Autonomic Network Management: Some Pragmatic Considerations

Richard Mortier
Microsoft Research
7, JJ Thomson Ave,
Cambridge CB3 0FB. UK.
+44 (0) 1223 479000
mort@microsoft.com

Emre Kıcıman
Microsoft Research
1, Microsoft Way,
Redmond, WA. USA.
+1 (425)-705-3659
emrek@microsoft.com

ABSTRACT

Autonomic Network Management (ANM) has the goal of increasing reliability and performance while reducing management cost using various automated techniques. These range from agent-based approaches relying on explicit models and ontologies to emergent techniques relying on gossip protocols, swarming algorithms or other biologically inspired work. In this paper, we review the failures, growing pains and successes of earlier techniques for automated and adaptive network control and management, from the simple control loops in TCP and OSPF to the more complicated emergent behaviors of BGP dynamics and overlay routing. From these examples we extract several lessons relevant to ongoing research in autonomic network management.

Categories and Subject Descriptors

C.2.m [**Computer-Communication Networks**]: Miscellaneous.

General Terms

Management, Measurement, Performance, Design, Reliability.

Keywords

Autonomic network management.

1. INTRODUCTION

As networks grow ever larger and more complex, they become harder to manage efficiently and reliably, affecting the dependability of mission critical network applications and services. *Autonomic Network Management (ANM)*, where the network itself helps to detect, diagnose and repair failures, as well as to adapt its configuration and optimize its performance and quality of service, is becoming an increasingly important research area.

ANM encompasses many different approaches, from explicit modeling of network semantics to techniques based on the emergent behavior of biologically-inspired systems. However, current ANM techniques are not the first to use automated responses and adaptation to help simplify the configuration and management of networks. Previous success stories include TCP's simple control loop for adjusting its congestion window, automated media negotiation in the Ethernet (802.3) and Ethernet (802.11) wireless protocols, and failure detection and adaptation techniques in link-state routing protocols such as OSPF. Each of these automated techniques is fundamental to the correct operation of today's networks—without them the large-scale networks we use today simply could not exist.

Though now successful, each of these automated techniques experienced "growing pains," where unanticipated situations and emergent (mis-)behaviors caused widespread problems, hindering the reliability of the network they were meant to help. For example, TCP has required many adjustments over the years, adding basic congestion control to avoid congestion collapse of the Internet and updates to improve performance over high speed and high-bandwidth delay paths. Even today, TCP's adaptation algorithms have been observed to behave poorly on wireless and satellite links, leading to many proposals for improvements.

In this paper, we take the position that there is much to be learnt from these historical successes and failures. To support this position, we present a few guidelines for the design of ANM system that we have drawn from these historical examples. Note that there are several things we are not doing: we do not present an exhaustive (or even representative) description of the many useful software engineering or control theoretical techniques that can be applied to automated system design; we do not present a framework by which ANM systems may be evaluated or compared; and we do not present a design for any particular ANM system.

Instead, we review several early experiences with automated network behaviors—in some sense, the precursors to the more advanced automation that is the goal of current ANM research—from the simple control loops in TCP and OSPF to the more complicated automated behaviors of BGP dynamics and overlay routing. We describe both the successes and failures of these systems in order to draw lessons and guidelines for current research in ANM:

Make assumptions explicit. Automated control systems necessarily make assumptions about the environment in which they operate and the expected results of automated actions. By making these assumptions explicit, it becomes easier to dynamically tell when a control system is operating under supported conditions.

Handle common failures. Once we understand our automated system's assumptions, we should check that they cover the common failures encountered in real-world environments.

Avoid (or understand) interactions between control loops. Interacting control loops often behave in unpredictable, complex ways which should be understood, and whose impact should be controlled.

Make goals explicit. If the problem being solved is unstated, it is unlikely that the solution will be either coherent or comprehensive.

Help people help the system. Most problems with previous automated systems required extensive and often challenging measurement and experimentation to resolve. To ease the understanding of future problems, extensive support for monitoring of system performance, reliability, and other behaviors should be designed in from the start.

Build-in validation. The system should self-test to ensure it is meeting its goals.

Some of these may appear to be fundamental software engineering guidelines—and indeed, they are! However, as one of our reviews commented, "*The fact that these guidelines are often not followed today says probably more about the state of the art in software and systems engineering than about our inability to engineer automated systems*". As the ANM community moves forward, it is critical that these guidelines (by no means an exhaustive list) be borne in mind. We hope that this paper will stimulate discussion around such matters.

2. BACKGROUND

While approaches to autonomic network management are many and varied, ranging from model-based agent systems to epidemic gossip protocols, all broadly rely on some form of feedback or control loop where a network or system adapts its behavior based on observations of its current state, performance and reliability. In this section, we briefly note some simple control loops which might be considered precursors to ANM. We later use some of these to provide examples of pitfalls and successes.

2.1 TCP Congestion Window

Perhaps the best known in terms of controlling performance is the the Transmission Control Protocol (TCP) [1][5] which uses a simple retransmission scheme based on maintaining a window of data in flight with acknowledgements to handle lost data. Coupled with an additive increase, multiplicative decrease control loop to adjust its congestion window, this ensures forward progress while trying to share network resources fairly between network flows.

2.2 Routing

In the world of IP routing, link-state routing protocols such as OSPF [5] provide autonomic functionality: after link weights have been configured (typically a manual process), the routing protocol will detect link failure through a variety of methods, such as repeated failures to receive packet acknowledgements. Following failure detection it will reroute, eventually converging on a new valid path.

Distance- or path-vector protocols such as BGP provide similar functionality between networks by describing costs to possible destinations. Routers then exercise policies (such as "select cheapest") to choose their preferred path.

2.3 Overlay Networks

Due to the perceived inflexibility of the IP Internet architecture, overlay networks and particularly *structured overlays* have received a great deal of attention and limited successful deployment in recent years [1]. These build forwarding structures between sets of end-systems, enabling experimentation with and deployment of many novel distributed systems without requiring wholesale replacement of the underlying IP infrastructure. Typically, nodes in the overlay maintain some small set of neighbors scattered through the Internet and use these neighbors to forward packets to destinations within the overlay. Neighbor sets and forwarding decisions are often made based on observed performance characteristics between nodes in the overlay. For example, in the Pastry overlay system, the default behavior is for nodes to use observed round-trip-time to their neighbors to select best-hops [16].

2.4 Other

There are a wide variety of other automatic network control mechanisms which taken together might be considered as first steps toward an autonomic network infrastructure. For example, the Ethernet (802.3) and wireless Ethernet (802.11) protocols use a variety of techniques such as media negotiation to achieve almost totally automatic configuration and operation.

3. LESSONS FROM FAILURE

In this section, we extract lessons from the "growing pains" suffered by otherwise successful automated network behaviors. These can be summarized as: understand the operating region of the algorithm and ensure that it holds during normal operation and common failures, and be careful to avoid interacting control loops that can cause unanticipated emergent behavior.

3.1 Know the Operating Region

Automating network management and behavior (*i.e.*, building a system that takes automatic action in response to observed inputs with the aim of achieving some system-wide property) necessarily involves making assumptions about both the meaning or underlying cause of observations, and how a given action will mitigate or improve the situation. When these assumptions hold, the actions are appropriate and will have the desired effect. However, if the assumptions are violated—for example, if the system is operating in a new and untested environment or an unanticipated situation arises in an existing environment—then the automatic response may in fact further damage the system.

For example, TCP famously assumes that packet losses are a sign of congestion and that reducing a connection's bandwidth usage (*i.e.,* reducing its window of unacknowledged data) is therefore always the appropriate response to packet loss. While this assumption generally holds in wired networks, TCP has been observed to have poor performance over wireless networks of various types [1][2][17]. The usual explanation is that there are other common causes of packet loss than congestion (*e.g.*, radio interference), breaking TCP's assumption about the cause of packet loss, and making its response (to reduce bandwidth usage) inappropriate.

This is not the only case where changes in TCP's environment have warranted modifications to its behavior over the years. Congestion control was only added in 1988 as the Internet grew beyond its initial design parameters. Further updates were applied in 1992 [7] to deal with high speed and high bandwidth-delay product paths enabled by new communications networks.

The lesson: *Automated systems necessarily make assumptions about their operating environment, but changes in the environment often invalidate these assumptions and require tweaks or wholesale modifications to their behavior.*

It is thus clearly important to understand the assumptions made by an automated system and ensure that these assumptions hold to prevent inappropriate actions from harming the dependability of a network. Control theory formalizes this concept as the *valid operating region* of a feedback loop, often specified as the range of control inputs where the feedback loop is known to work well [5]. In safety-critical cases, a controller which observes that the current system behavior is outside its valid operating region can notify an administrator or take other fail-safe actions.

3.2 Handle Common-Case Failures

A corollary to understanding the operating region of an automated system (and adapting it as the underlying environment evolves) is ensuring that the operating region encompasses likely failure scenarios, and does not exacerbate the occurring problems.

For example, when a network link between Autonomous Systems (ASes) fails, the Internet's Border Gateway Protocol (BGP) responds by removing routes that rely on that link, selecting new routes as required, and announcing any resulting changes to its neighbors. Although processing these updates can be an expensive operation, routers are built to handle the load that results from occasional link failures. However, a common-case failure scenario is that a network link will repeatedly fail and recover over a relatively short period of time, overloading BGP routers with continual route updates.

To deal with this situation, and prevent collapse of the routing infrastructure due to transiently failing links, router vendors developed *ad hoc* heuristics to damp the advertisement of such route flaps, leading to the development and deployment of a standard approach to *BGP route flap damping* [17].

Common failure modes have caused similar problems in other systems as well. For example, early versions of the OSPF routing protocol [5] could not establish adjacency connections over high latency links. Once two routers have formed an adjacency, they exchange their link state databases so that each can learn the current topology of the network efficiently, without having to wait to independently receive link state advertisements from all other routers. However, if the initial database description packet was lost or delayed past a timeout, early versions of the protocol [13] chose to reset the adjacency connection.

Link latencies and reliabilities were sufficiently low in initial deployments that this did not matter: the initial packet was rarely lost, the database synchronization proceeded correctly, and the protocol worked reasonably well. However, in the early nineties satellite links became common. Such links have significantly higher latencies than traditional land-based technologies. In particular, they could easily have latencies higher than the timeout for detecting lost packets. This would cause the initial database description packet to be delayed to an extent that it was perceived as lost, resulting in the OSPF process resetting the adjacency and preventing OSPF from ever forming a working adjacency across such a link. Subsequent versions fixed this problem by permitting retransmission of the database description packets.

The lesson: *This example illustrates an important corollary of the previous lesson. It is not enough just to understand the operating region and steady state of the system being controlled. It is also important to understand likely failure scenarios, and ensure that the automated behaviors of the control system are equally appropriate in these situations.*

3.3 Beware Interacting Control Loops

Recent studies have shown problems can arise when running overlay networks over IP infrastructure networks. Work by Keralapura *et al* has shown that overlays can interact badly with the underlying IP network and, where multiple overlays operate side-by-side, with each other [8][9]. Essentially, routing based on the mechanism of measured performance couples the routing systems, whether overlay with infrastructure, or overlay with overlay.

Although performance based routing is a well studied area and protocols have been deployed that worked well [10], this kind of *accidental* coupling can have wide-ranging effects, from the merely dangerous, to the malicious and potentially disastrous. Here we describe two instances of unanticipated behaviors arising from interactions between different components of the network and the global BGP routing infrastructure.

The first case arose with the addition of *route flap damping* to BGP [17]. As already mentioned, this is a mechanism intended to protect the routing infrastructure against a frequently changing route. If flap damping is enabled then a route that changes status too frequently will be ignored until it remains stable for a period of time based on how unstable it has been. Unfortunately, it has subsequently been shown that interaction between flap damping and the route convergence process can itself *cause* persistent oscillation in a network that would converge in the absence of flap damping [12].

The second example concerns transport layer attacks against the routing infrastructure. It has been demonstrated that relatively low bandwidth streams having the correct burst pattern can interfere with a selected existing TCP flow in the network [8]. If applied against infrastructure protocols such as BGP which rely on TCP, these techniques may have a significant negative impact on the network.

The lesson: *Even if the control loop is made explicit and the operating region for a given control loop is well-defined, interactions between control loops can result in complex and difficult to understand behaviors. As more components that can control the behavior of the system are added, it becomes progressively more difficult to understand how they will interact, and the affects of moving outside the operating regions of different parts.*

While interaction between control loops cannot be avoided in general, and can be very desirable in certain situations, there are techniques that can help minimize the negative impacts. For example, designers can try to decouple control systems by ensuring that they control different independent outputs based on independent inputs. If this is not possible, then tuning them so that they impose control at very different timescales can help to decouple systems that would otherwise couple. These are very well studied problems in other industries—the key message is to avoid these interactions happening by accident!

4. LEARNING FROM SUCCESS

In this section, we describe some of the lessons we can learn from the eventual successes of current and past automated techniques for network management.

4.1 Be Explicit

To date, successful automated techniques have tackled a well-defined explicit problem: "provide reliable transport over an unreliable network", "ensure transport sessions make forward progress while guaranteeing that the network does not enter a regime of congestive collapse", "provide automatic recovery of network routes in the face of link failure", and so on. Protocols such as TCP and OSPF are observed to be generally successful in achieving their goals.

However, even these protocols have run into difficulty when they have been deployed in regimes which do not satisfy assumptions made by their designs. Furthermore, protocols such as BGP which do not have an explicit statement of their goals are less successful: BGP aims in general to provide automatic policy routing including recovery from link failure in the global inter-provider routing system. However, it is not clear what its detailed goals are: in particular, if presented with a network topology and a set of policies, it can be very difficult to compute how BGP will behave in the face of a link failure.

The lesson: *Without an explicit goal, any (automated) system is likely to wallow in generalities. Since success cannot be defined, the inevitable result for such systems is enormous and ever-increasing complexity. Although they may "work," they rarely work in an efficient and understandable manner.*

4.2 Support Human Understanding

In order to support human intervention when things do go wrong, it is necessary to extract data from the network concerning its behavior, in terms of performance, reliability, and resilience (among other features). By supporting extensive, preferably *pervasive*, monitoring features, network elements make their behavior more transparent to operators and other network elements and control software alike. This allows any network management system, autonomic or not, to be better informed about system behavior and performance and thus to make better decisions when required.

It is unavoidable that humans *will* be kept in the loop to some extent. As environments change and unanticipated situations arise, systems must allow for human intervention to deal with unforeseen problems. While individual ANM solutions may remove humans from *that part* of the system, human intervention will still be required from time-to-time.

The lesson: *Large-scale networking involves many factors other than technology, and these factors simply cannot be addressed by the application of technology. As a result, people* will *be involved in running large networks for the foreseeable future, and so such networks had best be made comprehensible to the people that must run them. A precursor to this is the provision of correct and timely measurements of the network.*

4.3 Monitor and Validate

The process of discovering the failures described in Section 3 was both arduous and *ad hoc*, requiring many people to spend a significant amount of time examining documentation, constructing measurement harnesses, building and exercising simulations and testbeds, etc. Only once all this data and experience had been gathered could the problems be determined and solutions constructed.

It would be preferable if the network could be made *self-validating.* There are at least two precursors to this: explicit statement of goals as noted in Section 4.1; and extensive, preferably pervasive monitoring capability. A common barrier—perhaps more psychological than technical—to deployment of automated network management systems is fear that they will misbehave under as-yet-unforeseen circumstances. If data concerning the state and behavior of the network was made available via suitable programming interfaces, it should be possible to build self-validating components.

The lesson: *Given the complexity of modern large-scale networks, pre-supposing the existence of correct and timely measurement data, the burden on the human operator could be reduced by making the network self-validating. If network services can test themselves for correct operation, human operators' time can be better spent designing the network and planning for the future, rather than continuously monitoring the behavior of particular services.*

5. CONCLUSION

In this paper we have discussed what we believe to be some important issues surrounding autonomic network management. In particular we have argued that there are some significant pitfalls that must be avoided, and have suggested some guidelines

guidelines that, if followed, we believe will help.

In essence we believe that the growing pains of previous applications of automation to network control and management can provide guidance for the success of autonomic network management techniques. Simple automated techniques are at the core of today's large-scale networks, but have achieved such success only after resolving problems due to poorly understood operating assumptions, unanticipated failures, and unpredictable emergent behavior from interactions between different control systems. In general, these problems were only overcome after extensive monitoring, analysis and experimentation and human intervention.

To avoid repeating such problems, we advocate that new autonomic techniques should be explicit about the assumptions they make about their operating environment, and that these assumptions should explicitly include common-case failures. Furthermore, autonomic techniques should strive to avoid outright the unanticipated and unpredictable emergent behaviors (and often misbehaviors) caused by interacting control loops.

Recognizing that operating environments change over time and that some unanticipated situations inevitably arise over time, we claim that successful autonomic network management techniques should incorporate extensive real-time and historical monitoring, both for improved understanding by human operators as well as for the system to perform self-validation of its assumptions and operation.

By learning the lessons of earlier failures, as well as taking advantage of more rigorous monitoring and validation to more rapidly uncover and respond to future problems, we hope that autonomic network management techniques will help large-scale networks become more reliable and manageable.

6. REFERENCES

[1] M. Allman, D. Glover, L. Sanchez, "*Enhancing TCP Over Satellite Channels using Standard Mechanisms*", RFC 2488, IETF. January 1999.

[2] H. Balakrishnan, V.N. Padmanabhan, S. Seshan, R. Katz, "*A Comparison of Mechanisms for Improving TCP Performance Over Wireless Links*", Proceedings of ACM SIGCOMM. Stanford, CA, USA. August 1996.

[3] M. Castro, M. Costa, A. Rowstron, "*Debunking some myths about structure and unstructured overlays*", Proceedings of ACM/Usenix NSDI. Boston, MA, USA. May 2005.

[4] R. Chakravorty, S. Banerjee, P. Rodriguez, J. Chesterfield, I. Pratt, "*Performance Optimizations for Wireless Wide-Area Networks: Comparative Study and Experimental Evaluation*", Proceedings of ACM MOBICOM. Philadelphia, PA, USA. Sept 2004.

[5] J. L. Hellerstein, Y. Diao, S.Parekh, D. Tilbury, "*Feedback Control of Computing Systems*", Wiley Interscience, 2004.

[6] V. Jacobson, "*Congestion Avoidance and Control*", Proceedings of ACM SIGCOMM. pp. 314—329. Stanford, CA, USA. August 1988.

[7] V. Jacobson, R. Braden, D. Borman, "*TCP Extensions for High Performance*", RFC 1323, IETF. May 1992

[8] R. Keralapura et al, "*Can ISPs take the heat from Overlay Networks?*", Proceedings of ACM HOTNETS. San Diego, CA, USA. November 2004.

[9] R. Keralapura, C.-N. Chuah, N. Taft and G. Iannaccone, "*Can coexisting overlays inadvertently step on each other?*", Proceedings of IEEE ICNP. Boston, MA, USA. November 2005.

[10] A. Khanna, J. Zinky, "*A Revised ARPANET Routing Metric*", Proceedings of ACM SIGCOMM. Austin, TX, USA. September 1989.

[11] A. Kuzmanovic, E.W. Knightly, "*Low Rate TCP Targeted Denial of Service Attacks (The Shrew vs. the Mice and Elephants)*", Proceedings of ACM SIGCOMM. Karlsruhe, Germany. August 2003.

[12] Z. M. Mao, R. Govindan, G. Varghese, and R. Katz, "*Route Flap Damping Exacerbates Internet Routing Convergence*", Proceedings of ACM SIGCOMM. Pittsburgh, PA, USA. August 2002

[13] J. Moy, "*OSPF Version 2*", RFC 1583, IETF. March 1994.

[14] J. Moy, "*OSPF Version 2*", RFC 2328, IETF. April 1998.

[15] J. Postel (ed.), "*Transmission Control Protocol*", RFC 793, IETF. September 1981.

[16] A. Rowstron and P. Druschel, "*Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems*", IFIP/ACM International Conference on Distributed Systems Platforms (Middleware). pp 329—350. Heidelberg, Germany. November, 2001

[17] C. Villamizar, R. Chandra, R. Govindan, "*BGP Route Flap Damping*", RFC 2439, IETF. November 1998.

[18] Y. Zhang, Z.M. Mao, J. Wang, "*Impact of low rate TCP-targeted DoS attacks on BGP*", AT&T Labs Research Technical Memorandum, TD-6LLPQY (version 2). May 2006.