heartbeat (a health value as a pulse) giving that indication to the other autonomic managers. The analogy is instead of measuring the presence or absence of the heartbeat (dead or alive), opting to measure the actual pulse as a health indicator. This provides the same heartbeat guard (presence or absence of pulse) for the manager along with additional vital signs health information that may be used for reflex reactions.

The logical difference between the pulse signal and general event messages is essentially that the pulse provides the mechanism for a reflex reaction whereas the general event messages under fault conditions form part of the slower healing process–root cause analysis from the event stream [15],[2],[17].

Apoptosis

The biological analogy of autonomic systems has been discussed earlier. While reading this the reader is not consciously concerned with their breathing rate or how fast their heart is beating. Another typical biological example is that the touching of a sharp knife results in a reflex reaction to reconfigure the area in danger to a state that is no longer in danger (self-protection, self-configuration, and, if damage has occurred, self-healing) [7].

If you cut yourself and start bleeding, you will treat it and carry on with your tasks without any further conscious thought. Yet, often, the cut will have caused skin cells to be displaced down into muscle tissue [18]. If they survive and divide, they have the potential to grow into a tumor. The body's solution to dealing with this situation is cell self-destruction. There is mounting evidence that cancer is the result of cells not dying fast enough, rather than multiplying out of control, as previously thought.

It is believed that a cell knows when to commit suicide because cells are programmed to do so – self-destruct (sD) is an intrinsic property. This self-destruction is delayed due to the continuous receipt of biochemical reprieves. This process is referred to as apoptosis [19], meaning 'drop out', and was used by the Greeks to refer to the Autumn dropping of leaves from trees; i.e., loss of cells that ought to die in the midst of the living structure. The process has also been nicknamed 'death by default' [20], where cells are prevented from putting an end to themselves due to constant receipt of biochemical 'stay alive' signals.

Further investigations into the apoptosis process [21] have discovered more details about the self-destruct predisposition. Whenever a cell divides, it simultaneously receives orders to kill itself. Without a reprieve signal, the cell does indeed self-destruct. It is believed that the reason for this is self-protection, as the most dangerous time for the body is when a cell divides, since if just one of the billions of cells locks into division the result is a tumor, while simultaneously a cell must divide in order to build and maintain a body.

The suicide and reprieve controls have been compared to the dual-key on a nuclear missile [18]. The key (chemical signal) turns on cell growth but at the same time switches on a sequence that leads to self-destruction. The second key overrides the self-destruct [18].

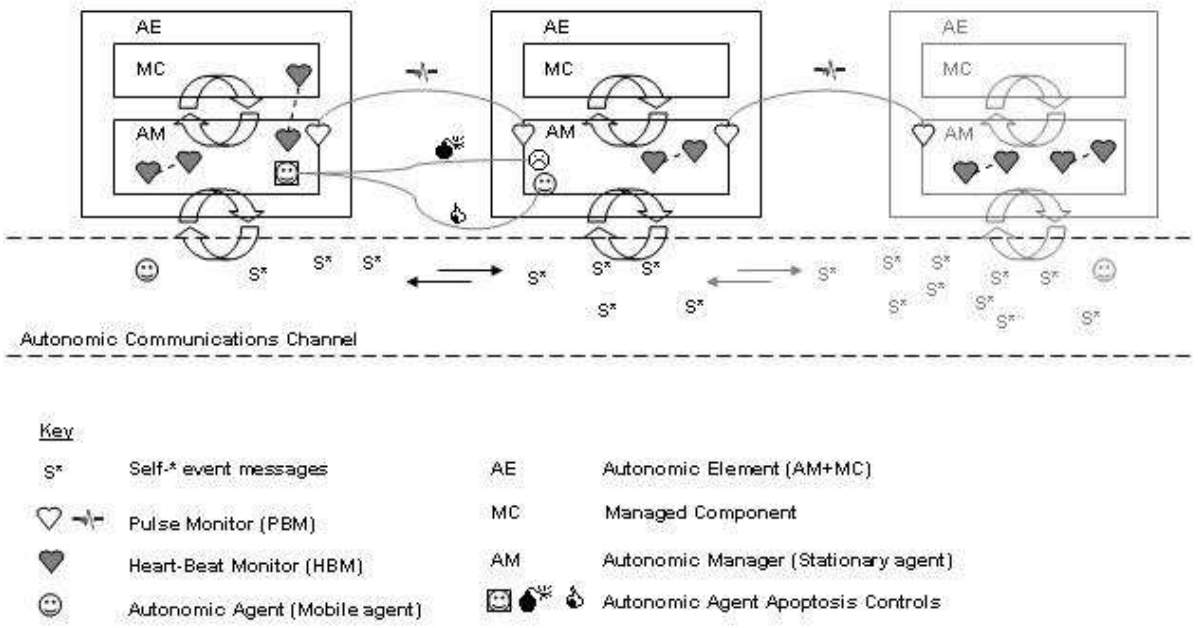

Figure 2 An Autonomic Environment consisting of Autonomic Elements with Autonomic Agents (stationary & mobile), HBMs, PBMs and Apoptosis.

Apoptosis within Autonomic Agents

Agent destruction has been proposed for mobile agents, in order to facilitate security measures [22]. Greenberg et al. highlighted the situation simply by recalling the situation where the server omega.univ.edu was decommissioned, its work moving to other machines. When a few years later a new computer was assigned the old name, to the surprise of everyone, email arrived, much of it 3 years old [23]. The mail had survived 'pending' on Internet relays waiting for omega.univ.edu to come back up.

Greenberg encourages consideration of the same situation for mobile agents; these would not be rogue mobile agents – they would be carrying proper authenticated credentials. This work would be done totally out-of-context due to neither abnormal procedure nor system failure. In this circumstance, the mobile agent could cause substantial damage, e.g., deliver an archaic upgrade to part of the network operating system, resulting in bringing down the entire network.

Misuse involving mobile agents comes in the form of: misuse of hosts by agents, misuse of agents by hosts, and misuse of agents by other agents.

From an agent perspective, the first is through accidental or unintentional situations caused by that agent (race conditions and unexpected emergent behavior), the latter two through deliberate or accidental situations caused by external bodies acting upon the agent. The range of these situations and attacks have been categorized as: damage, denial-of-service, breach-of-privacy, harassment, social engineering, event-triggered attacks, and compound attacks.

In the situation where portions of an agent's binary image (e.g., monetary certificates, keys, information, etc.) are vulnerable to being copied when visiting a host, this can be prevented by encryption. Yet there has to be decryption in order to execute, which provides a window of vulnerability [23]. This situation has similar overtones to our previous discussion on biological apoptosis, where the body is at its most vulnerable during cell division [24].

BIOLOGICAL INSPIRED APPLICATIONS

The pulse monitor has been recommended as an extension of the Globus Heartbeat Monitor (HBM) for Grid computing [15], as a construct within an autonomic manager [2],[7] and a reflex mechanism within a telecommunications fault management architecture [17]. It has been prototyped in a PC environment to construct a self-healing tool through utilising the pulse monitor together with a health check (vital signs) mechanism operating in a peer-to-peer (P2P) mode without any additional environment on top of the Windows OS [25] and proposed as a mechanism to share environment awareness responsibilities akin to a neighbourhood watch scheme [26].

Figure 2 represents a high level view of an autonomic environment with the previously discussed biologically influenced constructs and mechanisms highlighted (much of the detail is extracted for simplicity). Each of the AEs (*Autonomic elements*) is an abstract view of Figure 1 and in this scenario the *managed component* (MC) represents a self-managing computer system. Note that each of these autonomic elements (self-managing computer system) may have

many other lower level AEs (for instance an autonomic manager for the disk drive) while at the same time the AEs in Figure 2 may reside within the scope of a higher level AM (system-wide local area network domain's AE).

Within an AM, vital processes may be safe guarded by heart-beat monitors to ensure their continued operation, and to have an immediate indication if any fail (note in Figure 2 that all three AEs have HBMs within their AMs). An AM has a control loop continually monitoring (and adjusting if necessary) metrics within the MC, yet vital processes within the MC may also be safe-guarded by a HBM with it emitting a heart-beat as opposed to being polled by the AM to ensure its continued operation, resulting in avoiding lost time (time to next poll) for the AM to notice it has failed when it does fail (note in Figure 2 that the left-hand AE has a HBM between the AM and a process on the MC).

Since each AM is in a key position to be aware of the health of the computer system (through the continuous control loop with the MC) it may share this health indication through a pulse to another AM (for instance, in Figure 2 the left-hand AE to the middle AE) this not only allows self-managing options if the machines are, for instance, sharing workload as a cluster but protects the AM itself as the pulse also acts as a HBM from one AM to another. As such, if the vital process of the AM itself fails, the neighbouring AM will immediately become aware and for instance pursue a restart of the failed AM. This pulse signal may also act as a reflex signal (more direct than the AM processing lots of event messages to eventually determine an urgent situation) between AMs warning of an immediate incident.

It has been highlighted that the AMs communicate and cooperate through self-managing event messages and pulse signals. Mobile agents may also be used to assist in the self-managing tasks where one AM dispatches an agent to work on its behalf, for example to update a set of policies. The apoptosis (self-destruct mechanism) may be utilized in this scenario as self-protection, to withdraw authorization to continue operation, for example, if the policies become out-of-date (Figure 2 left-hand AE to the middle AE depicts both scenarios, one agent is to self-destruct the other still has an authorization signal to continue).

Space Exploration Missions, through necessity, have been incorporating more and more Autonomy. Autonomy may be considered as self-governance of one's own tasks/goals. In terms of ANTS (Autonomous Nano-Technology Swarm) missions this, for instance, results in a worker having responsibility for its goals. To achieve these goals many self-* properties such as self-configuration will be necessary, as well as utilization of HBM, PBM and reflex reactions within AMs.

NASA missions, such as ANTS, have Mission control and operations in a trusted private environment. This eliminates many of the wide range of agent security issues discussed earlier, just leaving the particular concerns; namely, is the agent operating in the correct context and exhibiting emergent behavior within acceptable parameters, whereupon apoptosis can make a contribution.

The ANTS architecture is itself inspired by biological low level social insect colonies with their success in the division of labor. Within their specialties, individual specialists generally outperform generalists, and with sufficiently efficient social interaction and coordination, the group of specialists generally outperforms the group of generalists. Thus systems designed as ANTS are built from potentially very large numbers of highly autonomous, yet socially interactive, elements. The architecture is self-similar in that elements and sub-elements of the system may also be recursively structured as ANTS [27].

Targets for ANTS-like missions include surveys of extreme environments on the Earth, Moon, or Mars, as well as asteroid, comet, or dust populations. The revolutionary ANTS paradigm makes the achievement of such goals possible through the use of many small, autonomous, reconfigurable, redundant element craft acting as independent or collective agents [28].

Let us consider the role of the self-destruct property, inspired by apoptosis, in the ANTS mission: suppose one of the worker agents was indicating incorrect operation, or when co-existing with other workers was the cause of undesirable emergent behavior, and was failing to self-heal correctly. That emergent behavior (depending on what it was) may put the scientific mission in danger. Ultimately the stay-alive signal from the ruler agent would be withdrawn [24].

If a worker, or its instrument, were damaged, either by collision with another worker, or (more likely) with an asteroid, or during a solar storm, a ruler could withdraw the stay-alive signal and request a replacement worker. Another worker could self-configure to take on the role of the lost worker; i.e., the ANTS adapt to ensure an optimal and balanced coverage of tasks to meet the scientific goals.

If a ruler or messenger were similarly damaged, its stay-alive signal would also be withdrawn, and a worker would be promoted to play its role.

All of the spacecraft are powered by batteries that are recharged by the sun using solar sails [29],[30]. Although battery technology has greatly advanced, there is still a 'memory loss' situation, whereby batteries that are continuously recharged eventually lose some of their power and cannot be recharged to full power. After several months of continual operation, each of the ANTS will no longer be able to recharge sufficiently, at which point their stay-alive signals will be withdrawn, and new craft will need to be assembled or launched from Earth.

CONCLUSIONS

Achieving the development of computer-based systems that can self-manage without the conscious effort of the user is the overarching vision of the Autonomic Computing initiative achieved through the utilization of self-* properties [3]. Many may consider that there are worlds between self-managing biological systems, such as the autonomic nervous system, and our corresponding software today, yet that is the point – this is a long term strategic vision seeking influences from biology. Secondly, it is a metaphor not an attempt to mimic biological systems. This vision may also be viewed as the evolution of the aim of creating robust dependable systems [9].

To facilitate this aim, additional biologically-inspired mechanisms are proposed to assist with autonomicity, and to be included within the autonomic element [15],[2]. For stationary agents we established the concepts of the HBM and PBM: Heart-Beat Monitor (I am alive) a fault-tolerant mechanism which may be used to safeguard the autonomic manager to ensure that it is still functioning by periodically sending 'I am alive' signals. Pulse Monitor (I am healthy) extends the HBM to incorporate reflex/urgency/health indicators from the autonomic manager, representing its view of the current self-management state. The analogy is with measuring the pulse rate instead of merely detecting its existence. For mobile agents, Apoptosis (Stay alive) is a proposed additional construct used to safeguard the system and agent; a signal indicates that the agent is still operating within the correct context and behavior, and should not self-destruct, providing a self-protection mechanism [31].

We have concluded with a brief description of ongoing work utilizing these biologically-inspired concepts to assist in achieving autonomicity.

ACKNOWLEDGEMENTS

The development of this paper was supported at University of Ulster by the Computer Science Research Institute (CSRI) and the Centre for Software Process Technologies (CSPT), funded by Invest NI through the Centres of Excellence Programme, under the EU Peace II initiative.

Part of this work has been supported by the NASA Office of Systems and Mission Assurance (OSMA) through its Software Assurance Research Program (SARP) project, Formal Approaches to Swarm Technologies (FAST), and by NASA Goddard Space Flight Center, Software Engineering Laboratory (Code 581).

REFERENCES

1. P. Horn, "Autonomic computing: IBM perspective on the state of information technology," IBM T.J. Watson Labs, NY, 15th October 2001. Presented at AGENDA 2001, Scottsdale, AR (available at http://www.research.ibm.com/autonomic/), 2001.
2. R. Sterritt, "Towards Autonomic Computing: Effective Event Management", Proceedings of the 27th Annual IEEE/NASA Software Engineering Workshop, Greenbelt, MD, Dec. 2002.
3. J. O. Kephart and D. M. Chess. The Vision of Autonomic Computing, Computer, 36(1):41–52, 2003.
4. D.B. Lance, M. Oshima, "Programming and deploying Java Mobile Agents with Aglets", Addison-Wesley, 1998.
5. Oxford Reference Encyclopedia, Oxford University Press, 1998.
6. A. G. Ganek, T.A. Corbi, "The dawning of the autonomic computing era", IBM Systems Journal, Vol. 42, No. 1, 2003
7. R. Sterritt, D.W. Bustard, "Towards an Autonomic Computing Environment", In Proceedings of IEEE DEXA 2003 Workshops - 1st International Workshop on Autonomic Computing Systems, Prague, Czech Republic, September 1-5, 2003, IEEE Computer Society Press, pp 694-698.
8. IBM White Paper, "An architectural blueprint for autonomic computing", IBM, April 2003.
9. R. Sterritt, D. Bustard, "Autonomic Computing-a Means of Achieving Dependability?", Proc. IEEE Int. Conf. on the Engineering of Computer Based System (ECBS'03), Huntsville, Alabama, USA, April 7-11 2003
10. N.R. Jennings, M. Wooldridge, Agent-oriented Software Engineering, In J. Bradshaw (ed.), Handbook of Agent Technology, AAAI/MIT Press, Cambridge, 2000.
11. M.N. Huhns, V.T. Holderfield, R.L.Z. Gutierrez, Robust Software via Agent-Based Redundancy, In Proceedings Second International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2003, July 14-18, 2003, Melbourne, Victoria, Australia, pp 1018-1019.
12. G. Kaiser, J. Parekh, P. Gross, G. Valetto, Kinesthetics eXtreme: An External Infrastructure for Monitoring Distributed Legacy Systems, In Proceedings Autonomic Computing Workshop – IEEE Fifth Annual International Active Middleware Workshop, Seattle, USA, June 2003.
13. E. Lupu, et al., EPSRC AMUSE: Autonomic Management of Ubiquitous Systems for e-Health, 2003.
14. T. Bapty, S. Neema, S. Nordstorm, S. Shetty, D. Vashishtha, J. Overdorf, P. Sheldon, "Modeling and Generation Tools for Large-Scale, Real-Time Embedded Systems", 10th

IEEE International Conference and Workshop on the Engineering of Computer-Based Systems, Huntsville, Alabama, USA, 7-10th April 2003, pp 11-16.
15. R. Sterritt, "Pulse Monitoring: Extending the Health-check for the Autonomic GRID", IEEE Workshop on Autonomic Computing Principles and Architectures (AUCOPA' 2003) in Proc. IEEE Int. Conf. Industrial Informatics (INDIN 2003), Banff, Alberta, Canada, 22-23 August 2003.
16. P. Stelling, I. Foster, C. Kesselman, C. Lee, G. v. Laszewski, "A Fault Detection Service for Wide Area Distributed Computations", Proceedings of the 7th IEEE Symposium on High Performance Distributed Computing, 1998.
17. R. Sterritt, D. Gunning, A. Meban, P Henning, "Exploring Autonomic Options in an Unified Fault Management Architecture through Reflex Reactions via Pulse Monitoring", IEEE Workshop on the Engineering of Autonomic Systems (EASe 2004) in Proc. 11th Ann. IEEE Int. Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2004), Brno, Czech Republic, 24-27 May 2004.
18. J. Newell, Dying to Live: Why our Cells Self-Destruct, Focus, December 1994.
19. R. Lockshin, Z. Zakeri, Programmed Cell Death and Apoptosis: Origins of the Theory, Nature Reviews Molecular Cell Biology, 2:542-550, 2001.
20. Y. Ishizaki, L. Cheng, A.W. Mudge, M.C. Raff, Programmed Cell Death by Default in Embryonic Cells, Fibroblasts, and Cancer Cells, Mol. Biol. Cell, 6(11):1443-1458, 1995.
21. J. Klefstrom, E.W. Verschuren, G.I. Evan, c-Myc Augments the Apoptotic Activity of Cytosolic Death Receptor Signaling Proteins by Engaging the Mitochondrial Apoptotic Pathway, J. Biol Chem,. 277:43224-43232, 2002.
22. J.D. Hartline, Mobile Agents: A Survey of Fault Tolerance and Security, University of Washington, 1998.
23. M.S. Greenberg, J.C. Byington, T. Holding, D.G. Harper, Mobile Agents and Security, IEEE Communications, July 1998.
24. R. Sterritt, M. G. Hinchey, "Apoptosis and Self-Destruct: A Contribution to Autonomic Agents?" In Proceedings FAABS-III, 3rd NASA/IEEE Workshop on Formal Approaches to Agent-Based Systems (April 2004), Greenbelt, MD, Springer Verlag LNCS 3228, 2005.
25. R. Sterritt, S. Chung, "Personal Autonomic Computing Self-Healing Tool", Proceedings of IEEE Workshop on the Engineering of Autonomic Systems (EASe 2004) at 11th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2004), Brno, Czech Republic, 24-27 May, pp 513-520.
26. R. Sterritt, D.F. Bantz, "PAC-MEN: Personal Autonomic Computing Monitoring Environments", In Proceedings of IEEE DEXA 2004 Workshops - 2nd International Workshop on Self-Adaptive and Autonomic Computing Systems (SAACS '04), Zaragoza, Spain, August 30 – 3 September, 2003.
27. S. A., J. Mica, J. Nuth, G. Marr, M. Rilee, M. Bhat, ANTS (Autonomous Nano-Technology Swarm): An Artificial Intelligence Approach to Asteroid Belt Resource Exploration, Curtis, International Astronautical Federation, 51st Congress, October 2000.
28. P.E. Clark, S. Curtis, M. Rilee, W. Truszkowski, J. Iyengar, H. Crawford, "ANTS: A New Concept for Very Remote Exploration with Intelligent Software Agents", Presented at 2001 Spring Meeting of the American Geophysical Union, San Francisco, 10-14 December 2001; EOS Trans. AGU, 82 (47), 2001.
29. W. Truszkowski, J. Rash, C. Rouff and M. Hinchey, Asteroid Exploration with Autonomic Systems, In Proceedings of IEEE Workshop on the Engineering of Autonomic Systems (EASe 2004) at the 11th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2004), Brno, Czech Republic, 24-27 May 2004, pp 484-490.
30. W. Truszkowski, M. Hinchey, J. Rash and C. Rouff, NASA's Swarm Missions: The Challenge of Building Autonomous Software, IEEE IT Professional, September/October 2004, pp 51-56.
31. R. Sterritt, M.G. Hinchey, "Engineering Ultimate Self-Protection in Autonomic Agents for Space Exploration Missions", Proceedings of IEEE Workshop on the Engineering of Autonomic Systems (EASe 2005) at 12th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2005), Greenbelt, MD, USA, 3-8 April, 2005, pp 506-511.