AUTONOMIC ROBOTICS FOR FUTURE SPACE MISSIONS

Roy Sterritt, George Wilkie, Guanitta Brady, Catherine Saunders, Martin Doran

Ulster University, Jordanstown Campus, Co. Antrim, Northern Ireland
Email:{r.sterritt, fg.wilkie, gm.brady}@ulster.ac.uk | { saunders-c, Doran-M18}@ email.ulster.ac.uk

ABSTRACT

This paper introduces at a high level the concept of *Autonomic Robotics* based on the Autonomic Computing paradigm with the aim to achieve a systematic means to obtaining self-managing and autonomous robotic software for future space missions. Three specific streams of research within this area are summarised; cooperation between the autonomic robots, an autonomic robotic architecture and the development of middleware for easier and agile software development for such systems.

1. INTRODUCTION

IBM's Paul Horn likened the needs of large scale systems management to that of the human Autonomic Nervous System (ANS). The ANS, through self regulation, is able to effectively monitor, control and regulate the human body without the need for conscious thought. This self-regulation and separation of concerns provides human beings with the ability to concentrate on high level objectives without having to micro-manage the specific details involved. The vision and metaphor of Autonomic Computing is to apply the same principles of self-regulation and complexity-hiding to the design of computer-based systems, in the hope that eventually computer systems can achieve the same level of self-regulation as the human ANS. This vision of creating self-managing and self-directing systems has become mainstream in both the academic and industrial research community under the Autonomic and Autonomous Systems (and related) initiatives [1]-[8].

This paper considers the research to apply the Autonomic Computing paradigm to the self-management of robotics; "Autonomic Robotics" specifically for Space Missions. It briefly starts with reviewing the AC paradigm and how this fits with Robotics. The paper then reports on three ongoing streams of the authors' research, innovation and development; 1) Autonomic Inter-Cooperation, 2) Autonomic Intra-Cooperation and 3) "SPAAACE-Ware" (*S*elf- *P*roperties for *A*utonomic, *A*poptotic and *A*utonomous *C*omputer-*B*ased *E*nvironments – soft*Ware*) middle-ware to engineer Autonomicity into space missions.

2. AUTONOMIC COMPUTING

In 2001, IBM launched the Autonomic Computing initiative, its focus being the development of self-managing systems, of which self-managing software is a major component, inspired by the human body's Autonomic Nervous System, or ANS [6][7].

The ANS is that part of the nervous system that controls the vegetative functions of the body, such as circulation of the blood, intestinal activity, and secretion and production of chemical "messengers" (hormones) that circulate in the blood. The sympathetic nervous system (SyNS) supports "fight or flight", providing various protection mechanisms to ensure the safety and wellbeing of the body. The parasympathetic nervous system (PaNS) supports "rest and digest", ensuring that the body performs necessary functions for long term health.

The general properties of an autonomic (self-managing) system can be summarised by four objectives: being self-configuring, self-healing, self-optimizing and self-protecting, and four attributes: self-awareness, self-situated, self-monitoring and self-adjusting. Essentially, the objectives represent broad system requirements, while the attributes identify basic implementation mechanisms [4].

Self-configuring represents a system's ability to re-adjust itself automatically; this may simply be in support of changing circumstances, or to assist in self-healing, self-optimization or self-protection.

Self-healing, in reactive mode, is a mechanism concerned with ensuring effective recovery when a fault occurs, identifying the fault, and then, where possible, repairing it. In proactive mode, it monitors vital signs in an attempt to predict and avoid "health" problems (reaching undesirable situations).

Self-optimization means that a system is aware of its ideal performance, can measure its current performance against that ideal, and has defined policies for attempting improvements. It may also react to policy changes within the system as indicated by the users. A self-protecting system will defend itself from accidental or malicious external attack. This necessitates awareness of potential threats and a means of handling those threats.

In achieving such self-managing objectives, a system must be aware of its internal state (self-aware) and

current external operating conditions (self-situated). Changing circumstances are detected through self-monitoring and adaptations are made accordingly (self-adjusting). As such, a system must have knowledge of its available resources, its components, their desired performance characteristics, their current status, and the status of inter-connections with other systems, along with rules and policies of how these may be adjusted. Such ability to operate in a heterogeneous environment will require the use of open standards to enable global understanding and communication with other systems.
These mechanisms are not independent entities. For instance, if an attack is successful, this will include self-healing actions, and a mix of self-configuration and self-optimisation, in the first instance to ensure dependability and continued operation of the system, and later to increase the self-protection against similar future attacks. Finally, these self-mechanisms should ensure there is minimal disruption to users, avoiding significant delays in processing [4].

2.1. Autonomic Element

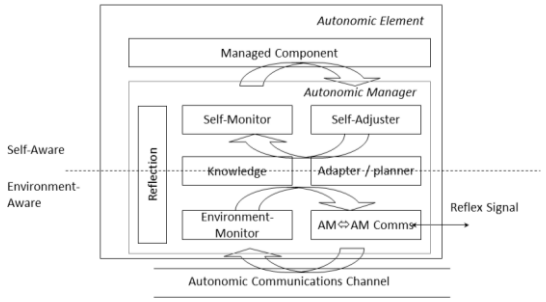

Figure 1. Autonomic Element

At the heart of the architecture of any autonomic system are sensors and effectors. A control loop is created by monitoring behaviour through sensors, comparing this with expectations (knowledge, as in historical and current data, rules and beliefs), planning what action is necessary (if any), and then executing that action through effectors. The closed loop of feedback control provides the basic backbone structure for each system component. Figure 1 highlights that there are two conceptual control loops in an Autonomic Element – one for self-awareness (around the Managed Component) and another for self-situation (environmental awareness, situation and context-awareness) [5].
IBM represents this self-monitor/self-adjuster control loop as the monitor, analyze, plan and execute (MAPE) control loop. The monitor-and-analyze parts of the structure process information from the sensors to provide both self-awareness and an awareness of the external environment. The plan-and-execute parts decide on the necessary self-management behavior that will be executed through the effectors. The MAPE components use the correlations, rules, beliefs, expectations, histories, and other information known to the autonomic element, or available to it through the knowledge repository within the AM [7] (this is also referred to as MAPE-K loop).

2.2. Autonomic Environment

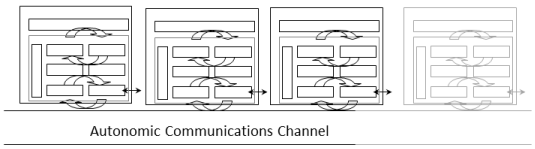

Figure 2. Autonomic Environment

The autonomic environment requires that autonomic elements and, in particular, autonomic managers communicate and cooperate with one another concerning self-* activities, in order to ensure the robustness and self-management of the total system (system of systems) as depicted in Fig. 2.
Some interpretations of the original AC paradigm's vision is that it should be a peer-to-peer approach, yet many of the solutions still operate on an client-server/slave-master basis. The NASA ANTS concept mission goes to the other end of the spectrum using the SWARMS paradigm [23][20]. The ideal form of cooperation is one of the topics of our research and discussed later.

2.3. Autonomic Robotics

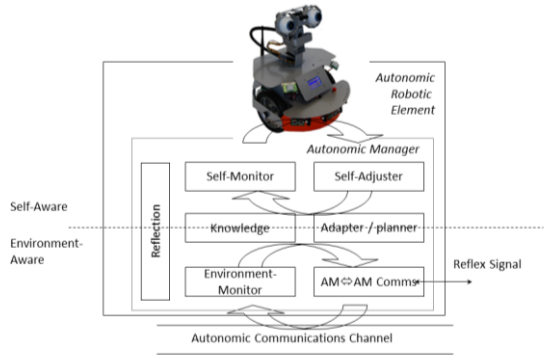

Figure 3. Autonomic Robotic Element

Upon consideration of Fig. 1 it may be noted that an Autonomic Manager is monitoring/adjusting an MC (Managed Component). In the majority of AC (Autonomic Computing) cases these are computer systems. We have hypothesised in the past that this should be extended to all computer-based systems, and that all CBS should be autonomic [21]. In this case the MC is the robot (Fig. 3). We are researching the

potential for the AC paradigm to offer a systematic approach to autonomy through its specific form of self-managing collaboration (Fig. 4).

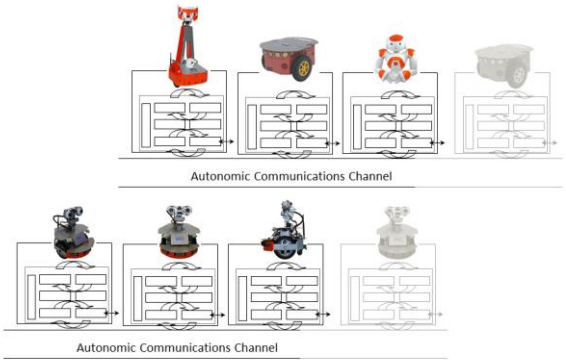

Figure 4. "Autonomic Robotics" Environment

3. AUTONOMIC ELEMENT INTER-COOPERATION (EXTERNAL)

The Autonomic Computing (self-managing systems) paradigm accepts that components and elements will fail but intends that the system does not by providing self-healing, self-configuring and reconfiguring, self-protecting and self-optimising strategies through the cooperation of the elements within the system.

Cooperation between elements seems straight forward when you consider the logical Autonomic architecture in Fig. 2. An AE (Autonomic Element) consists of an Autonomic Manager (AM) and the Managed Component (MC), where in this case the MC is the robots and their supporting architecture with each having its own AM and thus referred to as an ARE (Autonomic Robotic Element). Yet this logical view hides substantial complexity.

When one considers the spectrum of cooperation, they range from client-server/master-slave, through peer-to-peer, to the extreme of SWARM computing modes of collaboration or cooperation between elements.

Grid Computing systems tend to follow a client-server approach but recommendation have been made to become more proactive in recognising faults by incorporating more of a peer-to-peer and Autonomic approach [22]. We have worked with NASA Goddard Space Flight Center (GSFC) on SWARM based approaches for future concept missions e.g. ANTS [1][20][23] yet once the mission configures into a sub-swarm to do the actual exploratory science, it very much resembles a client-server/master-slave or P2P approach.

We are investigating the best scenarios and scalability ranges for each of these cooperation paradigms with Robots [9]. Autonomic Computing also has self-adaptability and self-organisation as part of its mandate, as such we are also investigating the potential that the nature of the system can change depending on the situation facing it. For instance, taking the NASA ANTS PAM concept mission as an example, the 2.5 year flight of 1000s of robotic craft to the asteroid belt may best be achieved in a Swarm operational mode; upon reaching the belt switching to peer-to-peer mode while surveying for potential asteroids of interest and then upon scientific study of specific asteroids self-configuring into a client-server/master-slave mode with sub-swarms or clusters having a ruler directing the operation and communicating with other sub-swarm leaders, to for instance, share rare resources due to damaged craft.

In our research we are experimenting with actual robots but also building a swarm simulator to obtain large scale scenarios (Fig 5).

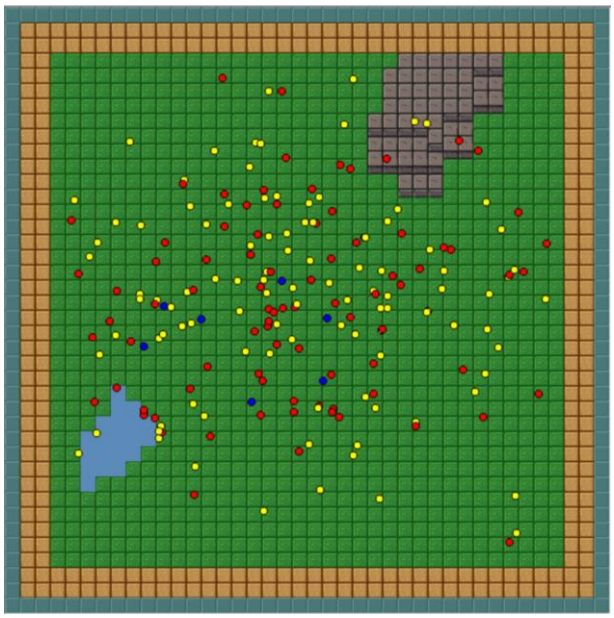

Figure 5. Autonomic Robot Swarm Simulator

Figure 6. X80-H and X80 communicating under comms fault conditions [10]..

As has been highlighted, the Autonomic Computing paradigm aims to ensure that the *system of systems* or

mission continues to operate even under fault conditions by self-adapting (and accepting this will be at less than optimal performance). As such the Autonomic System requires contingency strategies at its disposal.

One example scenario we explored was total failure of communications between robots and attempting to communicate by semaphores through flashing lights and then extending this to robots reading messages from each other's display screens (Fig 6) [10].

4. AUTONOMIC ELEMENT INTRA-COOPERATION (INTERNAL)

In pursuit of future space exploration, researchers have described the concept of moving from missions based on single rovers towards multiple rovers and indeed in the ANTS scenario potentially 1000s of flying rovers; this concept is based on the fact that multiple rovers are capable of completing more tasks and covering a larger area than a single rover. However, the amount of expenditure put into each rover is likely to be greatly reduced compared to that of a single mission rover. At the extreme Swarm end of the spectrum the crafts will be expendable. These lower spec. rovers could be more vulnerable to hardware faults. However, if the software system built into each rover is based on autonomic principles, then the ability of the rover to continue to operate would be greatly increased, as well a redundancy provided through the Autonomic Cooperation.

These approaches require the that ARE is evaluating its own health (Fig.1 self-awareness control loop) but at more than one dimension. For instance, self-configuring (sC), self-healing (sH) and self-optimising (sO) in reaction to an immediate danger is very different to sC, sH and sO in relation to a reflection process, that being from analysing data patterns over several weeks and identifying a better operational mode.

Note we have already incorporated the Reflex Reaction (via Pulse Monitoring [12]) and Reflection into our AE (Fig. 1). But this also clearly operates at a system level and not just within an AE.

This has led us to research into a suitable architecture both internal (intra-cooperation within the AE) and intra-cooperation between layers or vertical orchestration within a system.

One approach we had considered in the past was VSM or Beer's model.

The Viable System Model [14]-[16] provides a theoretically supported cybernetic model of organization. Viable systems may be defined as being robust against internal malfunction and external disturbances and have the ability to continually respond and adapt to unexpected stimuli allowing them to survive in a changing and unpredictable environment. The model specifically attempts to imbue the system with the ability to adapt to circumstances not foreseen by the original designer and identifies the necessary and sufficient communication and control systems that must exist for any organization to remain viable in a changing environment. In doing so, the model does not attempt to specify nor prescribe the activities that must occur in each system, instead activities are described or typified by a cybernetic rationale to allow either the design of activities to match the cybernetic criteria or for actual activities to be identified by their system type and hence assigned to the appropriate element of the model. Such a generalized approach allows the model to be applied to any organization regardless of size (Fig. 7).

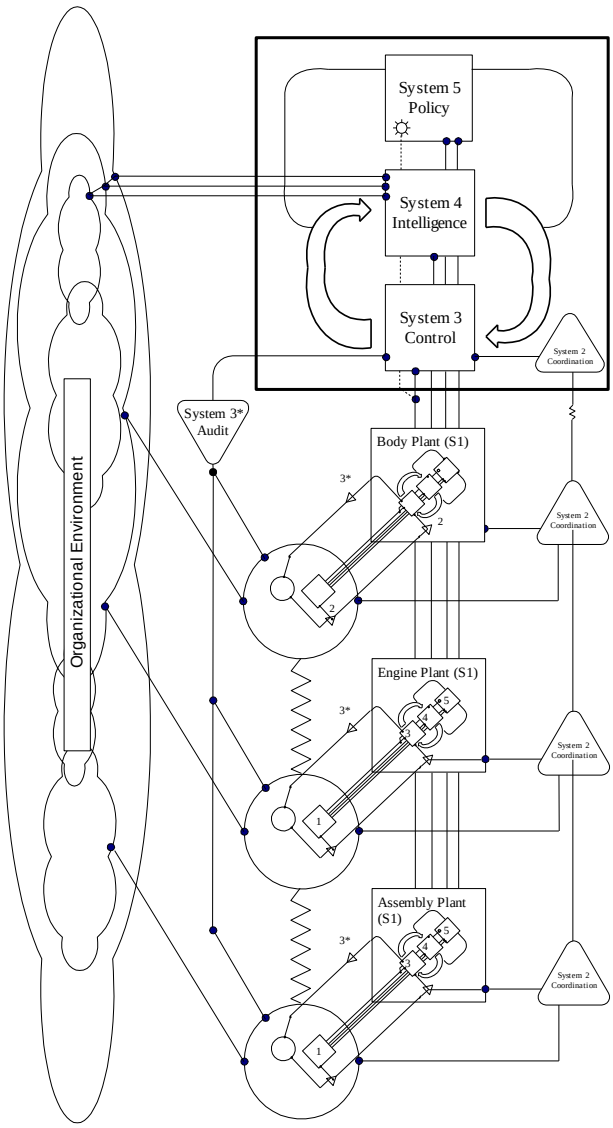

Figure 7. The Viable System Model [14]-[16]

The Systems shown in Fig. 7 concern the management structure at one level of the systems and consequently specify the communication and control structures that must exist to manage a set of S1 units. However, the power of the model derives from its recursive nature. Each S1, consisting of an operational element and it's

management unit, is expected to develop a similar VSM structure, consequently, the structure of systems is open ended in both directions and may be pursued either upwards to ever wider encompassing systems or downwards to ever smaller units. However, at each level the same structure of systems would occur although their detail would necessarily differ depending on context.

The value of assuming such a viewpoint is in the immediate provision not only of the outline architecture that the autonomic software system itself must assume, namely that of the Viable System Model, but also the identification of the requisite communication links to bind the system to the organization. [13][17].

VSM offers a valuable way to model the total system from a cybernetic model of the organization yet we found it difficult to translate into the bottom up approach depicted in Figs. 1 & 2. It still has value in looking at the overall complex design [13][17]. We found another simpler approach related better from an older perspective for an intelligent machine design Fig. 8 [18],[19].

The layers are:

1. Reaction—lowest level, where no learning occurs but there is immediate response to state information coming from sensory systems.
2. Routine—middle level, where largely routine evaluation and planning behaviors take place. Input is received from sensors as well as from the reaction level and reflection level. This level of assessment results in three dimensions of affect and emotion values: positive affect, negative affect, and (energetic) arousal.
3. Reflection—top level, receives no sensory input or has no motor output; input is received from below. Reflection is a meta-process, whereby the mind deliberates about itself. Essentially, operations at this level look at the system's representations of its experiences, its current behavior, its current environment, etc.

Input from, and output to, the environment only takes place within the reflex and routine layers. One may consider that reaction level essentially sits within the "hard" engineering domain, monitoring the current state of both the machine and its environment, with rapid reaction to changing circumstances; and, that the reflection level may reside within the AI domain utilizing its techniques to consider the behavior of the system and learn new strategies. The routine level may be a cooperative mixture of both (Figure 3).

This high-level intelligent machine design is appropriate for autonomic systems as depicted here since the case has been made for the dynamics of responses including reflex reactions and also for reflection of the self-managing behavior.

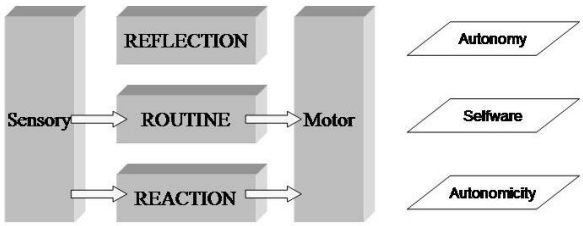

Figure 8. Comparing intelligent machine design and system level autonomy and autonomicity

Some researchers hold the perception that autonomic computing resides solely within the domain of the reaction layer. This is understandable due to the metaphoric link with the autonomic nervous system, where no conscious or cognitive activity takes place. These researchers would point to other biologically-inspired computing (also referred to as nature-inspired computing, organic computing, etc.) as providing such higher level cognitive approaches for instance as in swarm intelligence. Within the autonomic computing research community, autonomicity is not normally considered to imply this narrower view. Essentially, the autonomic self-managing metaphor is considered to aim for a user/manager to be able to set high-level policies, while the system achieves the goals. Similar overarching views exist in other related initiatives and, increasingly, they are influencing each other.

In terms of autonomy and autonomicity, autonomy may be considered as being self-governing while autonomicity is considered being self-managing. At the element level, an element will have some autonomy and autonomic properties, since to self-manage implies some autonomy, while to provide a dependable autonomous element requires such autonomic properties as self-healing along with the element's self-directed task. From this perspective, it would appear that the separation of autonomy and autonomicity as characteristics will decrease in the future and eventually will become negligible. On the other hand, at the system level if one considers again the three tiers of the intelligent machine design (reaction, routine, and reflection) and accepts the narrower view of autonomicity, there is a potential correlation between the levels. That is, the reaction level correlates with autonomicity, and the reflection level with autonomy, as in self-governing of the self-managing policies within the system. In the end, different classifications or different perspectives on the matter will be academic unless they assist and inspire new means to achieve the self-managing vision [19].

This perspective has enabled us to build in reaction, routine and reflection into the Autonomic Robotic Element and resulting system. One such example is that of monitoring the path of the robot over time and self-adjusting in field any misalignment [11].

5. SPAAACE-Ware

This SPAAACE-Ware (Self- Properties Autonomic Apoptotic Autonomous Computing Environments Software) is an InvestNI funded Proof of Concept (PoC) project with the intent to increase the Technology Ready Level (TRL) of our Autonomic Technology jointly patented with NASA e.g. [24]. This PoC will consist of middleware to enable easier development of the autonomic software for missions.

One example, the development of an autonomic element to provide self-management (such as Fig. 1) with the Pulse-Beat Monitoring element for reflex reactions and monitoring cooperation between Autonomic Elements. The PBM extends the principle of Heart-Beat Monitoring (HBM) elements that provide an "I am alive" signal. The Pulse-Beat Monitoring element is being geared specifically towards space software arena and will provide vital information on the health and activity of the environment, both internal systems environment and external operating environment. As has been highlighted previously, to ensure system robustness, the autonomic environment requires the self-managing elements to communicate with each other regarding the various self-activities and environment conditions. These communications between autonomic elements should also include a reflex signal, which a pulse monitor—with the capability to encode system health and urgency signals as a pulse. Just as a human heart has a double beat, the pulse monitor has an encoded double beat—a self health/urgency measure and an environment health/urgency measure [19] that corresponds with the autonomic element's self- and environmental-awareness logical control loops (Fig. 1), for instance providing reflex autonomic reactions between craft on a mission.

The Pulse Beat Monitoring (PBM) element incorporates reflex/urgency/health indicators and provides an "I am healthy" signal 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. From this pulse monitor the system can be self-managed by restarting systems or subsystems, check-pointing them, stopping its operations, disabling itself, or having other systems disregard or weighting data coming from the unhealthy system. In addition, other information can also be included that will provide a view of the activity of parts of the system. For example, if one part of a system starts receiving data (e.g., certain acoustic signals, seismic, etc.) over a threshold and becomes more active, a warning can be sent that an event may be occurring even before data starts arriving.

These patented technologies are being implemented as a proof of concept library focusing on the Autonomic (self-managing) capabilities and further developed into a sampler space sector application proof of concept. It is intended that the libraries as far as possible will be generic to be used in future work and other commercialisation opportunities beyond the space sector, as the original research and prototypes have demonstrated (in clusters, telecommunications, grid & cloud computing etc).

The popularity of the CubeSat approach highlights the need for such self-managing software.

6. CONCLUSION

This paper introduced at a high level the concept of *Autonomic Robotics* based on the Autonomic Computing paradigm. It is the belief of the authors that the AC paradigm will offer a systematic means to obtain self-managing and autonomous robotic software.

The first area to be considered in this paper was the self-management collaboration and cooperation between system entities. Many space agencies, including ESA, are activity moving away from singular space craft mission paradigm to multiple craft missions. From constellation missions of three cooperating craft to the NASA "ANTS" (Autonomous Nano-Technology Swarm) concept mission with potentially 1000's of craft working as a swarm. With the larger the scale of entities the more reliance on autonomy and self-management techniques. We briefly described how in this project we are investigating autonomic cooperation and collaborating strategies between elements from a small cluster to large swarm scale. The second area of interest was specifically focused on the internal self-managing cooperation with an entity and the best architecture to enable vertical orchestration within the system in a scalable fashion to enable the first. Lastly, a research project for making the programming of such systems less reliant on bespoke software development by deriving standard self-management activities and providing standard software artefacts and middleware was briefly described.

ACKNOWLEDGEMENTS

Some of the PhD research projects are funded by the Northern Ireland Department for Employment and Learning (DEL http://www.delni.gov.uk). The SPAAACE-Ware Proof of Concept project is funded by InvestNI. Some of the technologies described in this paper are patent protected or pending by NASA and assigned to the United States government.

REFERENCES

[1] Truszkowski, W., Hallock, H.L., Rouff, C., Karlin, J., Rash, J., Hinchey, M.G. and Sterritt, Roy (2010) Autonomous and Autonomic Systems With Applications to NASA Intelligent Spacecraft Operations and Exploration Systems. NASA Monographs in

Systems and Software Engineering, Springer-Verlag. 289 pp ISBN 978-1-84628-232-4
[2] Sterritt R, (Dec 2002) "Towards Autonomic Computing: Effective Event Management", Proceedings of 27th Annual IEEE/NASA Software Engineering Workshop (SEW), Maryland, USA, December 3-5, IEEE Computer Society, Pages 40-47
[3] Hinchey MG, Sterritt R, (Feb 2006) "Self-Managing Software", Computer, Vol. 39, No. 2, IEEE Computer Society, ISSN 0018-9162, Pages 107-109
[4] Sterritt R, Bustard DW, (Apr 2003) "Autonomic Computing-a Means of Achieving Dependability?", Proceedings of IEEE International Conference on the Engineering of Computer Based Systems (ECBS'03), Huntsville, Alabama, USA, April 7-11, IEEE CS Press, Pages 247-251
[5] Sterritt R, Bustard DW, (Sep 2003) "Towards an Autonomic Computing Environment", Proceedings of IEEE DEXA 2003 Workshops - 1st International Workshop on Autonomic Computing Systems, Prague, Czech Republic, September 1-5, 2003, Pages 694-698
[6] Horn P., "Autonomic computing: IBM perspective on the state of information technology", IBM T.J. Watson Labs, NY, 15th October 2001. Presented at AGENDA 2001, Scotsdale, AR. (available http://www.research.ibm.com/autonomic/), 2001
[7] Kephart J., Chess D.. The vision of autonomic computing. IEEE Computer 36(1), pages 41–52. January 2003.
[8] Sterritt R, (2005) "Autonomic Computing", Innovations in Systems and Software Engineering, Vol. 1, No. 1, Springer, ISSN 1614-5046, Pages 79-88
Papers are to be submitted in PDF format only. See conference-specific instructions for further details.
[9] Saunders, Catherine, Sterritt, Roy and Wilkie, George (2014) The Utility of Robot Sensory Devices in a Collaborative Autonomic Environment. In: 11th IEEE International Conference and Workshops on the Engineering of Autonomic & Autonomous , APL, Laurel, Maryland, USA. IEEE CS. 8 pp
[10] Saunders, Catherine, Sterritt, Roy and Wilkie, George (2015) Computer Vision Techniques for Autonomic Collaboration between Mobile Robots. In: 7th International Conference on Adaptive and Self-Adaptive Systems and Applications, Nice, France. IARIA. 7 pp
[11] Doran, Martin, Sterritt, Roy and Wilkie, George (2014) Autonomic Wheel Alignment for Mobile Robots. In: 11th IEEE International Conference and Workshops on the Engineering of Autonomic & Autonomous , APL, Laurel, Maryland, USA. IEEE CS. 6 pp.
[12] Sterritt, Roy; Bantz, D.F., "Personal autonomic computing reflex reactions and self-healing," Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on , vol.36, no.3, pp.304,314, May 2006 doi: 10.1109/TSMCC.2006.871592
[13] Taleb-Bendiab, A.; Bustard, D.W.; Sterritt, R.; Laws, A.G.; Keenan, F., "Model-Based Self-Managing Systems Engineering," Database and Expert Systems Applications, 2005. Proceedings. Sixteenth International Workshop on , vol., no., pp.155,159, 26-26 Aug. 2005
doi: 10.1109/DEXA.2005.137
[14] Beer, S., The Heart of the Enterprise, John Wiley & Sons, Chichester, 1979.
[15] Beer, S., Brain of the Firm, 2nd ed, John Wiley & Sons, Chichester, 1981.
[16] Beer, S., The Viable System Model: Its Provenance, Development, Methodology and Pathology, Journal of the Operational Research Society, Vol. 35, pp. 7-26, 1984.
[17] Bustard, D.; Sterritt, R.; Taleb-Bendiab, A.; Laws, A.; Randles, M.; Keenan, F., "Towards a systemic approach to autonomic systems engineering," Engineering of Computer-Based Systems, 2005. ECBS '05. 12th IEEE International Conference and Workshops on the , vol., no., pp.465,472, 4-7 April 2005
doi: 10.1109/ECBS.2005.71
[18] Norman, D.A., Ortony, A. and Russell, D.M., Affect and Machine Design: Lessons for the Development of Autonomous Machines, *IBM Systems Journal*, 42(1):38–44.
[19] Sterritt R, Hinchey MG, (Jun 2005) "SPAACE ::Self- Properties for an Autonomous & Autonomic Computing Environment", Proceedings of Autonomic & Autonomous Space Exploration Systems (A&A-SES-1) at 2005 International Conference on Software Engineering Research and Practice (SERP'05), Las Vegas, June 27-30, CSREA Press, Pages 3-8
[20] Emil Vassev, Roy Sterritt, Christopher Rouff, Mike Hinchey, "Swarm Technology at NASA: Building Resilient Systems," IT Professional, vol. 14, no. 2, pp. 36-42, March-April, 2012
[21] Sterritt, Roy; Hinchey, M., "Why computer-based systems should be autonomic," Engineering of Computer-Based Systems, 2005. ECBS '05. 12th IEEE International Conference and Workshops on the , vol., no., pp.406,412, 4-7 April 2005
doi: 10.1109/ECBS.2005.75
[22] Sterritt, Roy, "Pulse monitoring: extending the health-check for the autonomic grid," Industrial Informatics, 2003. INDIN 2003. Proceedings. IEEE International Conference on , vol., no., pp.433,440, 21-24 Aug. 2003
doi: 10.1109/INDIN.2003.1300375
[23] Hinchey, M.G.; Sterritt, R.; Rouff, C., "Swarms and Swarm Intelligence," Computer , vol.40, no.4, pp.111,113, April 2007
doi: 10.1109/MC.2007.144
[24] NASA, Sterritt, Hinchey (17th March 2015) Autonomic and Apoptopic Systems in Computing, Robotics, and Security. [US Patent 8983882]