The Knowledge Engineering Review, Vol. 00:0, 1–24. © 2004, Cambridge University Press
DOI: 10.1017/S0000000000000000 Printed in the United Kingdom

A Health-Check Model for Autonomic Systems Based on a Pulse Monitor

ROY STERRITT[1] and DAVE BUSTARD[2]

[1]*School of Computing and Mathematics, Jordanstown Campus, Newtownabbey, County Antrim, BT37 0QB, Northern Ireland.*
E-mail: r.sterritt@ulster.ac.uk
[2]*School of Computing and Information Engineering, Coleraine Campus, Coleraine, County Londonderry, BT52 1SA, Northern Ireland.*
E-mail: dw.bustard@ulster.ac.uk

Abstract

Like the autonomic responses in the human body, autonomic computing systems recognise their own *health* problems and where possible respond to correct them. Failing that, external help is required. The purpose of this paper is to consider how autonomic systems might be structured to facilitate health monitoring. The approach uses a 'pulse' monitor for each autonomic element, which provides a reflex reaction facility and basic information on the current state (health) of that element. The pulse mechanism extends the NASA beacon monitor concept. The different ways that pulse information might be communicated and used are examined. The discussion is illustrated with a personal computing example.

1 Introduction

Autonomic Computing, first proposed by IBM in 2001 (Horn 2001) has become established as a valuable approach to the design of robust computing systems (Mainsah 2002, IBM 2003). The general concepts involved are summarised in Figure 1 (Sterritt and Bustard 2003a). An autonomic system is self-managing, meaning that it performs a range of operations to ensure its own viability. These primarily include self-protection, self-configuration, self-healing and self-optimisation. Self-healing is concerned with ensuring effective recovery when a fault occurs. This means successfully identifying the fault and then, where possible, repairing it. Also, there should be minimal disruption to users, avoiding loss of data and significant delays in processing. Self-optimisation means that a system is aware of its ideal performance, can measure its current performance against that ideal and has strategies for attempting improvements. A self-protecting system will defend itself from accidental or malicious external attack. This means being aware of potential threats and having ways of handling those threats. This may include self-healing actions if an attack is successful, and perhaps some self-optimisation to increase protection. Finally, self-configuration is a system's ability to readjust itself automatically to changing circumstances. This may simply be in support of ongoing development or to assist in self-healing, self-optimisation or self-protection. To achieve these objectives a system must be aware of its internal state (self-aware) and current external operating conditions (environment-aware). Changing circumstances, internal or external, are detected through self-monitoring and adaptations made accordingly (self-adjusting). In more detail, this means a system having 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 on how these may be adjusted. Such autonomic systems can be created through a traditional systems engineering design process

(Bustard *et al.*, 2005) or through inclusion of adaptive learning logic (IJCAI AI&AC Workshop 2003).

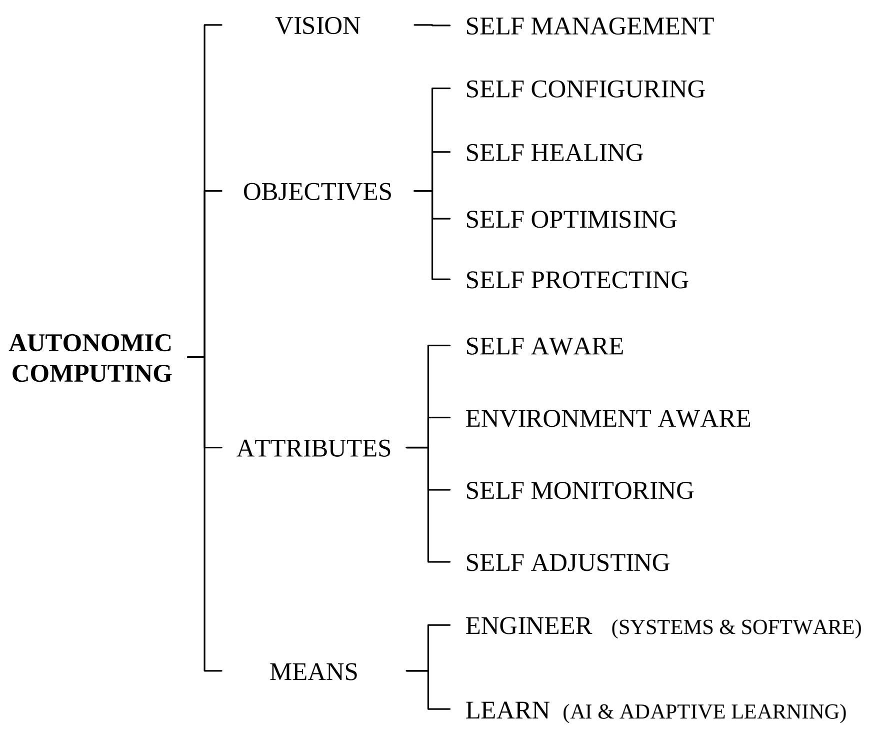

Figure 1 Autonomic Computing Concepts

The concept of self-managing systems has attracted significant attention from both industry and academia. In addition to the autonomic computing work at IBM, there are similar industry initiatives at HP (Adaptive Infrastructure), Sun (N1) and Microsoft (Dynamic Systems Initiative). Examples of academic involvement include work at Rutgers (active middleware) (Bhat and Parashar, 2003), CMU (self-healing systems)(Garlan *et al.*, 2003), Columbia (retrofitting legacy systems) (Kaiser et al., 2002), and Imperial College (autonomic management of ubiquitous eHealth systems) (Lupu *et al.*, 2003).

Achieving the grand vision of autonomic computing is likely to involve contributions from research in many existing fields, including systems management, distributed computing, networking, operational research, software engineering, artificial intelligence, agent technology and control theory (Ganek and Corbi, 2003). Dependable and fault tolerant computing should be especially influential, as dependability covers many system properties relevant to autonomic behavior, such as reliability, availability, safety, security, survivability and maintainability (Avizienis *et al.*, 2000, Randell 2000, Sterritt and Bustard 2003a, Avizienis *et al.*, 2004).

This paper is particularly concerned with the way in which an autonomic system can provide information on its state for use by dependent autonomic elements or elements responsible for health monitoring. The next section of the paper presents a general architecture model for autonomic systems. This introduces the concept of a pulse monitor for each autonomic element through which it can report a summary of its general health. Details of the operation and use of the pulse monitor, based on the NASA beacon monitor (Wyatt *et al.*, 1998, 1999), are covered in the following section. The paper concludes with a consideration of related work which utilises these concepts in various application areas.

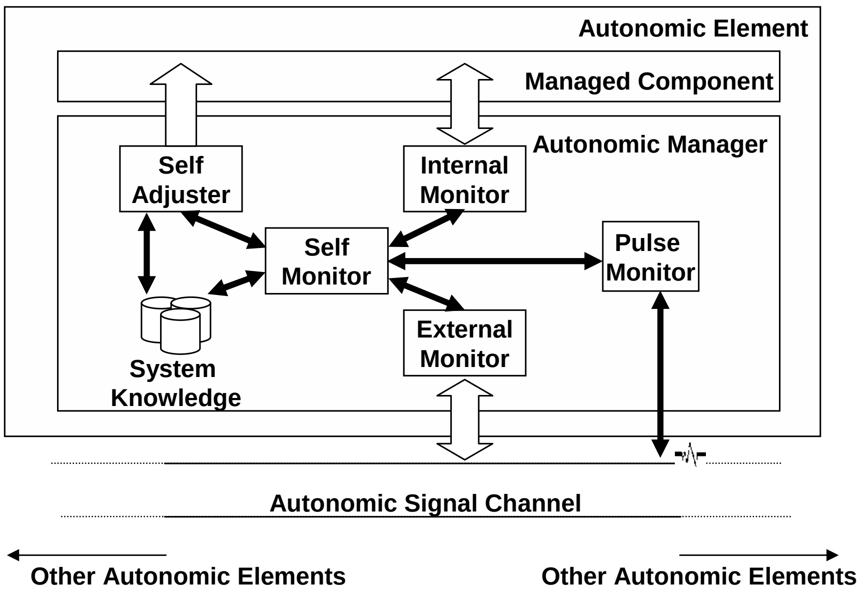

Figure 2 Abstract Architecture of an Autonomic Element

2 Autonomic Architecture

An autonomic system requires open standards to understand and communicate with other systems in a heterogeneous environment (Horn 2001, IBM 2003). Communication standards are also valuable within an autonomic system to facilitate system design and construction. For conceptual simplicity and convenience, let us assume that an autonomic computing system is made up of a connected set of autonomic elements, each with the same basic structure and mode of operation. Figure 2 is a proposed architecture for those elements (Sterritt and Bustard 2003b).

Here, an autonomic element is made up of a managed component and an autonomic manager. The self-monitor actively observes the state of the component (internal monitor) and its external environment (external monitor), drawing conclusions using information in the system knowledge base. If necessary, this can lead to adjustments to the managed component (self-adjuster). The external monitor observes the state of the environment through an autonomic signal channel (for instance an asynchronous event bus), which provides linkage to other autonomic managers. This linkage may be virtual (in the same physical system), peer-to-peer, client-server (Bantz *et al.*, 2003) or a Grid connection (Deen *et al.*, 2003).

A novel feature in this design is the suggested use of a pulse monitor (Sterritt 2002, 2003b). This is used to communicate basic information about the state of each managed component to other relevant autonomic elements. In essence, its purpose is to provide a simple, general mechanism for reporting the 'health' of a managed component. This extends the biological autonomic metaphor. The design of such a pulse monitor is examined in detail in the next section.

3 Pulse Monitoring

A standardised pulse monitor can provide a convenient framework for the design of fault-handling software in autonomic systems. Ideally, it should be relevant to all autonomic elements and have a simple conceptual base. One approach is to assume that the pulse monitor sends out a steady 'heartbeat'. When detected, this indicates that the autonomic element is 'alive' and when missing indicates an operational problem. It is then a small step to introduce the notion of a variable pulse rate to indicate intermediate states. NASA has used this general idea in the design of the Beacon Monitor for autonomy in space missions (Sherwood *et al.*, 1999).

3.1 NASA Beacon Monitor

NASA missions, particularly those to deep space, have had to increase onboard autonomy because of the lengthy lag-time between a craft encountering new situations and the round-trip delay in obtaining guidance from mission control (Swartwout 1998, Wyatt et al. 1998,). Two of the first missions to take this approach were DS1 (Deep Space 1) (Wyatt *et al.*, 1998) and the Mars Pathfinder (Muscettola *et al.*, 1998). The Beacon Monitor concept, first used in the DS1 mission work, automates the routine task of health monitoring, moving many responsibilities from ground to the spacecraft. The spacecraft sends a beacon signal to the ground indicating how urgent it is to track the spacecraft for telemetry. A tone is used to indicate the degree of urgency involved. Five different states are used, as follows (Sherwood *et al.*, 1999):

Nominal: functions as expected; no need to establish a downlink (a communications link to download data).

Interesting: interesting, non-urgent event; establish communications when convenient.

Important: communications need to take place quickly or the state could deteriorate.

Urgent: emergency: a critical component has failed; it is not possible to recover autonomously and intervention is needed immediately.

No Tone: beacon mode is not operating.

There are a number of points to note here in relating the beacon approach to the needs of pulse monitoring in autonomic systems. The first is that there are two levels of communication involved in each case. The spacecraft uses both the beacon signal and, when necessary, direct communication, to downlink telemetry information for analysis at mission control. The same is true of autonomic systems. If an autonomic element indicates that it is operating normally then no further action is required. If a problem is indicated, however, then the autonomic manager (through the external monitor) must be interrogated further to clarify the issue. The type of information required will be specific to the problem detected and the particular functions/services provided by the managed component.

Another similarity is that the behavior of mission control need not be dictated by the signal from the beacon monitor. They can make direct contact with the spacecraft to obtain information whenever they wish, even if there are no apparent problems present. The same is true of communicating autonomic elements.

There are, however, some significant differences between the NASA situation and the general needs of autonomic systems. One is that interaction between the spacecraft and mission control is point-to-point, whereas many-to-many links are present in autonomic systems. Although, in principle, all autonomic elements could receive pulse signals from every other element in the system, this would quickly become overwhelming in systems of any reasonable size. The implication, therefore, is that autonomic elements need to establish explicit communication connections with each other before communication occurs. Using the biological metaphor, this can be considered equivalent to 'taking a pulse'.

Another difference is that autonomic systems have communication sequences to consider. For example, if an autonomic element detects that an element on which it relies is having problems, then it needs to report any consequential secondary problems it is experiencing. This can have a substantial knock-on ripple effect throughout an autonomic system when a serious failure occurs. It is important that this failure information be transmitted quickly to all affected elements, corresponding to the biological 'reflex reaction'. To instigate a repair, it will also be helpful to have primary and secondary problems clearly distinguished.

3.2 Pulse Monitor States

Based on the requirements for pulse monitoring outlined in the preceding sub-section, and taking account of the NASA beacon monitor states used, the following five-state model is proposed:

Healthy the autonomic element is operating as required, with environment conditions as expected.

Active the autonomic element is operating as required, but work is being done to improve its operation, triggered either by the results of an internal analysis or by recognition of environmental changes.

Injured (Primary and/or Secondary): the autonomic element is experiencing difficulties affecting its performance. This may be due to internal problems (primary) and/or difficulties with its environment, such as problems with other elements on which it depends (secondary).

Incapacitated (Primary and/or Secondary): the autonomic element is no longer able to function. This may be due to serious internal problems (primary) and/or extreme difficulties with its environment, such as the failure of other elements on which it depends (secondary).

Dead: the autonomic element is not communicating.

To illustrate what these states mean consider the simple example of an autonomic element responsible for managing a file server:

Healthy: the file server is operating at the performance levels expected, saving and retrieving files successfully in all circumstances, and making security back-ups where required.

Active: the file server is making adjustments to file locations, perhaps to improve performance or in response to the addition or (controlled) removal of available storage.

Injured (Primary and/or Secondary): an example of a primary problem is the file server detecting a hardware failure resulting in the loss of files. This 'injury' may be healed autonomically through retrieval of backup copies. Shortage of file space would also be treated as a primary wound. An example of a secondary problem is difficulty in retrieving files through the autonomic element managing DVD backup storage. Note that this may occur at the same time as the primary problem of file loss.

Incapacitated (Primary and/or Secondary): the file server has run out of disk apace. A similar problem might occur with a DVD backup management autonomic element, although this alone would only 'injure' the file server.

Dead: the file server cannot be contacted because, for example, the server software has crashed, the server host machine has failed, or the server has been disconnected from the network unexpectedly. The next two subsections consider how the pulse monitor might be implemented and illustrate its use in a personal computing system, expanding on the file server example.

3.3 Pulse Monitor Implementation

The NASA Beacon Monitor uses a variable tone to indicate its different states. An autonomic pulse monitor could use a similar mechanism, indicating different states through different frequencies of communication. This has the advantage of being consistent with the biological metaphor, in which 'no signal' means 'death' and higher pulse rates indicate greater degrees of stress. In computing terms, however, each signal can easily include extra information to avoid the receiving element having to deduce a state change. Also, once the receiving element has acknowledged receipt of a problem there seems little point in repeating the message frequently, consuming processing power that might be needed to provide assistance. It is still important, however, to provide the basic carrier heartbeat to be able to recognise quickly when an autonomic element has 'died'. This must be communicated rapidly through the system to avoid further problems and perhaps contact human operators who may be needed to resolve the problem.

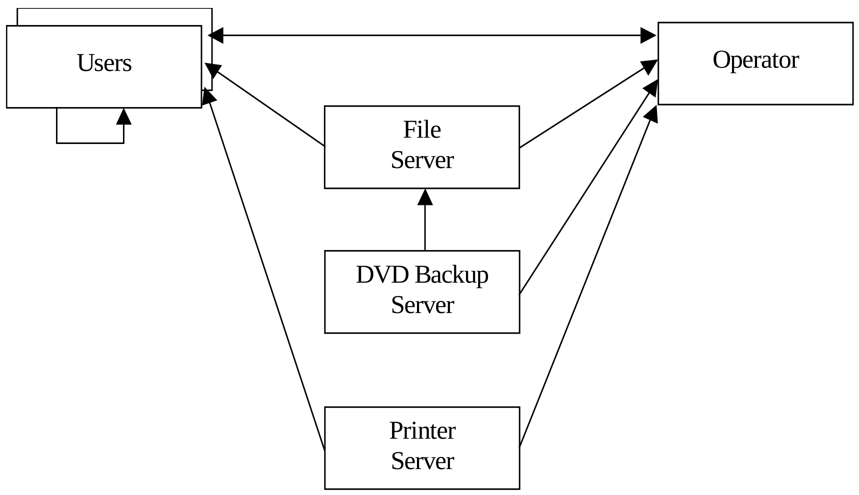

Figure 3 Pulse Monitor Connections in a Personal Computing System

3.4 Personal Computing Example

To illustrate the way that the 'health' of an autonomic system might be monitored consider the following personal computing example. A local network is set up as shown in Figure 3. This comprises autonomic elements for: (i) each connected user; (ii) a shared file server; (iii) a shared printer server; (iv) a DVD backup server; and (v) an operator with overall responsibility for the network. The lines in the diagram indicate monitoring relationships between autonomic elements. These indicate, for example, that each user monitors the printer server, the operator monitors the users, and the users monitor the operator.

In general there are a number of ways in which the 'health' of autonomic elements might be monitored. One is to have dedicated system elements for that purpose. Another, more robust approach, is to share this responsibility among the autonomic elements so that, in effect, they monitor each other. It is convenient, for example, for an autonomic element to monitor the health of any other system element on which it depends. To improve robustness it is also desirable to have every autonomic element monitored at least twice, as indicated in the example. The users in this case might be connected in a two-way ring, with each monitoring their left and right neighbour.

Most autonomic research, especially that in industry, has so far focused on server management (Bantz and Frank 2003), since requirements for reliability have increased, while servers have become more complex and hence more difficult to maintain. The contribution of autonomic computing to personal computing is different being much less about achieving optimum performance or exploiting redundancy and more about simplifying use of the equipment and the associated services involved (Bantz and Frank 2003, Sterritt and Bantz 2004). This is an important area because, potentially, it can affect every computer user. Issues to be considered include, for example:

- How to design file servers to help users become better organised, and hence make better use of the storage space available.
- How to deal effectively with laptop connections to a network to facilitate users making the connection while protecting the network from viruses and other threats.
- How to involve users in autonomic activity, using their technical knowledge to recover from operational problems (e.g. a hardware problem while the operator is unavailable) or provide assistance for other users.

- How to design autonomic elements to respond effectively to system problems that occur. Certainly it is important for elements to be informed quickly (reflex reaction) but what action should then follow? As in biological systems, a short-term reflex action coupled with a longer-term healing process seems the necessary strategy to adopt (Bapty *et al.*, 2003). Human reflex reactions enable a rapid response to pain, such as when a hot object is touched. In computing terms, it is likely that each autonomic system will have to reconfigure itself in response to problems encountered, while maintaining its overall operation as far as possible. This may result in the system operating with a reduced set of resources and services (Bapty *et al.*, 2003).

4 Related Work

The section briefly discusses two application areas where the pulse monitor may be considered beneficial.

4.1 Grid Heartbeat Monitor

The Globus Open Grid Services Architecture (OGSA) has a health-check facility known as the Globus Heartbeat Monitor, which is designed to provide a simple, highly reliable mechanism for monitoring processes (Stelling, 1998). The Globus OGSA specifies three aspects of the heartbeat monitor: a Client Library, a Local Monitor, and a Data Collector. Essentially, a Local Monitor runs on each host checking and reporting on the status of its local processes and the overall system, generating 'I-am-alive' messages (heartbeats) to confirm working health.

The Client Library enables processes to register when they are activated within the Local Monitor and sign out when they terminate. Each Local Monitor periodically executes a review cycle in which it checks the status of the client processes it is monitoring, updates the local status information of those processes, and sends a report on each process to one or more external agents (Data Collectors) specified at registration. There can be any number of Data Collectors defined, but typically there is at least one for tracking all of the monitored processes associated with the computing environment, and one for each distributed application. A Data Collector infers that monitored components have failed or are unavailable, based on reports it is expecting from Local Monitors but has not received (missing heartbeats) (Stelling, 1998).

This mechanism could usefully be extended with the pulse concept to provide more information on operational health. For instance, a Local Monitor recognising that a process is no longer executing optimally might send this health indicator pulse to the Data Collector. The Data Collector would then have the benefit of additional information in assessing the heath of a service and be able to detect deterioration prior to complete failure, taking remedial action as appropriate to reduce the risk identified (Sterritt 2003b).

4.2 Fault Management in Telecommunication Systems

These approaches have been investigated in relation to more effective fault management in telecommunication systems, particularly in exploring the relationship between a rapid response to faults that occur (reflex reaction) and the slower analysis and healing process that follows. Figure 4 shows a typical fault management architecture for a large telecommunications network (Sterritt *et al.*, 2004). Faults occur in network elements at the bottom of the reporting tree in the figure. Many different network technologies are involved (e.g. SDH (SONET), PDH, ATM, IP) and each has a specific fault manager. The network technologies are inter-dependent to some extent however, so it is only at the next layer up, the cross technology network fault manager, that the faults reported can be correlated to produce a coherent understanding of the root cause of the problem.

Once the root cause has been determined, either automatically or through operator assistance, the fault is assigned a trouble ticket and handed over to field force management for attention.

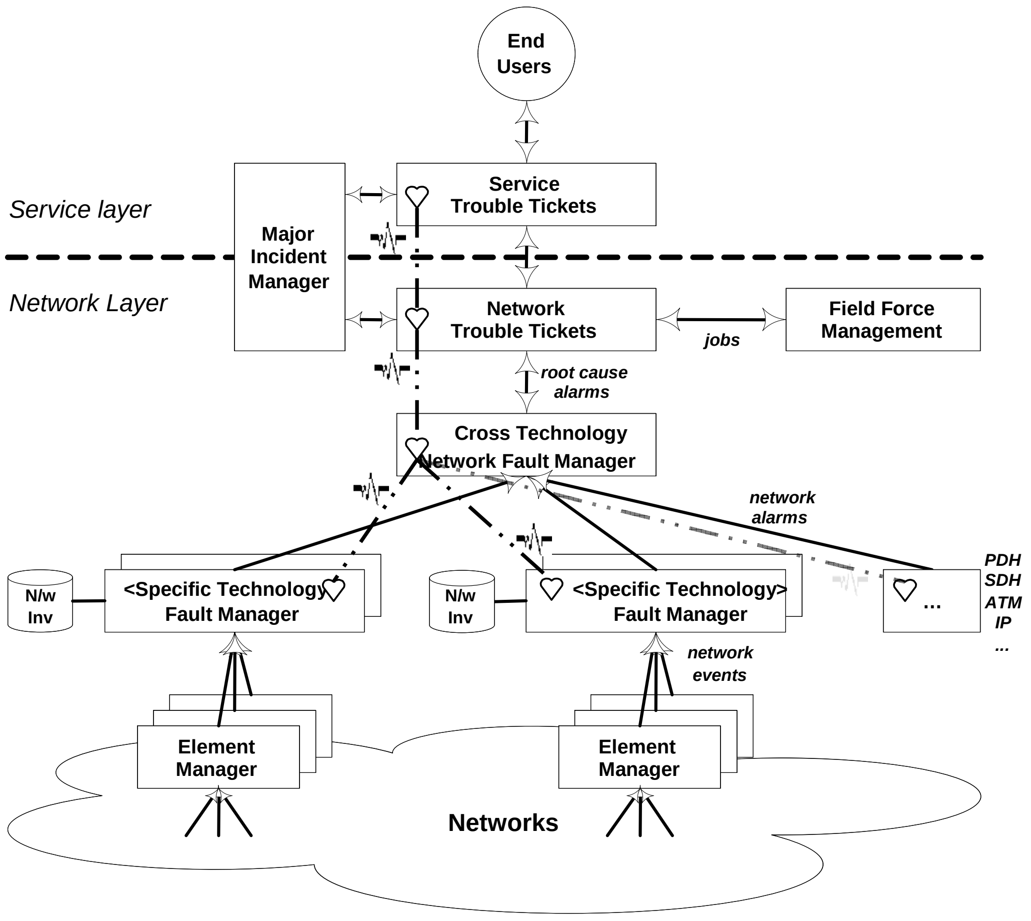

Figure 4 Large Telecommunications Fault Management Architecture

Because of the amount of processing involved in the various layers shown it can take up to 15 minutes for details of a fault to reach the service layer, by which time customers may have already reported the problem.

A study took place in 2003 at British Telecom (Sterritt 2003a, Sterritt *et al.*, 2004, 2005) to examine the potential of utilising autonomicity to assist. It was found that this delay could potentially be reduced substantially by having pulse monitors in each (autonomic) fault management element that can supply reflex health signal about fault traffic ahead of details on the exact source of the problem emerging through the hierarchy. This may be achieved through extending the existing heartbeat monitors between the managers to carry a network health indicator (the pulse), as depicted in Figure 4.

5 Conclusions

Potentially, the autonomic computing concept could have an impact on system design similar to that of the object-oriented paradigm. It promises to improve the robustness of future systems, as well as reducing complexity for users, by taking on more responsibility for identifying and handling system problems that arise. The purpose of this paper was to consider how system 'health' might be monitored. This was based on an assumption that autonomic systems were made up of a set of autonomic elements with the same basic architecture. Included in that architecture was a pulse monitor, to provide a reflex reaction and essential information on the current state of health of each autonomic element. This is an extension of the more commonly used 'heartbeat' mechanism, typically found in embedded systems and fault-tolerant computing applications.

Each pulse monitor can report one of five different states for its associated element: healthy, active, injured, incapacitated and dead. These are a variation of the states developed for the NASA Beacon Monitor, adapted to cover general computing and communications health summarisation. The use of the states was illustrated with a file server autonomic element, operating in a personal computing network. The discussion identified a number of areas for future research, particularly in personal computing. Overall, however, it seems possible that such a mechanism could be of value in the construction of almost any system.

Acknowledgements

This work was undertaken through the Centre for Software Process Technologies, which is supported by the EU Programme for Peace and Reconciliation in Northern Ireland and the Border Region of Ireland (PEACE II).

References

Avizienis, A., Laprie, J.-C., Randell, B., 2000, "Fundamental Concepts of Dependability", *UCLA CSD Report* 010028.

Avizienis, A., Laprie, J.-C., Randell, B., Landwehr, C., 2004, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Transactions on Dependable and Secure Computing*, **1(1)**, pp. 11-33, January-March.

Bantz, D.F., Bisdikian, C., Challener, D., Karidis, J.P., Mastrianni, S., Mohindra, A., Shea, D.G., Vanover, M., 2003, "Autonomic personal computing", *IBM Systems Journal*, **42(1)**, pp. 165-176.

Bantz, D.F. and Frank, D., 2003, "Challenges in Autonomic Personal Computing, with Some New Results in Automatic Configuration Management", *Proceedings of IEEE Workshop on Autonomic Computing Principles and Architectures (AUCOPA 2003) at INDIN 2003*, Banff, Alberta, Canada, 22-23 August.

Bapty, T., Neema, S., Nordstorm, S. , Shetty, S., Vashishtha, D., Overdorf, J., Sheldon, P., 2003, "Modeling and Generation Tools for Large-Scale, Real-Time Embedded Systems", *Proceedings of 10th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS)*, Huntsville, Alabama, USA, 7-10th April, pp 11-16.

Bhat, V. and Parashar, M., 2003 "Discover Middleware Substrate for Integrating Services on the Grid", *Proceedings of the 10th International Conference on High Performance Computing (HiPC '03)*, Hyderabad, India, December.

Bustard D.W., Sterritt R., Taleb-Bendiab A., Laws A., Randles M., Keenan F., (2005) "Towards a Systemic Approach to Autonomic Systems Engineering", *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, pp 465-472.

Deen, G., Lehman, T., Kaufman, J., 2003, "The Almaden OptimalGrid Project", *Proceedings of the Autonomic Computing Workshop, 5th Int. Workshop on Active Middleware Services (AMS 2003)*, Seattle, WA, pp 14-21, June.

Ganek A.G. and Corbi T.A., 2003, "The dawning of the autonomic computing era", *IBM Systems Journal*, **42(1)**, pp. 5-18.

Garlan, D., Shang-Wen Cheng, and Bradley Schmerl, 2003, "Increasing System Dependability through Architecture-based Self-repair,", in *Architecting Dependable Systems*, R. de Lemos, C. Gacek, A. Romanovsky (Eds), Springer-Verlag.

Horn, P. 2001, "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.

IBM, 2003, "An architectural blueprint for autonomic computing", *White Paper*, release:April'03.

Kaiser, G., Phil Gross, Gaurav Kc, Janak Parekh and Giuseppe Valetto, 2002, "An Approach to Autonomizing Legacy Systems" *Workshop on Self-Healing, Adaptive and Self-MANaged Systems*, June.

Lupu, E., M. Sloman, N. Dulay, J. Sventek, 2003, "AMUSE: Autonomic Management of Ubiquitous Systems for e-Health", *EPSRC project*, http://www.doc.ic.ac.uk/ ecl1/projects/AMUSE/.

Mainsah, E., 2002, "Autonomic computing: the next era of computing", *IEE Electronics Communication Engineering Journal*, **14(1)** (Feb), pp 2-3.

Muscettola, N., Nayak, P.P., Pell, B. and Williams, B., 1998, "Remote Agent: To Boldly Go Where No AI System Has Gone Before", *Artificial Intelligence* **103(1-2)** pp 5-48.

IJCAI Workshop, 2003, "AI and Autonomic Computing: Developing a Research Agenda for Self-Managing Computer Systems", *Proceedings*, Acapulco, Mexico, August 10, http://www.research.ibm.com/ACworkshop

Randell, B., 2000, "Turing Memorial Lecture Facing Up to Faults", *Comp. J.* **43(2)**, pp 95-106.

Sherwood, R., Wyatt, J., Hotz, H., Schlutsmeyer, A., Sue, M., 1999, "Lessons Learned During Implementation and Early Operations of the DS1 Beacon Monitor Experiment," *Third International Symposium on Reducing the Cost of Ground Systems and Spacecraft Operations*, Tainan, Taiwan.

Stelling,P.F., Foster, I., Kesselman, C., 1998, "Fault Detection Services for Wide Area Distributed Computations", *Retreat.*

Sterritt, R., 2002, "Towards Autonomic Computing: Effective Event Management", *Proceedings of the 27th Annual IEEE/NASA Software Engineering Workshop*, IEEE CS Press, Pages 40-47.

Sterritt, R., 2003a, "xACT: Autonomic Computing and Telecommunications", *British Telecom Research Fellowship.*

Sterritt R,, 2003b, "Pulse Monitoring: Extending the Health-check for the Autonomic GRID", *Proceedings of the IEEE Workshop on Autonomic Computing Principles and Architectures (AUCOPA' 2003) at IEEE Int. Conf. Industrial Informatics (INDIN 2003)*, Banff, Alberta, Canada, 22-23 August.

Sterritt R., and Bantz D.F., (2004) "PAC-MEN: Personal Autonomic Computing Monitoring Environments", *Proceedings of IEEE DEXA 2004 Workshops - 2nd International Workshop on Self-Adaptive and Autonomic Computing Systems (SAACS 04)*, Zaragoza, Spain, August 30th - September 3rd, IEEE CS Press, Pages 737-741.

Sterritt, R., and Bustard, D.W., 2003a, "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, pp 247-251.

Sterritt, R., and Bustard, D.W., 2003b, "Towards an Autonomic Computing Environment", *1st International Workshop on Autonomic Computing Systems at 14th International Conference on Database and Expert Systems Applications (DEXA'2003).* Prague, Czech Republic Sept. 1-5.

Sterritt R, Gunning D, Meban A, Henning P, 2004, "Exploring Autonomic Options in an Unified Fault Management Architecture through Reflex Reactions via Pulse Monitoring", *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, Pages 449-455.

Sterritt R, Bustard DW, Gunning D, Henning P, 2005, "Autonomic communications and the reflex unified fault management architecture", *Advanced Engineering Informatics*, Elsevier, **19** pp 189-198.

Swartwout, M.A., 1998, "Engineering Data Summaries for Space Missions" *SSDL.*

Wyatt, J., Hotz, H., Sherwood, R., Szijjaro, J., Sue, M., 1998, "Beacon Monitor Operations on the Deep Space One Mission", *5th Int. Sym. AI, Robotics and Automation in Space*, Tokyo, Japan.

Wyatt, J., Sherwood, R., Sue, M., Szijjarto, J., 1999, "Flight Validation of On-Demand Operations: The Deep Space One Beacon Monitor Operations Experiment", *5th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS '99)*, ESTEC, Noordwijk, The Netherlands, 1-3 June.