Autonomic Closed Control Loops for Management, an idea whose time has come?

Liam Fallon, John Keeney, and Ram Krishna Verma

Ericsson Software Technology, Athlone, Co. Westmeath, Ireland
Email: Liam.Fallon, John.Keeney, Ram.Krishna.Verma (at) est.tech

***Abstract*—Taking an autonomic approach to management and using closed control loops has been the subject of much research in the Network management community since the early 2000s. It is fair to say that Network Management system developers and users have not adopted Autonomic Management approaches very widely. Most network management systems continue to use an ITU TMN inspired layered approach to management.**

In recent years, a trend towards implementing autonomic management and closed control loops on management systems built using a TMN architecture has emerged in practice. This trend is requirement driven; an autonomic approach is taken when there is no other option for implementing a feature. It is clear to see a closed control loop approach being taken to implement C-SON (Centralized Self Organizing Networks) features in 4G network management systems in the early 2010s. Autonomic approaches are even more apparent in systems such as ONAP that implement SDN and NFV orchestration. However, the implementation of closed control loops is often pragmatic and rigid, focused on the feature being delivered. Providing systemized support for control loops is in its infancy and has much to learn from the extensive autonomic management literature

This paper surveys the current state of autonomic management in practice and outlines some research challenges that must be addressed to allow it to be systematically supported in current management systems, with a particular focus on ONAP.

***Index Terms*—Autonomic Management, Control Loops, Virtualization, NFV, VNF, ONAP, 5G**

I. Introduction

The development of conventional management systems and autonomic management over time is illustrated in Fig.1.

The study and application of control loops is a very old discipline, going back to the age of steam. Control loops have been successfully applied in maritime, aeronautical, automotive, and manufacturing systems for centuries. They have also been successfully applied in microprocessors and embedded systems, as well as in operating systems and other low level software components such as device drivers. Even though such systems are often complex, they are deterministic and well bounded and are very suitable for the application of control theory. Telecommunication systems are poorly bounded stochastic systems and it has proven to be very difficult to apply control theory to manage telecommunication systems.

Today's management architectures and systems emerged using protocols and architectures defined in a flurry of standardization carried out in the 1980s [1] and 1990s [2]. Management architectures are layered in hierarchies with NEs (Network Elements) and their management at the bottom and network application management systems at the top. Management systems are usually also layered, with data flowing from NEs via a mediation layer at the bottom up to applications on the top and configuration flowing in the opposite direction.

Autonomic management was proposed in the early 2000s as a new management paradigm. Its proponents proposed revising the architecture of network management into a series of cascaded control loops. Much research was undertaken on this topic in the 2000s. However, this shift did not occur due to factors such as the technical complexity of implementation, the difficulty of migrating existing systems, and inertia.

In the 2010s, a number of applications have emerged that exhibit emergent autonomic characteristics such as C-SON and orchestration of SDN/NFV. These applications have been implemented on conventional management systems using pragmatic approaches. Such pragmatic approaches are not generalizable, but point to a future where a generic autonomic framework can run on a conventional management system to enable autonomic closed control loops.

This paper is structured as follows. §II outlines the structure of conventional management systems. §III describes autonomic management. §IV places autonomic management in the context of conventional management systems and §V plots how to enable autonomic loops on those systems. §VI concludes the paper.

II. TMN Inspired Layered Management Systems

Most network management systems have an architecture inspired by the ITU Telecommunications Management Network (TMN) hierarchical layered approach [2] [3] [4], often drawn as a 5-layer pyramid. Systems at each layer manage systems at the layer below. NEs are managed by element managers using protocols such as SNMP [5]. NEs are at the lowest layer and management cascades up to application management at the top layer. Management systems structured in this manner have been the backbone of network management for decades.

Management systems at each TMN level often have a layered internal architecture such as in Fig.3. System support provides functions such as starting, stopping, and monitoring applications and common components provide functions such as data storage and message passing. Network mediation towards NEs is usually implemented using a plug-in approach with standard plugins supplied for standardized interfaces. Additional plugins can be added to support non standard and vendor specific interfaces. Common applications such as Fault Management and custom applications can be deployed.

978-3-903176-24-9

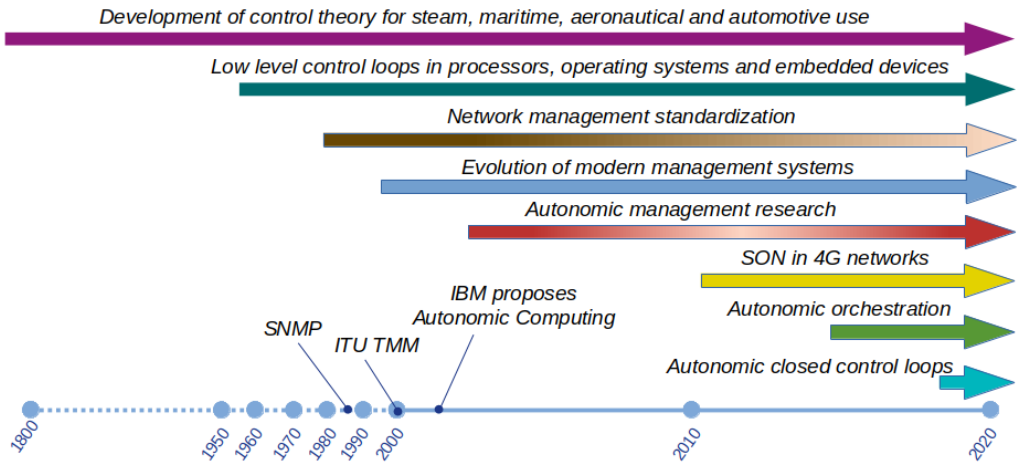

Fig. 1. Timeline of Network Management and Control Loop Research and Development

Customizable northbound mediation and a web-based user interface is also usually provided. In modern systems, components are often containerized and virtualized and are deployed using a container orchestration system such as Kubernetes[1].

Thirty years after its definition, the TMN approach is almost exclusively used to design, implement, and deploy cascaded

[1]kubernetes.io, the Kubernetes container orchestration system

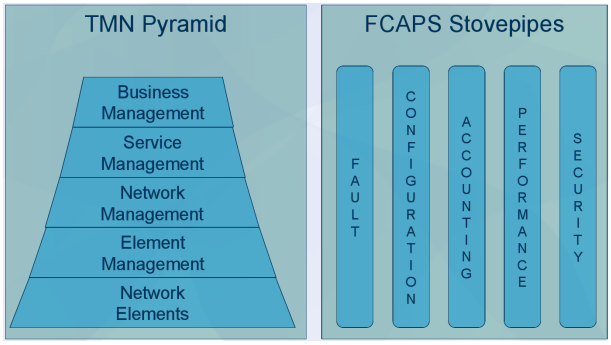

Fig. 2. The TMN Layered Pyramid and Functional Silos

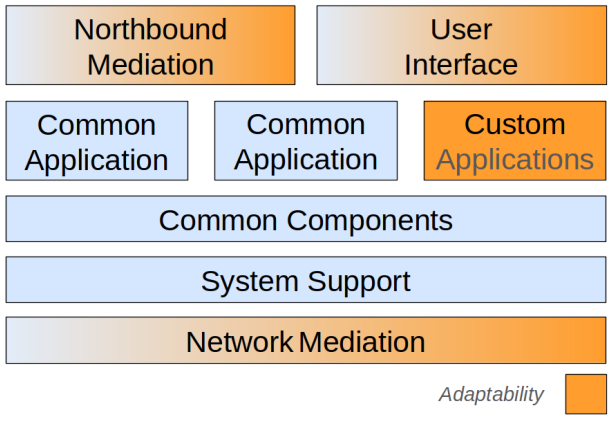

Fig. 3. A typical layered Network Management System Architecture

management systems because the approach provides an easily understood framework into which management system developers and implementers can position their systems. It is relatively easy to deploy a management system to do FCAPS type management (Fig.2): getting a TMN system to do alarm presentation, monitoring data collection, and command-based configuration is relatively straightforward. TMN makes the typically valid assumption that network elements have limited management capability and that most management decisions are made in the upper layers of the TMN pyramid. Increases in processing power, memory, storage, and bandwidth available to network management systems, and the use of containerization and virtualization has allowed centralised TMN approach to management to scale to manage large modern networks.

The TMN approach has drawbacks when it is used to implement features that require the management system to react to collected information and apply changes back to the network based on that data. Data collected from the network is well structured but traditionally such data is stored in a data repository and used to generate reports. The centralised nature of the TMN structure, and the compartmentalization of functions inhibits knowledge flow and sharing across features: all information must always flow up and down the hierarchy. Configuration is often manual and labour intensive, carried out using scripts or command files generated by external planning systems. Integration of systems from many vendors, with various network and implementation technologies is difficult: system integration of management systems has become an industry in its own right.

III. Autonomic Management

In the early 2000s, IBM proposed *Autonomic Computing* as an approach to manage the complexity of emergent computer systems. They presented a biologically inspired Vision for Autonomic Computing [6]. Autonomic computing promises an approach where an Autonomic Element (AE) manages itself locally using high level goals received from administrators, from other AEs that have authority over it, and from peer ele-

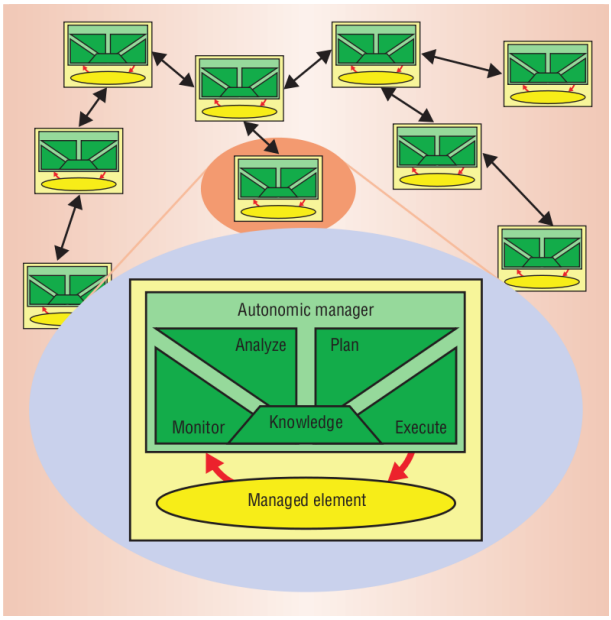

Fig. 4. Autonomic Elements (from [6])

ments with which it is cooperating. An AE is self-configuring, self-optimising, self-healing, and self-protecting.

The Vision of Autonomic Computing [6] laid out in broad strokes how an autonomic system would run. An autonomic system is a network of autonomic elements as shown in Fig. 4. The autonomic system manages itself. The operator provides the system with a set of high level goals. Those goals may be expressed using utility functions [7], or as a set of policies or rules. The autonomic elements cooperate with each other in order to comply with those high level goals. The system self-configures, self-optimizes, self-heals, and self-protects, the so called *self-** [8] properties of autonomic computing.

A system is made up of a collection of AEs, with each AE implementing an autonomic loop that is executed continuously. Each autonomic element implements an autonomic loop in which four phases, *Monitor*, *Analyse*, *Plan*, and *Execute*, are executed iteratively. The loop is usually referred to simply as the *MAPE* loop. Shared knowledge is central to the autonomic approach, with all the MAPE features sharing that knowledge.

The *Monitor* function keeps track of the state of the feature it is managing, checks the status of other autonomic elements with which it has a relationship, and checks for environmental changes that might influence the operation of the autonomic entity. The function records the changes it observes to the knowledge base. The *Analyse* function assesses the knowledge base as it changes, and determines if the autonomic element is operating within the goals that have been set for the system. The *Plan* function uses the results preformed by the *Analyse* function and uses those results together with the knowledge base to decide which if any changes should be performed on the managed system. The *Execute* function takes those changes and applies them to the network. This loop of functions is applied continuously to the managed system. The *Knowledge* repository is central to the autonomic element. Each of the four MAPE functions use the repository to store, retrieve, and exchange data.

In the 2000s, Autonomic Management received much attention in the network management research community [9], [10]. It represented a departure from the accepted approach to network management. Autonomic management provided a framework where researchers in areas such as intelligent agents [11], policy based management systems [12], the use of peer to peer techniques in network management [13], machine learning [14], and data modelling [15] could work together to provide modern unattended management. Autonomic management was a topic of interest in a number of research projects such as the FP6 ANA and Ambient Networks projects, and the FP7 4Ward and EFIPSANS projects.

Research interest in autonomic management waned in the 2010s. It became clear that a paradigm shift to autonomic management architectures was not going to happen for reasons explained in §IV. Researchers increasingly struggled to find applications for their work and they moved to look at other fields.

IV. Autonomicity in Current Systems

This section discusses why a paradigm shift to autonomic network management has not occurred. It goes on to describe some applications developed on current network management systems that exhibit emergent autonomic characteristics.

A. Rejection of Autonomic Management as a Paradigm

To date, autonomic management has not been accepted by network management developers or users as a system architecture, and deployed management systems continue to use TMN management principles. There are a number of reasons why it has not been more widely accepted commercially.

- Technical complexity. Compared to other systems where control loops have been applied (well bounded and deterministic), communication networks are complex poorly bound stochastic systems. Applying autonomic closed loop techniques to such systems is extremely complex.
- Inertia: It is expensive and technically risky to change the system architecture of large cooperating systems. Increases in processing power, memory, storage, and bandwidth allowed centralised management to scale, diffusing the need to embrace new approaches to management.
- Migration. An autonomic approach must work in parallel with existing systems for some considerable time. It is impossible to replace an entire existing management systems at once. A more realistic approach is to develop and deploy an autonomic management approach in phases, perhaps introducing autonomic management for new network features first. The existing management system could then be replaced on a phased basis, perhaps over an extended period of time. Network operators must learn to trust the autonomic system.
- Management interfaces to Network Elements. Any assumption of an autonomic world, where all NEs are autonomic elements is not valid. While the capacity,

resource budgets, and throughput of NEs is now many times what it was, the management features and interfaces on those NES are resource constrained. NE manufactures prefer to focus their development resources on traffic functions rather than on improving management.
- Piecemeal research. Research in autonomic management was fragmented, with activities related to autonomic management such as knowledge engineering, policy based management, distributed agents, and machine learning largely executed in isolation. Research into how efforts in each of these disciplines could come together in a systematic way in order to provide an overall autonomic approach was lacking.

B. Applications with Emergent Autonomic Characteristics

Autonomic approaches have been used to implement some applications on current network management systems. This has typically occurred when there are tight automation requirements on the application and there is no alternative but to design the applications in an autonomic manner. Such applications generally use *Monitoring* and *Execution* support provided by the management system and custom *Analyse* and *Plan* functions to complete the loop for their applications.

1) Centralized SON: C-SON functions execute automatically in the management system [16]. Typical C-SON functions include management of handover relations between radio networks with different technologies and configuring interworking between network equipment from different vendors. C-SON features are implemented using whatever approach the organization developing the C-SON function decides to use.

2) SDN and NFV Orchestration: The advent of virtualization, specifically SDN (Software Defined Networks) and NFV (Network Function Virtualization), allows communication systems and service to be defined in software using metadata. Orchestration procedures such as onboarding and scaling of services in systems such as The Linux Foundation ONAP project [17] use autonomic approaches because the configuration of SDNs and NFVs is metadata driven and well modelled. However, the autonomic approach is implemented in the orchestrator and cannot easily be used to develop control loops not related to orchestration.

3) Control Loops for Services deployed as NFVs: ONAP supports control loops that follow the MAPE approach [18] for non-orchestration applications. Standard ONAP data collection and controllers are used for the *Monitor* and *Execute* phases respectively. *Analysis* is performed by the ONAP DCAE (Data Collection and Analytics Engine) component and *Plan* is performed by the ONAP Policy Framework. However, the implementation is rather rigid and is not truly metadata driven.

V. Enabling Autonomics in Management Systems

While autonomic management research has stalled, there is now strong evidence that applications exhibiting autonomic behaviour can be implemented on traditional management systems and that these applications perform very well (see §IV-B). The challenges identified in [6] and [19] have largely been addressed in current management systems. Engineers have taken a pragmatic approach and have used autonomic principles to develop applications on current systems. For example, the AT&T ECOMP [20] and Ericsson COMPA [21] reference architectures both include an autonomous approach for closed control loops.

The *Monitor* and *Execute* functions of the loop are perhaps the easiest to address because network monitoring and execution of changes on networks is very well understood. The *Analyse* and *Plan* functions has been more technically challenging to address, but advances in fields such as machine learning, artificial intelligence and policy based management mean that implementations of those functions also exist. Data analysis is now a mature field and many well established frameworks exist, among them the DCAE component of ONAP [17]. Recent work on the ONAP Policy Framework [22] provides a robust implementation of the MAPE *Plan* function. Such functions are now core common function in network amangement systems, used in a variety of usecases across layers.

Challenges remain, not least formalizing autonomics on current management systems, which are being addressed by the ONAP Control Loop Subcommittee[2]:

- Control Loop Modelling. There must be a common way of describing control loops in a modelling language such as TOSCA [23]. This model must define the triggers, chain of entities, and actions of the control loop, as well as aggregating the metadata for each component in the control loop chain.
- Reference Implementation. A reference implementation for executing control loops as part of a management system must be built. This reference implementation must consume the control loop model and execute the control loop in response to its triggers. The reference implementation must include support for management and monitoring of control loops, extending the current features of the ONAP CLAMP component. Having a reference implementation will remove the need for bespoke control loop implementations.
- Control Loop Design. A design environment for building control loops is required. ONAP already has some support for control loop design in SDC, but this support must be extended to support modelling of arbitrary control loops.

VI. Conclusion

While Autonomic Network Management has not displaced conventional management architecture paradigms, there are an increasing number of applications and use cases that can only be addressed by taking an autonomic approach. Although research efforts on autonomous systems has waned in recent years, a small but significant number of applications have been deployed using autonomous approaches on otherwise conventional management systems using pragmatic approaches. However a number of challenges remain to make control loop development and deployment more generic. These challenges are being addressed by the ONAP Control Loop subcommittee.

[2] wiki.onap.org/display/DW/Control+Loop+Subcommittee

REFERENCES

[1] IETF, "A simple network management protocol," RFC 1067, IETF, Tech. Rep. RFC 1067, Dec. 1988.

[2] ITU-T, "Overview of tmn recommendations," ITU-T, Tech. Rep. M.3000, Feb. 2000.

[3] ——, "Principles for a telecommunications management network," ITU-T, Tech. Rep. M.3010, Feb. 2000.

[4] ——, "Generic network information model," ITU-T, Tech. Rep. M.3100, Apr 2005.

[5] IETF, "An architecture for describing simple network management protocol (snmp) management frameworks," RFC 3411, IETF, Tech. Rep. RFC 3411, Dec. 2002.

[6] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *IEEE Computer Magazine*, January 2003.

[7] W. E. Walsh *et al.*, "Utility functions in autonomic systems," in *Proceedings of the International Conference on Autonomic Computing (ICAC'04)*. IEEE, 2004.

[8] O. Babaoglu *et al.*, *Self-Star Properties in Complex Information Systems: Conceptual and Practical Foundations*, ser. Lecture Notes in Computer Science. Springer, 2005, vol. 3460.

[9] N. Agoulmine *et al.*, "Challenges for autonomic network management," in *Proceedings of 1st IEEE International Workshop on Modelling Autonomic Communications Environments (MACE)*. IEEE, 2006.

[10] R. Mortier and E. Kiciman, "Autonomic network management: Some pragmatic considerations," in *INM '06: Proceedings of the 2006 SIGCOMM workshop on Internet network management*. New York, NY, USA: ACM, 2006, pp. 89–93.

[11] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.

[12] J. Strassner, *Policy-based Network Management: Solutions for the Next Generation*. Morgan Kaufmann, 2004.

[13] J. Mischke, B. Stiller, and T. Series, "Peer-to-peer overlay network management through agile," in *Integrated network management VIII: managing it all: IFIP/IEEE Eighth International Symposium on Integrated Network Management (IM 2003), March 24-28, 2003, Colorado Springs, USA*. Kluwer Academic Pub, 2003, p. 337.

[14] T. Fawcett and P. Utgoff, "Automatic feature generation for problem solving systems," in *Proceedings of the 9th International Conference on Machine Learning*. Citeseer, 1992, pp. 144–153.

[15] A. Liotta, G. Knight, and G. Pavlou, "Modelling network and system monitoring over the internet with mobile agents," in *IEEE/IFIP Network Operations and Management Symposium Conference Proceedings*. Citeseer, 1998, pp. 303–312.

[16] S. Hämäläinen, H. Sanneck, and C. Sartori, *LTE Self-Organising Networks (SON): Network Management Automation for Operational Efficiency*. John Wiley & Sons, 2012, no. ISBN: 978-1-119-97067-5.

[17] The Linux Foundation. (2017) The open network automation platform (onap). The Linux Foundation.

[18] ONAP. (2019) Closed control loop automation. [Online]. Available: https://docs.onap.org/en/latest/guides/onap-developer/architecture/onap-architecture.html#closed-control-loop-automation

[19] J. O. Kephart, "Research challenges of autonomic computing," in *Proceedings of ICSE'05, St. Louis, Missouri, U.S.A.* ACM, 2005.

[20] AT&T. (2016) ECOMP (Enhanced Control, Orchestration, Management & Policy) Architecture White Paper. Online Document. [Online]. Available: http://about.att.com/content/dam/snrdocs/ecomp.pdf

[21] L. Fallon *et al.*, "Using the COMPA Autonomous Architecture for Mobile Network Security," in *Integrated Network Management (IM 2017), 2017 IFIP/IEEE International Symposium on*, May 2017.

[22] ONAP. (2019) Policy framework architecture. [Online]. Available: https://docs.onap.org/en/latest/submodules/policy/parent.git/docs/architecture/architecture.html

[23] OASIS, "Tosca simple profile in yaml," Oasis Open, Tech. Rep. Version 1.2, Jan 2019. [Online]. Available: https://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.2/os/TOSCA-Simple-Profile-YAML-v1.2-os.pdf