Towards Self-Managing Web Sites: The Link Integrity Problem

David Bustard, Adrian Moore, Declan Higgins, David Ayre
School of Computing and Information Engineering, University of Ulster, UK
{dw.bustard, aa.moore}@ulster.ac.uk

Abstract

Most web sites contain faulty links. These affect the perceived quality of the information on the site and the reliability of any associated services. Despite such motivation, and a wide discussion of the issue over many years, achieving a reasonable level of link integrity across the web remains an open challenge. This paper describes some work towards clarifying and resolving that challenge. The approach taken in this paper involves the embedding of automatic checks for link integrity in each site. The problem is examined using Soft Systems Methodology, in the general context of creating and maintaining a web site to provide user information and associated services. Some experimental work towards implementing such a system is described, discussing opportunities for autonomic behaviour.

1. Introduction

From a user perspective, a *web site* is a collection of *pages* (of information) that are accessible via a *browser* over the *internet*. Each page typically contains references or *links* to resources used on the page, such as images, or connections to other pages. These may be *internal* to the same web site, or *external* to it.

A user expects each link to be *valid*, meaning that it leads successfully to the intended page or other resource. In practice, however, faults are relatively common, with the problem often reported as "error 404: not found", indicating that communication has been made with a web server but the resource required could not be located. Such faults may be present when a page is first created or occur later, when a change is made. The change may be to the link on the host page or result from a resource being moved, made unavailable through a change in access rights, or its content modified to an extent that it becomes inappropriate.

This is essentially a *link integrity* problem [1, 2], but for the web is more popularly known as *link rot* [3], implying a process of decay [4]. Analyses of the server logs on specific web sites have found that "file does not exist" is by far the most significant problem detected, accounting for about 90% of the faults reported [5, 6], with "permission denied", taking the total close to 100% [5].

While broken links affect everyone, some rely on them more than others; for example, failure of a business web site has a bearing on the revenue of the organisation concerned. Also professional researchers and academics find this problem frustrating, because even after a few months many citations in a publication will have moved or been deleted. Such inactive article references affect the credibility of the work and hence its value to the community [7].

In the 7th GVU WWW User Survey in 1997 [8] half of the respondents described broken links as a "significant problem"—one with no immediate solution. By the 9th Survey [8], a year later, the level of dissatisfaction had increased with the analysts noting that "although solutions for dealing with broken links are well known to web designers…most sites don't seem to employ these techniques." The situation is no better today. Indeed the number of broken links has increased with the growth of the Internet and, unfortunately, the problem has become so commonplace that link rot now appears accepted as a feature of the web [3].

There is clearly potential, however, to improve link integrity substantially but not, apparently, if left to personal discipline. Tool support seems essential. One approach is to introduce monitoring code to each web site to check link integrity (Figure 1). This is not a new suggestion but simply a return to what was expected in the implementation of early hypertext systems [1].

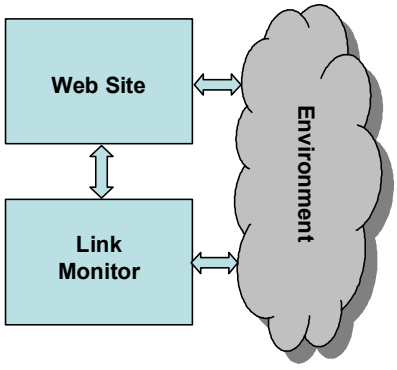

Figure 1 Autonomic Web Site Structure

Figure 1 is also recognisable as an *S1 operational sub-system*, as defined by Stafford Beer in the *Viable System Model* [9]; that is, a system providing a distinct product or service, containing an *operational element* (web site) controlled by a *management process* (link

Fourth IEEE International Workshop on Engineering of Autonomic and Autonomous Systems (EASe'07)
0-7695-2809-0/07 $20.00 © 2007 IEEE

monitor) and in contact with the *operational environment*. Seeing the web site as part of a general system can help in determining how it should be constructed and maintained.

Figure 1 also has the characteristics of an *autonomic element*, represented by IBM in the form shown in Figure 2 [10]. This identifies a *managed element* (web site) under the control of an *autonomic manager* (link monitor), using *sensors* and *effectors* to understand and adjust both the managed element and its environment.

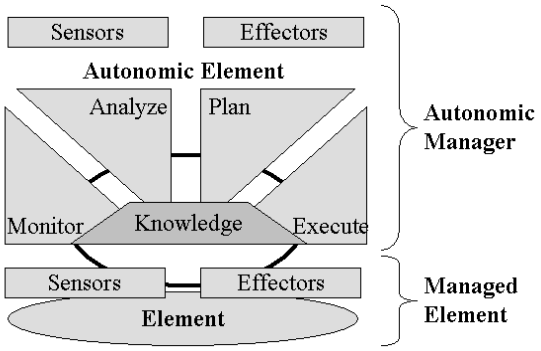

Figure 2 IBM Autonomic Element Structure

The purpose of this paper is to review the issue of broken links on a web site, taking a broad systems view of the nature of the problem, and an autonomic perspective on its possible solution. The next section examines the problem through a partial analysis of its context using Soft Systems Methodology [11-13]. That is followed by a consideration of the general design of a web site as an autonomic element and then a description of some experimental work undertaken to better understand the issues in managing broken links. Opportunities for further work in this area are identified.

2. Analysis of the Broken Links Problem

Superficially, the problem of handling broken links seems straightforward. Web site administrators are motivated to remove broken links because they know that such faults throw doubt on the usability of a web site and, by implication, undermine the perceived quality of any services that it supports. Detecting broken links is also simple as there are a number of link checkers available, at low or no cost [14-16]. Such a tool could be used each time a web site is modified to pick up any errors created, and then run again periodically to check for errors resulting from external changes.

In practice, however, such influences are not sufficiently strong to keep the broken links problem at an acceptable level. The main confounding factors seem to be that:

- There are no quality standards for publishing on the web; as almost anyone can publish almost anything, at any time, subject only to very basic legal constraints, variable quality is probably inevitable.
- Some material published on the web is subsequently abandoned, becoming progressively inaccurate as time goes on. Mechanisms for identifying and then either ignoring or removing such sites may emerge as the web becomes ever more cluttered, but for the moment neglected sites contribute significantly to the relatively high frequency of broken links.
- Web site owners tend not to be aware of broken links on their sites so they, or web administrators, are rarely under pressure to correct them.
- Correcting bad links is not always straightforward. Certainly this is true of the web pages that are generated dynamically and require software or database content to be adjusted. Also, the ownership of web pages is not always clear, so although a web administrator may be able to identify broken links quickly, locating the content provider to define the repair needed is likely to be much more time-consuming.

Overall, therefore, although broken links are a problem in themselves, they are also a symptom of the much wider problem of poor web site management.

To help solve the broken links problem, it is desirable to first analyse the broader issue of effective web site control. One approach is through Soft Systems Methodology (SSM) [11-13], which supports the general investigation of problem situations. SSM is particularly relevant in this case as it encourages the use of process control techniques to monitor and refine system performance [17].

SSM involves the creation of *root definitions* corresponding to particular perspectives on a situation of concern. As an illustration, consider the case of a web site viewed as a means of promoting and facilitating use of the services offered by an organisation. Table 1 shows possible root definition components corresponding to this perspective (*Weltanschauung*).

Table 1: Root Definition for Web-Based Services

Components	Meaning
Customers	Current and prospective service users
Actors	Web site administrator and content providers
Transformation	Implement web access to available services
Weltanschauung	Promote and facilitate the use of the organisation's services
Owner	An 'organisation'
Environment	In the environment in which it is used, the web site should function correctly and have content that is judged accurate and timely

0-7695-2809-0/07 $20.00 © 2007 IEEE

A web administrator and content providers (*actors*) are responsible for implementing web access to such services (*transformation*) for current and prospective customers of the organisation (*customers*), with the constraints that the web site should function correctly and that its content should be accurate and timely (*environment*).

A root definition is usually presented as a single statement combining the six individual components shown in Table 1. In this case it might be:

> *An [organisation] owned system, operated by [a web site administrator and content providers], to [promote and facilitate the use of the organisation's services] by [implementing web access to such services] for [current and prospective service users], taking account of the need for the [web site to function correctly and have content that is judged accurate and timely in the environment in which it is used].*

In SSM, each root definition is expanded into a conceptual model, identifying the activities necessary for the system to meet the purpose specified, and indicating relationships among the activities involved. For example, Figure 3 shows a conceptual model based on the root definition described in Table 1. The activities have been labelled for convenience.

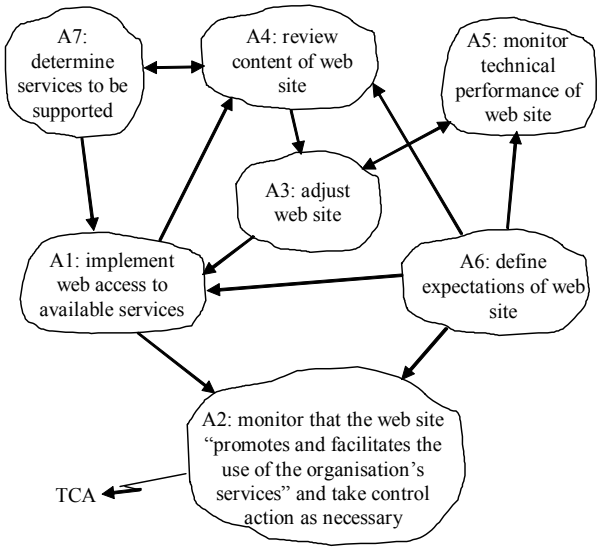

Figure 3: Conceptual Model for Promoting and Facilitating use of an Organisation's Services through a Web Site

The model includes the transformation taken directly from the root definition (A1). This is essentially the central activity of the model. Another important activity following from the root definition is A2, which monitors that the defined Weltanschauung (viewpoint) is achieved, taking control action if necessary (TCA); this can affect any other activity in the model. Activities are also added to handle the environmental constraints identified in the root definition (A3-A5) and to cover consequential or implied activities (A6-A7).

Although the model in Figure 3 is relatively simple, it makes clear in a succinct way why the web site exists and the basic operations required both in establishing it and then ensuring its ongoing relevance, efficiency and effectiveness. In effect, the conceptual model helps put the broken links problem into the wider context of the technical performance of the web site and the even wider context of ensuring that the web site meets its intended purpose of promoting and facilitating the use of the services of the organisation.

The seven activities in the conceptual model can be used as a checklist against which to assess the current operation and management of the web site and so highlight a broad range of opportunities for improvement. Each activity can also be examined to see how it might be supported by information systems or other technology [18].

In relation to the broken links problem, for example, it might be concluded that the activity A5: *monitor technical performance of web site*, is currently too informal and could be improved by a monitoring application that checks all links, either on demand (following a change) or routinely overnight, reporting any that are found faulty. In relation to activity A6: *define expectations of web site*, this might include a goal to minimise broken links across the site, together with a performance target to repair such links within a specified period.

The next section considers how the suggested link monitoring software might be implemented, approaching it from an autonomic perspective.

3. Autonomic Web Link Manager

Conceptually, a fully autonomic web link manager would be able to identify broken links and repair them automatically. So, for example, if a target web page was moved to a different location it would be found and the link adjusted accordingly. There are a number of ways of detecting the missing page [1], based on a search using information stored around the link or in a separate location. One approach is simply to keep a complete copy of each external page and look for a match when a link is broken.

Keeping link information has maintenance overhead, however, as the target page may change from time to time, making it necessary to detect this change and update the associated descriptive information accordingly. In particular, if a page is moved it may be because its content has also changed. If so, then there

are also consequences for replacing the broken link automatically. Essentially, a direct replacement can only be made if there has been no semantic change to the target page. Unfortunately, drawing that conclusion through software analysis alone is very difficult and, indeed, not always possible. The implication therefore, is that the repair of broken links will often have to be done in consultation with web administrators and content providers.

This example illustrates that while fully autonomic behaviour can be achieved in some cases, corrections to broken links will typically require some form of human input. At best, that will be to approve a suggested correction, but may well involve other work. Thus, in general, a *blended* approach to autonomic system design is required, in which the overall goal is to find a suitable balance between what is performed in software and what should be referred 'up' for human consideration.

In turn, this implies that the autonomic system has some model of its *environment*, identifying the humans involved and the circumstances in which they are to be consulted [18]. For example, in the case of the broken links problem, it would be useful to introduce the concept of a web site 'owner'. One responsibility of the autonomic system might then be to produce a daily report on the status of broken links on the site for this person to review. In practice, this step alone is likely to have a significant impact on the broken links problem as it makes the web site owner directly aware of the extent of the issue and any ongoing progress towards bringing it under control.

Other obvious roles in this situation are the *web site administrator* and the *content providers*. The daily report for the site owner would also be copied to the web administrator, who would be responsible for ensuring that broken links are repaired.

Usually, changes will have to be defined by the appropriate content providers. To automate communication with providers, each web page must be associated with a specific person. Thus, following a scan for broken links, separate error reports would be sent electronically to each affected provider. A summary would also be sent to the web site administrator and to the owner to allow them to monitor progress with repairs.

3.1 LinkMan II Experimental Prototype

An experimental prototype system, LinkMan II [14] has been created to gain insights into the potential for autonomic support in handling broken links on a web site and identifying any practical issues in implementing adequate solutions.

As indicated, there are already a large number of link checkers on the market [16], all with the same basic functionality to check a web site as quickly as possible and produce a report on any problems encountered. None of the checkers attempt to help fix broken links, probably because of the many different situations that have to be handled. In some cases, for example, it may simply be a matter of inserting a new address but wider changes around the broken link may also be necessary, or the link deleted. Similarly, the way that the change to the faulty page is implemented will be different for static and dynamic pages.

Surprisingly, most link checkers tend to describe the location of broken links rather than highlight them on the affected pages. Figure 4, for example, shows the type of report produced by the *Fast Link Checker* [14]. Here the position of each error is described using a mixture of page text and the location of the link in the HTML source. The example shows three different types of error: (i) a broken internal link (bookmark not found); (ii) a broken external link; and (iii) a broken link to an image.

Figure 4: Error Report from Fast Link Checker

The approach taken in developing LinkMan II was to evaluate the available checkers, select one and then add features to it. The main requirement was that the link checker could be used as a component of another program. In effect, this meant that it could be invoked from a separate application, with output directed to a file for subsequent analysis, as illustrated in Figure 5.

Four of the more popular link checkers were evaluated [14], with *Fast Link Checker* finally selected for the prototype. It met the basic connection requirements for inclusion in the planned autonomic link checker. As well as being 'fast' (because of its use of multi-threading), it also offers good search control facilities and flexibility in defining the content and format of the broken links report.

The aim of the prototype was to allow a user to work through a broken link report one entry at a time, displaying the broken link highlighted on the faulty page, and allowing the link to be replaced on static pages.

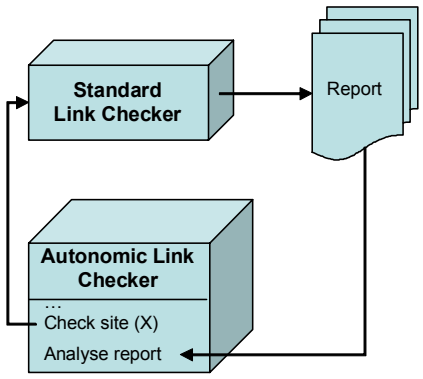

Figure 5: Autonomic Link Checker Structure

The main lessons learned from the prototype were:

1. Displaying a copy of a faulty page with the link highlighted required more processing than expected. Where the link was represented by text, tags were added to make the text 'flash'. For links represented by images, however, a small piece of JavaScript was needed to make the image appear and disappear to achieve the same effect. Also, to display a page after processing, all relative addresses in the HTML code behind the page have to be made absolute.
2. The link checker isn't always able to establish a connection to the host server. This may appear as "socket connection error" when a connection is refused or "domain name lookup fail", when there is a fault in the address of the host, or "403 Forbidden", when the server refuses to handle the request. Each such situation needs to be examined individually to determine the underlying issue; in some cases the error may be transient.
3. Many page addresses are remapped by a server to other locations. This seems to be the best way to 'move' pages in that both the old addresses and new addresses remain valid. The fact that rerouting has occurred is reflected in the server response of "page permanently moved to…" There is potential, therefore, to change such addresses to the new target. This should improve the efficiency of page access though the saving may be relatively small. A more detailed investigation of this opportunity is required.
4. Some broken links are not obvious because a substitute page is returned instead, with an OK code [4]. These cases are currently ignored but would be picked up if the autonomic checker was extended to track page changes, as discussed earlier.

The overall conclusion drawn from building the prototype is that a useful autonomic system can be constructed on top of an existing link checker, with the potential to increase its value by progressively including coverage for the different cases that can occur.

3.2 Other Opportunities for Autonomic Support

The prototype autonomic link checker has focused exclusively on locating broken links and helping to repair them. This sub-section looks more broadly as providing autonomic support for web site construction and maintenance. This is approached through a consideration of the four main autonomic self-managing activities of: *self-configuring*, *self-optimising*, *self-healing* and *self-protecting*, together with the basic autonomic properties of being *self-aware* and *environment-aware* [22].

1. *Self-aware*. The checking of links is one form of self-awareness in that it leads to a measure of the integrity of the web site. Further information can be obtained from the server logs [5, 6] to identify the errors that have actually been encountered and the number of different users affected. The server logs also identify patterns of usage of the web site, highlighting those that are most and least popular. With suitable organisation it may also be possible to determine the level of interest in particular services through reference to specific pages. The web site might also attempt to gather user information explicitly in cases where required information can not be derived automatically.
2. *Environment-aware*. Checking links also contributes to environment-awareness in that some of the links will be to external resources. An autonomic system could produce a map of such dependency, supplemented with usage information extracted from the server log. It has been useful to identify the web site administrator, content provider and owner roles as these help identify the functionality required by the autonomic system. Further work is desirable here to develop business processes that place these roles in context, and define how they interact. This can help in the design of the autonomic system and influence the design of the web site itself.
3. *Self-configuring*. In principle, at the user level, a web site can monitor access by individuals and adjust what is presented accordingly. Some, however, are uncomfortable when such monitoring is made clear or when changes occur unexpectedly in an attempt to be 'helpful'. Further research is needed here to determine what self-configuring is desirable and how it might be implemented in a way that takes user preferences into account.
4. *Self-optimising*. In general, this aspect of autonomic behaviour has been given most attention in relation to web site management [21, 22] because it addresses the most significant problem of controlling demand to avoid slow-loading pages. Work is also needed here to facilitate users in the

efficient location of information, especially as web sites grow larger.

5. *Self-healing*. This paper has considered the most obvious problem of broken links and how they might be repaired. A more significant problem in the longer term is the detection and repair of information that is inaccurate or out-of-date both within a web site and in pages to which it refers. Research is needed into efficient ways of identifying and handling such material.
6. *Self-protecting*. Checking broken links is also a form of self-protection so when changes are made the pages affected should be checked before they are made live. This is consistent with the general principles of configuration management [23]. It is likely that a full history would be kept of all pages on a site so that external references to it would never produce broken links. Similarly, copies would be kept of external pages referenced from the site to aid recovery if a link to any such page or its contents change significantly.

The discussion here is based on a standard context were nothing can be assumed about how web pages are described and managed. If progress were made towards adopting the principles of the semantic web [24], many aspects of web site management would become more straightforward.

4. Conclusions

This paper continues a theme of promoting autonomic computing as a general concept in systems development [24], through a consideration of commonly-occurring problems in broad application areas [18]. Such an approach is intended to illustrate how autonomic ideas can benefit the design of all systems and not just those that are highly complex.

The particular topic covered was the handling of broken links in a web site as part of the general problem of effective web site organisation and control. The paper explored the problem through a partial SSM analysis which identified some of the wider factors contributing to the problem and its possible solution. The design of an autonomic link checker was then considered, taking a blended approach that balanced responsibilities between the autonomic system and the human agents involved.

Some specific issues in handling broken links were explored through the development of a prototype autonomic system, LinkMan II, that finds broken links, highlights their location and, where possible, helps repair them. This revealed some of the practical and technical difficulties in moving towards a solution that involved more extensive autonomous behaviour.

The paper concluded with a discussion of autonomic support for web site construction and maintenance, in general, identifying a number of areas for further research.

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

[1] Davis, H.C., Hypertext Link Integrity, *ACM Computing Surveys*, Vol. 31, No. 4, pp. 1-5, 1999
[2] Ashman, H., Electronic Document Addressing: Dealing with Change, *ACM Computing Surveys*, Vol. 32, No. 3, pp. 201-212, 2000
[3] Wikipedia, http://en.wikipedia.org/wiki/Link_rot
[4] Bar-Youssef, Z., Broder, A.Z., Kumar, R., and Tomkins, A., "Sic Transit Gloria Talae: Towards an Understanding of the Web's Decay", in *Proc 13th International Conference on World Wide Web,* New York, USA, pp. 328-337, 2004
[5] Tian, J., Rudraraju, S. and Li, Z., "Evaluating Web Software Reliability Based on Workload and Failure Data Extracted from Server Logs", *IEEE Transactions on Software Engineering*, Vol. 30, No. 11, pp. 754-769, 2004
[6] Huynh, T. and Miller, J., "Further Investigations into Evaluating Website Reliability", in *Proc. IEEE International Symposium on Empirical Software Engineering*, Queensland, Australia, pp. 162-171, Nov 2005
[7] Mark, J., Brooks, D. "Broken links: The ephemeral nature of educational WWW hyperlinks", *Journal of Science Education and Technology*, 11, 105-108, 2002
[8] Graphical, Visualization & Usability Centre, WWW User Surveys, http://www.gvu.gatech.edu/user_surveys/, Georgia Institute of Technology, 1994-98
[9] Beer, S., The Viable System Model: Its Provenance, Development, Methodology and Pathology, *Journal of the Operational Research Society*, Vol. 35, pp. 7-26, 1984
[10] A.G. Ganek and T. A. Corbi, The dawning of the autonomic computing era, *IBM Systems Journal*, Vol. 42, No. 1, pp. 5-18, 2003
[11] Checkland, P., Systems Thinking, Systems Practice (with 30-year retrospective), John Wiley & Sons, 1999
[12] Stowell, F.A. (ed.), *Information Systems Provision: The Contributions of SSM*, McGraw-Hill, London, 1995.
[13] Mingers J., Taylor S., "The Use of Soft Systems Methodology in Practice," Journal of the Operational Research Society, 43(4), 1992, pp. 321-332
[14] Higgins, D., "To Explore the Potential and Issues in Developing an Autonomic Web Site", MSc Thesis, University of Ulster, Oct 2006
[15] Fast Link Checker, www.fastlinkchecker.com/
[16] Link Checker Review, www.cryer.co.uk/resources/link_checkers.htm

[17] Wilson, B., *Systems: Concepts, Methodologies and Applications*, John Wiley & Sons, 1990

[18] Bustard, D.W., Sterritt R., Taleb-Bendiab, A., Laws, A., Randles, M. and Keenan, F, Towards a Systemic Approach to Autonomic Systems Engineering, Proceedings of *2nd IEEE Workshop on Engineering of Autonomic Systems* (EASe 2005), IEEE Computer Society Press, pp. 465-472, 2005

[19] Bustard, D.W., Hassan, S., McSherry, D.M.G., "Standard Exemplars for Autonomic Computing Concepts", Proceedings of 3rd *IEEE Workshop on Engineering of Autonomic and Autonomous Systems* (EASe 2006), Potsdam, Germany, March 2006, IEEE Computer Society Press, Pages 47-53

[20] Horn, P., "Autonomic Computing: IBM Perspective on the State of Information Technology". IBM T.J. Watson Labs., 2001. Presented at *AGENDA 2001*, Scottsdale, AR (www.research.ibm.com/autonomic/

[21] McCann, J.A. and Jawaheer, G., The Patia Autonomic Webserver: Feasibility Experimentation, Proceedings of *14th International Workshop on Database and Expert Systems Applications*, Prague, Czech Republic, pp. 661-665, 2003

[22] Bouchenak, S., De Palma, N., Hagimont, D., Krakowiak,S. and Taton, C. "Autonomic Management of Internet Services: Experience with Self-Optimization". Proceedings of the *3rd IEEE International Conference on Autonomic Computing* (ICAC'06), Dublin, Ireland, June 2006 (short paper)

[23] Hass, A.M.J., *Configuration Management Principles and Practice*, Addison-Wesley, 2003

[24] Breitman, K., Casanova, M.A., and Truszkowski, W., *Semantic Web: Concepts, Technologies and Applications*, NASA Monographs in Systems and Software Engineering, Springer, 2006

[25] Bustard, D.W., Sterritt, R., "A Requirements Engineering Perspective on Autonomic Systems Development", in Parashar, M., Hariri, S. (eds.), "Autonomic Computing: Concepts, Infrastructure, and Applications", CRC Press, 2006