Towards Autonomic Management of Communications Networks

Brendan Jennings, Sven van der Meer, Sasitharan Balasubramaniam,
Dmitri Botvich, Mícheál Ó Foghlú, William Donnelly
Waterford Institute of Technology, Ireland

John Strassner
Motorola Labs, USA

Abstract

As communications networks become increasingly dynamic, heterogeneous, less reliable and larger in scale, it becomes difficult, if not impossible, to effectively manage these networks using traditional approaches relying on human monitoring and intervention to ensure they operate within desired bounds. Researchers and practitioners are pursuing the vision of *autonomic network management*, which we view as the ability for network entities to self-govern their behavior within the constraints of the business goals that the network as a whole seeks to achieve. However, applying autonomic principles to network management is challenging for a number of reasons, including: (1) a means is needed to enable business rules to determine the set of resources and/or services to be provided; (2) contextual changes in the network must be sensed and interpreted, since new management policies may be required when context changes; (3) as context changes, it may be necessary to adapt the management control loops used to ensure that system functionality adapts to meet changing user needs, business goals, and environmental conditions; and (4) we need a means to verify modeled data and add new data *dynamically*, so that the system can *learn* and *reason* about itself and its environment. This paper provides an introduction to the FOCALE autonomic network management architecture, which is designed to address these challenges.

1 Introduction

The telecommunications industry has changed dramatically in recent years. Explosive growth of the Internet, the proliferation of mobile technologies, and fixed-mobile convergence has led to a progressively more complex, interconnected networking infrastructure. The ever-increasing difficulty in managing multi-vendor environments and the services they provide has altered forever the dynamics of the industry, the expectations of its customers and the business models with which it operates. In hardware, the impact of Moore's Law has had a profound effect across all sectors of the industry, encouraging equipment manufacturers, network operators, and service providers to continually strive to rapidly deploy the latest technology in order to gain competitive advantage. We believe that a downside of this rapid technology deployment is that current communications service offerings are inflexible in nature: they are rigidly defined; closely coupled to specific network technology; exhibit static functionality; and are often prone to security breaches. Critically, they are largely manually deployed and managed, requiring highly labor-intensive support structures, with consequent inflexibility and significant time to market constraints.

To address this problem, academic researchers and industrial implementers have been moving away from traditional network management approaches based on centralized control of a (relatively) small number of managed entities. Observing that networks are becoming more dynamic, more heterogeneous, less reliable and larger in scale, they are instead actively investigating the application of autonomic principles, aiming to simplify network management processes by automating and distributing the decision making processes involved in optimizing network operation. The goal is to allow expensive human attention focus more on business logic and less on low-level device configuration processes.

In this paper we introduce our approach to autonomic network management. We contend that the essence of autonomic management is the ability for a system to self-govern its behavior, but only within the constraints of the (human-specified) goals that the system as a whole seeks to achieve. We propose the use of information and ontological modeling to capture *knowledge* relating to network capabilities, environmental constraints and business goals/policies, together with reasoning and learning techniques to enhance and evolve this knowledge. Knowledge embedded within system models is used by policy-based network management systems [1] incorporating translation/code generation and policy enforcement processes that automatically configure network elements in response to changing business goals and/or environmental context. This realizes an autonomic control loop in which, as depicted in Figure 1, the system senses changes in itself and its environment, analyses this information to ensure that business goals and objectives are being met; expedites changes should these goals and objectives be threatened, and, closing the loop, observes the result.

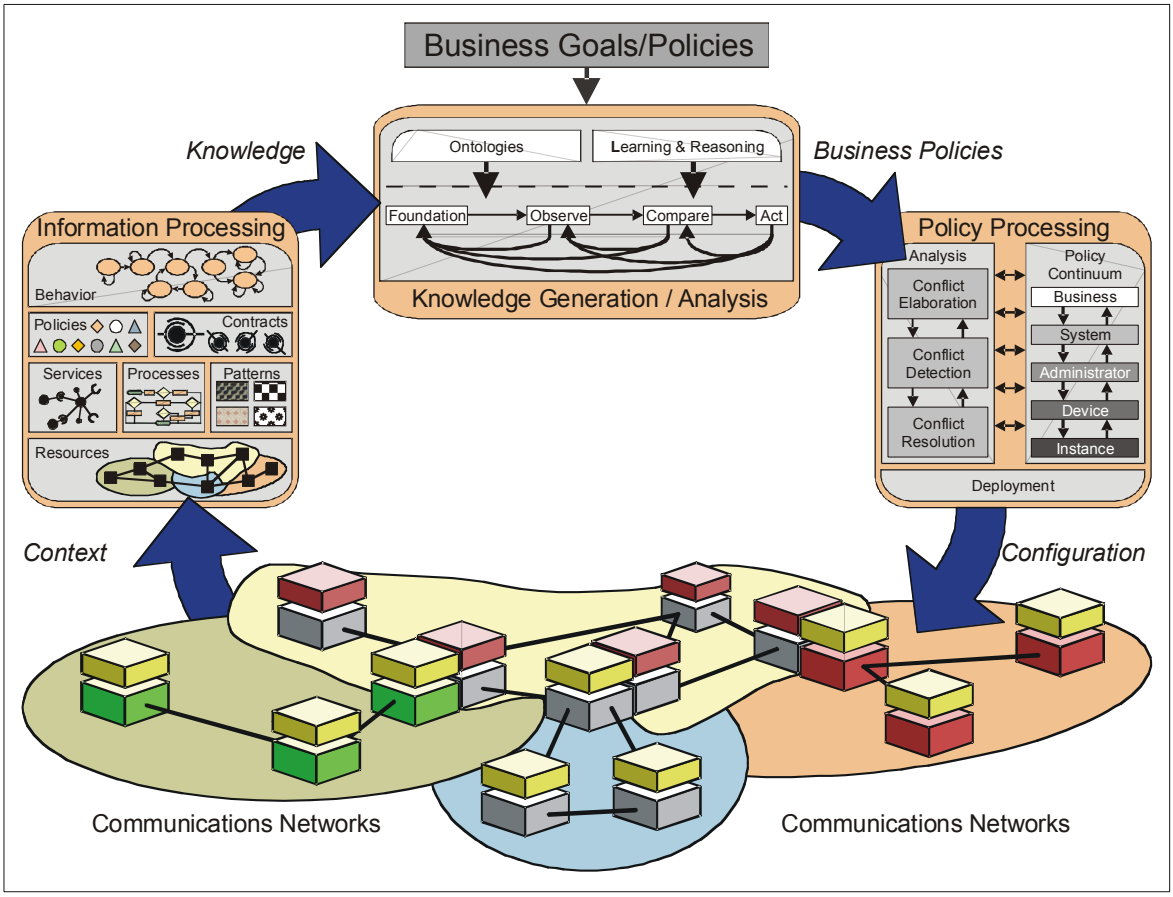

Figure 1. Conceptual Autonomic Control Loop for Network Management

We believe the information and ontological modeling based approach will deliver considerable improvements over existing manually configured network management systems, since it will support context-driven reconfiguration of networks with minimal human intervention at all but the high-level business view. Nevertheless, in order to deliver full autonomic network management capabilities, we believe it is also necessary to introduce decentralized processes and algorithms into the network infrastructure to maintain optimal or near-optimal behavior in terms of global stability, improved performance and adaptability, robustness and security. As described by Babaoglu et al. [2], many of these processes and algorithms can be profitably modeled on various biological processes found in the natural world. However, to ensure that they act in accordance with business goals, we argue that such processes and algorithms should themselves be modeled, so that their operation can be automatically configured via appropriate management policies.

This paper is structured as follows: in Section 2 we briefly summarize current research in the areas of autonomic computing and autonomic networking, contrasting our approach with other ongoing efforts. Section 3 introduces our conceptual model of an autonomic network management system, highlighting the main components that are needed to deliver effective management. Section 4 describes our work on embodying these concepts in FOCALE, our architecture for autonomic network management, whilst Section 5 discusses an initial prototypical realization of FOCALE. Finally, Section 6 summarizes the paper and discusses future work and open issues relating to the development and standardization of our work.

2 Autonomic Computing and Autonomic Networking

For many years, researchers have been aware that the structural and operational complexity of communications networks, and indeed distributed computing systems in general, has been increasing to the point where it is negatively impacting the economic viability of introducing new products and services to the market. In recent years the paradigms that have created the most interest as means of addressing this problem are that of *autonomic computing* [3], and later, *autonomic networking* [4]. Put simply, the autonomic paradigm seeks to reduce the need for human intervention in the management process through use of one or more control loops that continuously re-configure the system to keep its behavior within desired bounds.

The term autonomic computing was coined by IBM as an analogy of the autonomic nervous system, which maintains homeostasis (essentially maintaining equilibrium of various biological processes) in our bodies without the need for conscious direction. Autonomic computing attempts to manage the operation of individual pieces of IT infrastructure (such as servers in a data center), through introduction of an *autonomic manager* that implements an autonomic control loop in which the managed element and the environment in which it operates is monitored, collected data is analyzed, and actions are taken if the managed element is deemed to be in an undesired (sub-optimal) state [3]. The autonomic control loop is made up of Monitor, Analyze, Plan, and Execute components, all of which rely on a common knowledge repository. The Monitor component gathers data, filters, and collates it as required, and then presents it to the Analyze component, which seeks to understand the data and determine if the managed element is acting as desired. The Plan component takes these data and determines if action should be taken to reconfigure the managed element. The Execute component translates the planned actions into a set of configuration commands that can be applied to the managed element.

The autonomic computing vision can be summarized as that of a “self-managing” IT infrastructure in which equipment will have software deployed on it that enables it to self-configure, self-optimize, self-heal and self-protect. That is, it will exhibit what has come to be known as “self-*” behavior. Clearly this is a powerful vision, albeit one that is acknowledged by

Kephart [5] as requiring significant advances in the state-of-the-art over a number of years. It was therefore natural that the networking community would extend this vision from autonomic management of individual elements to autonomic networking – the collective (self-) management of networks of communicating computing elements. Of course, some of the network management work of the 1980s and 1990s could be retrospectively termed "autonomic networking", as some of the self-* issues were addressed, but in practice the term is a 21st Century one. Autonomic networking is currently a burgeoning research area, which seeks to integrate results from disciplines ranging from telecommunications network management to artificial intelligence, and from biology to sociology. For a good summary of the state-of-the-art in this fast-moving area, the reader is referred to Dobson et al. [6].

Much of the focus of research in autonomic network management is on the development of highly distributed algorithms that seek to optimize one or more aspects of network operation and/or performance, in essence aiming to provide various self-management capabilities. In this context, many researches are investigating the potential use of biologically-inspired algorithms and processes. As noted in [6], complex biological systems "tend to exploit decentralized and uncoupled coordination models, relying primarily on environment-mediated local information transfer." Examples include homeostasis (the tendency towards stable equilibrium between interdependent elements), chemotaxis (movement of a cell in a particular direction corresponding to a chemical gradient), and stigmergy (indirect communications between organisms through modification of their local environment), all of which have been used as inspiration for algorithm development – for descriptions of specific examples see [2].

Whilst work on development of decentralized self-management algorithms is crucial, and noting that significant advances have been made, we believe that deployment of these algorithms, whilst necessary, will not be sufficient. Equally important will be the flexible specification and enforcement of the goals these algorithms collectively seek to achieve – goals designed to ensure that the network successfully delivers services to users. Policy-based network management [1], is widely seen as an appropriate management paradigm to facilitate higher-level, human-specified cognitive decision making; therefore, many researchers are examining how policy-based management can be leveraged to help realize the autonomic vision (a good example is the work of Agrawal et al. [7]). However, to our knowledge little work has been done to date on integrating distributed self-management algorithms with policy-based management – for example by allowing policies to re-parameterize such algorithms to change their behavior to adapt to changing business goals. We believe this step is essential to provide a solution that balances the need for explicit control over network operation with the benefits of highly efficient and robust self-management algorithms and processes. Given this, a key goal of our work is to develop an architecture for autonomic network management that can profitably integrate the "top-down" explicit control model of traditional network management with the "bottom-up" emergent behavior associated with biologically-inspired, distributed algorithms. This requires numerous advances in the state of the art, chiefly with regard to:

- *Management of heterogeneous functionality*
  One of the problems in applying autonomic principles to networks is that networks are made up of many different devices that have different programming models and provide different management data describing the same, or similar, concepts. This makes it imperative to harness information models and ontologies to abstract away vendor-specific functionality, in order to facilitate a standard way of reconfiguring that functionality. Achieving this will allow legacy resources with no inherent autonomic capabilities be managed by autonomic systems.
- *Adaptability*

One of the promises of autonomic operation is to be able to *adapt* the functionality of the system in response to changes in user needs, business rules, and/or environmental conditions. This requires a more flexible governance approach than has been provided to date. In particular, the system needs to be able to sense context changes, and use policies specific to the new context to effect the required re-configuration of network devices.

– *Application of learning and reasoning techniques to support intelligent interaction*
Current examples of network device configuration and management rely on vendor-specific snapshots of static data. For example, statistics can be gathered and analyzed to determine if a given device interface is experiencing congestion. However, existing management data will not tell the user *why* congestion is incurring. This information must be inferred using these and other data, and retained for future reference. Hence, there is a need to incorporate sophisticated, state-of-the-art learning and reasoning algorithms into autonomic network management systems.

3 Conceptual Representation of an Autonomic Network Management System

Figure 2 presents a conceptual representation of an autonomic network management system incorporating an autonomic control loop enabled by the presence of one or more system models and ontologies that abstract the static structure, functionality, and dynamic behavior of the underlying network infrastructure, management functionality and offered services. These models and ontologies are continuously updated by the Model-based Information Processing component in response to the changing operational context of the network. This component

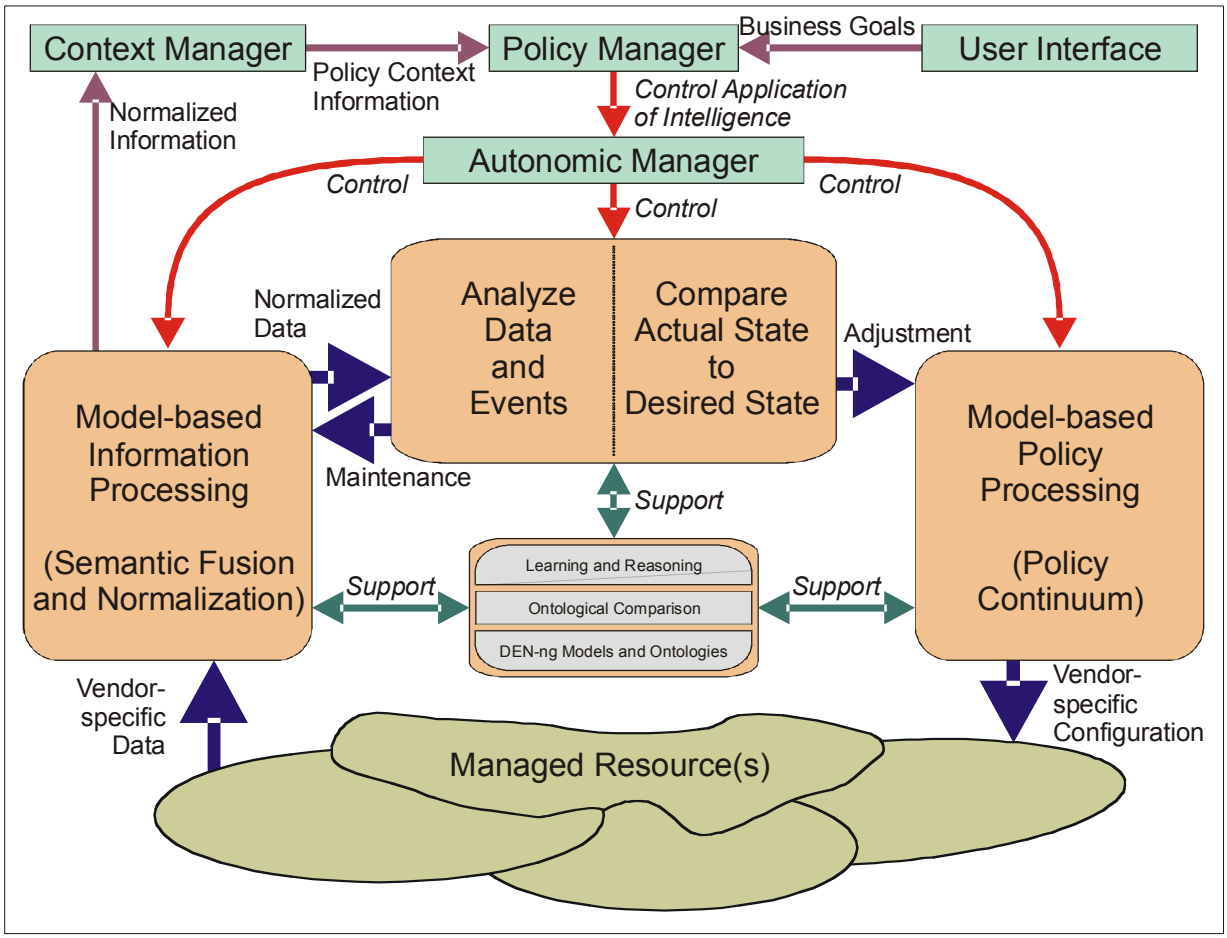

Figure 2. Conceptual Representation of an Autonomic Network Management System

gathers raw context information from managed resources (e.g., SNMP alarms) and using various analysis techniques, infers the impact, or potential impact, of this information (e.g., a network failure means that customer X is not being delivered the QoS level indicated in their SLA for service Y). It then passes normalized data relating to current operational context to the Event Analysis component, which employs ontological engineering, in conjunction with learning and reasoning techniques, to analyze whether the system's actual state corresponds to the desired state (as indicated by currently deployed set of policies).

If there is a mismatch between the detected system state and the desired state, two courses of action are possible. Firstly, if a there is a deployed policy specifying what should be done in this particular scenario that policy will be triggered via the Policy Processing Component, which utilizes knowledge embodied within systems models to automatically generate and apply updated network device configurations, which should bring the network back to the desired state. Alternatively, if no such policy exists (as may happen occasionally, since it will never be possible to model all possible operational scenarios for a complex network) information models and ontologies will be analyzed to determine what actions are required to bring the network back to the desired state. This will be codified in a new policy, which will be passed to the Policy Processing Component, where, as described above, it will be triggered to appropriately reconfigure the network.

The control loop described above is controlled by an Autonomic Manager, which influences the deployment of the policies that effect decision making within the loop. The Autonomic Manager receives up-to-date business level policies from the Policy Manager, which in turn manages policies that are created or modified by humans via a User Interface (e.g., business analysts may create policies indicating which services a new customer may access), or which are modified by the system itself based on information supplied by the Context Manager (e.g., in cases of network failures, policies relating to certain customers could be modified to deny them access to services in order to give preferential access to other, more important, customers).

The entire system, but particularly the Model-based Information Processing component, is reliant on the presence of information and data models that embody the knowledge necessary to represent managed resources (routers and other network devices) and their control using autonomic principles. In our work we use the DEN-ng information model (outlined in [8]), which we are currently enhancing with finite state machines to model behavior, and augmenting with ontological models that embody semantic information that cannot be represented in the Unified Modeling Language (UML). DEN-ng is a comprehensive information model for telecommunications, capturing everything from business concepts (e.g., products, service level agreements, and customers) down to low-level device functionality (e.g., packet marking, forwarding, and queuing). It is designed so that it can be readily augmented with vendor specific information and data models; it thereby provides a highly flexible and extensible modeling solution. Probably the best known application of DEN-ng is in the TM Forum standards, where parts of it have been used as the basis of the Shared Information and Data (SID) modeling effort.

The combination of a set of enhanced DEN-ng information and data models, combined with domain-specific ontologies for augmenting those models with required semantics allows information gathered from the network to be analyzed and used to ensure the models accurately reflect the current operational status. Following the examples above, when devices in the network fail resulting in localized lack of connectivity or decreases in available bandwidth, the myriad alarms raised can be collectively analyzed to ascertain which services, and therefore which customers, are being affected. Forwarding knowledge expressed in these terms facilitates decisions regarding which customers should be given preferential access to the network during the period in which the network is congested and therefore incapable of supporting all of its

customers. Moreover, the DEN-ng models and associated ontologies provide a knowledge base that can be used by machine learning and reasoning algorithms to both analyze collected data and automatically generate new knowledge that can then be leveraged to autonomically manage the underlying network infrastructure. In particular, this allows data gathered from the network to be analyzed and used to ensure the models accurately reflect its current operational status.

Another significant benefit of DEN-ng is its comprehensive policy model, which facilitates the specification of policy representation languages that are tightly coupled to the information model of the network which policies authored in that language will govern (as discussed below this characteristic is particularly useful for policy analysis). Furthermore, we can apply the DEN-ng policy continuum [8], in which policies at different levels of abstraction are organized in stratified business, system, network, device and device instance views – mirroring the different constituencies of people that will work together to define and deploy the policies that provide a product or service. Implementation of the policy continuum enables these constituencies, who understand different concepts and use different terminologies, manipulate sets of policy representations at a view appropriate to them, and to have those view-specific representations mapped to equivalent representations at views appropriate for manipulation by other constituencies.

The task of automating the refinement of business level policies (specified in terms of entities such as products, services and customers) down through the continuum, into corresponding device instance policies (specified in terms of entities such as packet marking rules or firewall configuration rules) is hugely challenging, since information needs to be added (and removed) as the policies become more specific in nature. Our approach is to harness the expressive power of ontologies to detail the nature of the relationships between concepts at different continuum levels. Policy refinement processes can access this knowledge from the ontologies, and applying ontological engineering techniques propose candidate refinements, which in certain cases will need to be ratified by humans; thus, we envisage policy refinement as a (semi-)automated process, where the level of human intervention decreases over time as the system learns from the outcome of previous refinements.

The Model-based Policy Processing component must also incorporate policy conflict analysis algorithms that: (1) elaborate newly defined/modified policies (e.g. by adding conditions relating to system constraints not evident to the policy author) so that conflicts are easier to detect; (2) detect sets of policies that will, or could potentially, conflict given certain network context; and (3) resolve conflicts by modifying or removing polices based on separate resolution policies, or by referring back to the appropriate policy author for a decision. Policy conflict analysis needs to be done at each level of the continuum, with high level policies only being "deployed" if they, and all the policies associated with them at lower levels of the continuum, are detected as being conflict-free. Policy conflict analysis is widely research and is acknowledged as an extremely difficult challenge; however, we believe significant advances can be made by harnessing the semantic information available in DEN-ng and associated ontologies to facilitate more powerful conflict analysis algorithms then those currently available – our initial work on this approach is described in Section 5, and in more detail in [9].

Finally, we note that the model-centered approach outlined above primarily provides for explicit control of network behavior and, as such, can be viewed as an evolution of traditional network management approaches. However, this approach is limited by the capabilities (and configurability) of the network devices and the ability to maintain up-to-date information and ontological models of very complex and highly dynamic network topologies. True autonomic network management will, we believe, additionally require the deployment of processes and algorithms within network devices themselves. These would act in a highly distributed manner,

serving to optimize network behavior with respect to stability, performance, robustness and security – in effect providing the kind of self-management functionality discussed in Section 2. We argue that these processes and algorithms need to be incorporated into the overall model-centered management process so that their operation can be re-parameterized to modify their behavior to satisfy high-level policies. This provides the necessary integration point between the (top-down) model-centered management approach and the (bottom-up) self-management approach in which highly distributed algorithms applying local rules give rise to desired emergent global behavior.

4 The FOCALE Autonomic Network Management Architecture

In this section we describe how the concepts for autonomic management described in Section 3 are realized in the context of a concrete architectural model for distributed autonomic network management systems. The architecture, named FOCALE (**F**oundation – **O**bservation – **C**omparison – **A**ction – **L**earn – r**E**ason) is based on the observation that business objectives, user needs and environmental context all change dynamically. Therefore a single, statically defined, management control loop is insufficient – we need the ability to adapt the behavior of the control loop so that it can effectively manage the network to react appropriately to observed or hypothesized changes. FOCALE implements two control loops: a *maintenance control loop* is used when no anomalies are found (i.e., when either the current state is equal to the actual state, or when the state of the managed element is moving towards its intended goal); and an *adjustment control loop* is used when one or more policy reconfiguration actions must be performed, and/or new policies must be codified and deployed.

Of course, it is unreasonable to assume that a single entity will be able to maintain all the information required to realize the FOCALE control loops for large scale networks containing large numbers of heterogeneous (in terms of available functionality, vendor-specific programming model and specific configuration) devices. Therefore, FOCALE must be a distributed architecture, to the degree that even individual network devices may incorporate autonomic management software implementing the maintenance and adjustment control loops. To this end FOCALE assumes that any managed resource (which can be as simple as a device

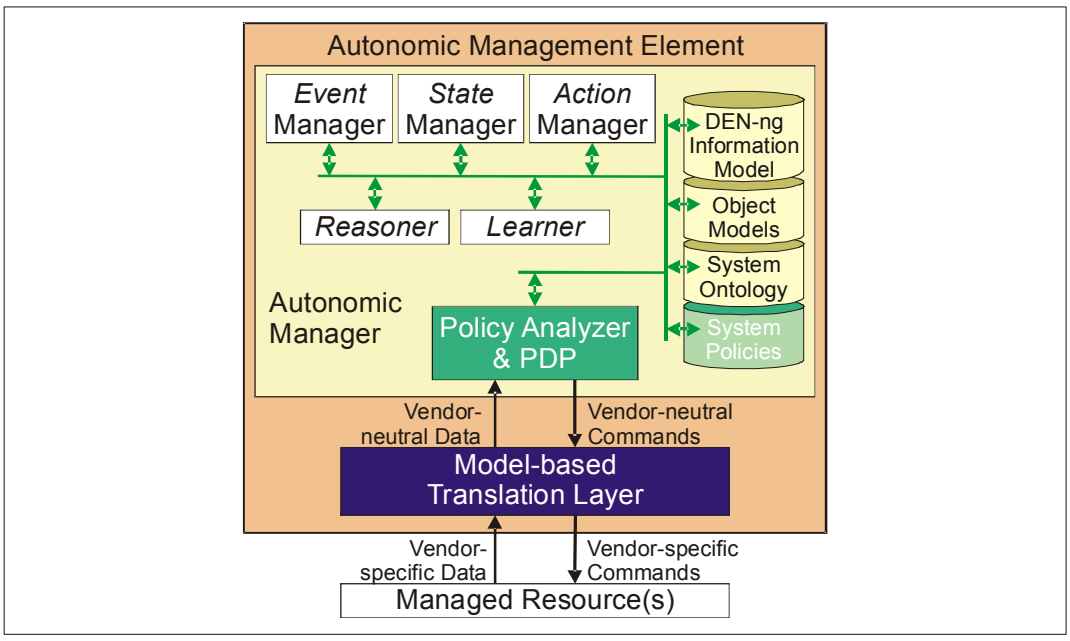

Figure 3: FOCALE Autonomic Management Element Functional Architecture

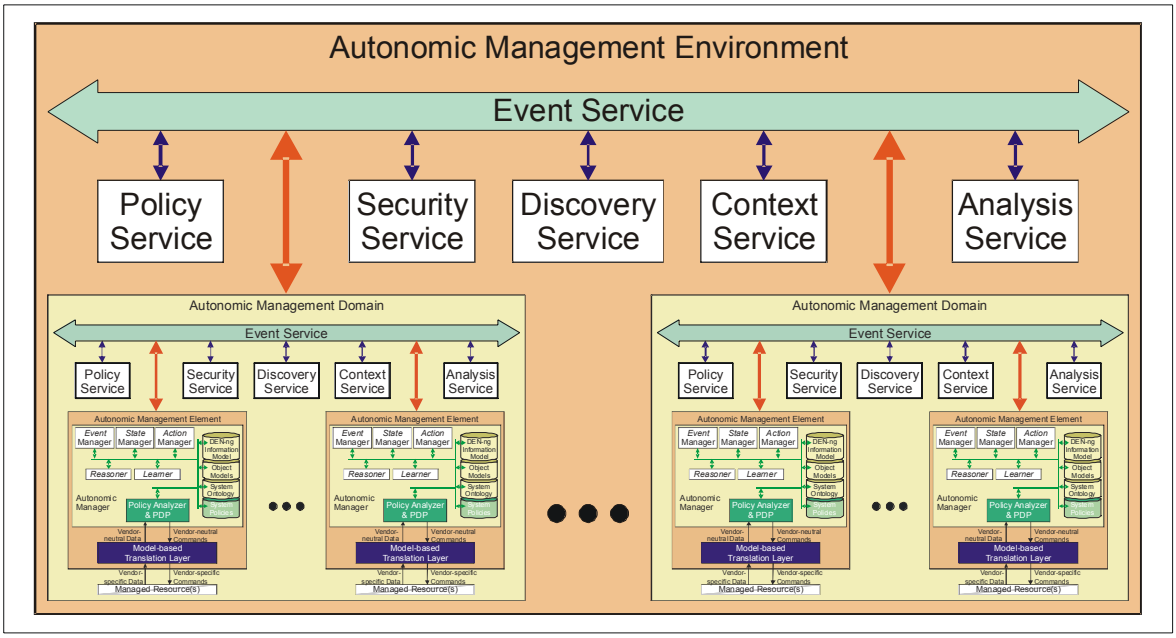

Figure 4: FOCALE Autonomic Management Environment Functional Architecture.

interface, or as complex as an entire system or network) can be associated with an *Autonomic Management Element (AME)*, by interfacing the functionality of the Managed Resource to the functionality of an *Autonomic Manager (AM)* using a *Model-Based Translation Layer (MBTL)*, as shown in figure 3. As figure 4 shows, AMEs can be modularized to first form a uniform *Autonomic Management Domain (AMD)*, and then to an *Autonomic Management Environment*; with each level containing Policy, Security, Discovery, Context and Analysis services that serve to harmonise the operation of the AMEs/AMDs.

The Autonomic Management Architecture contains two main functional components: the Autonomic Manager (AM) and the Model-based Translation Layer (MBTL). The AM is independent of the vendor-specific functionality/data of the underlying managed resource(s), which facilitates easier communication between AMEs for coordination of management decision making. Each AM realizes the autonomic management functionality described in section 3 via an Event Manager, a State Manager, an Action Manager, a Reasoner, a Learner, and a Policy Analyzer / Policy Decision Point (PDP). All these sub-components can communicate with each other and have access to the DEN-ng information model, an object model reflecting the current state of the AME's managed resource(s), the system ontology, and the set of deployed policies governing the AME's managed resource(s). When the AM receives context information via the MBTL, the Policy Analyzer/PDP ascertains if the conditions of any deployed policies are satisfied; if they are the corresponding actions are applied via the MBTL. If the Policy Analyzer / PDP does not recognize the context it contacts the Event and State Managers, which use the models/ontologies to ascertain if the system is in a desired state. If it is not, the State Manager employs the Reasoner to identify actions that will lead the system back towards its desired state. Once identified, the Action Manager coordinates the enforcement of these actions by the Policy Analyzer / PDP. Subsequently, the Learner monitors the effectiveness of actions identified in this manner; if successful these actions are codified as one or more policies that are then added to the set of system policies. Of course AMs also has the ability to communicate with other AMs to coordinate activities such as analysis of global network state, or introduction of new policies.

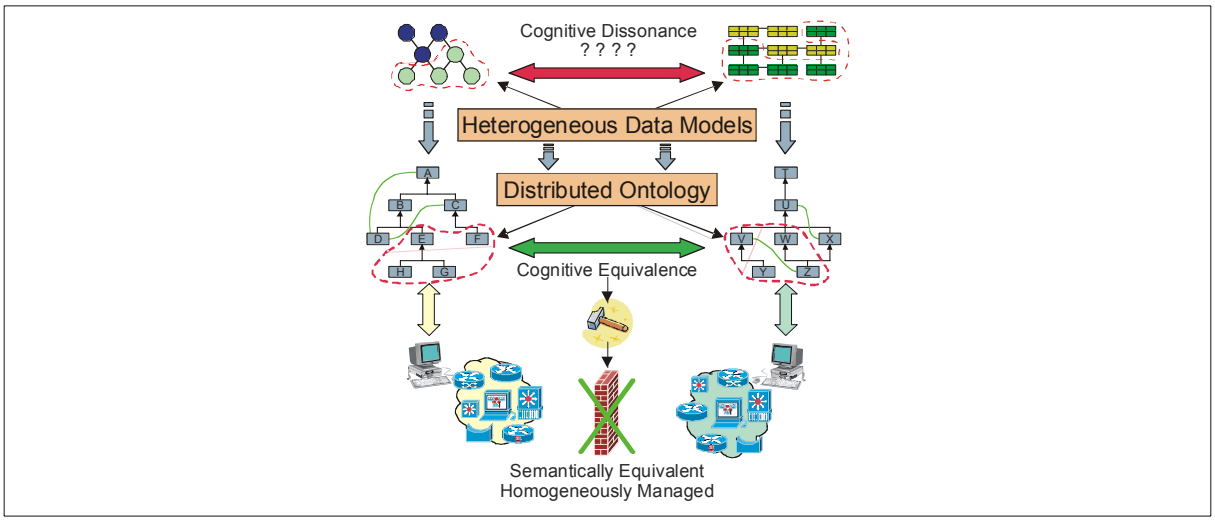

Figure 5: Use of Ontologies to Identify Cognitive Equivalence across Heterogeneous Data Models.

Unlike the AM, the MBTL must have in-depth knowledge of the managed resource(s) to allow it translate normalized vendor-specific data gathered from the managed resource(s) into DEN-ng compliant vendor neutral data (context information) to pass to the Policy Analyzer / PDP, and vice versa for configuration commands. As alluded to in section 3, DEN-ng can be readily extended with vendor specific information and data models (e.g., relating to new releases of CLI command sets for a family of network devices). Assuming that the DEN-ng information model is extended in this manner for all the managed resource(s), and furthermore, that the system ontology is extended to incorporate semantic information detailing the meaning of various vendor specific data/commands, the MBTL can employ ontological engineering techniques, including semantic similarity matching (for a description see [10]), to map between DEN-ng vendor neutral representations and vendor specific representations.

The basis of the MBTL approach is depicted in figure 5, which shows a typical network scenario in which different devices having different data models are managed using different tools. This creates cognitive dissonance between data in the two data models – since there is no common vocabulary with established meanings defining the data and relationships, it is impossible to directly compare data from different sources, which in turn means that it is impossible to see if those data are related to each other. By using an ontology to augment the facts represented in these data models, each fact can be mapped into a common vocabulary, which enables each fact to be augmented with appropriate semantics – enabling cognitive similarity between these different facts to be established. Conceptually, the association between a set of nodes in a model and the set of nodes in an ontology creates a new set of associations, bridging the gap between how knowledge is represented between these different approaches.

5 Prototype Implementation

We now provide a brief overview of our ongoing work in building a prototype realization of the FOCALE architecture; for a fuller description of this prototype, the reader is referred to [9]. Figure 6 depicts our current implementation of a single FOCALE AME, which targets aspects of traffic conditioning in a simulated IP-based network of an ISP, over which customers are offered a small number of communications services. It should be noted that the simulated network is very loosely coupled to the AME implementation – in the next phase of development we plan to replace the simulation with real routers that will be configured by CLI commands generated by the AME and which will provide context information to the AME via SNMP. As it

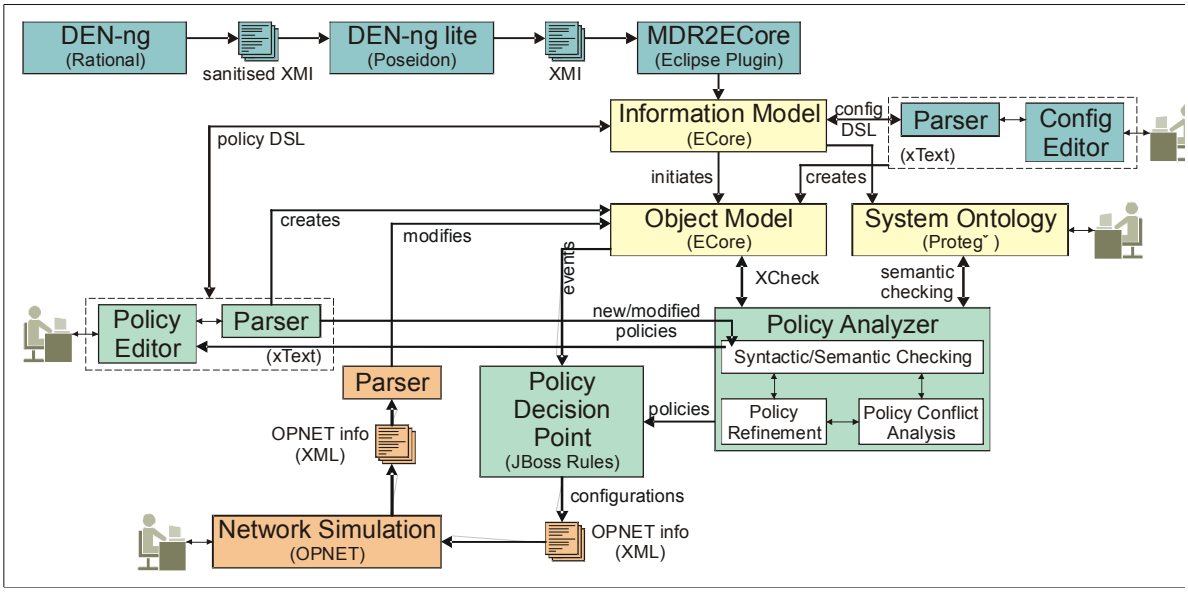

Figure 7: FOCALE Autonomic Management Element Prototype

stands our OPNET™ based simulation is configured to emit information relating to network events, and read and apply new router configurations generated by the AME.

Core to the prototype are the object model and system ontology, which provide synchronized models representing the current state of the simulated network, the customers and services it supports, and the policies deployed in it. To initially set up these models we have created a configuration Domain Specific Language (DSL) and editor that enables the creation of information model instances (i.e., object models) to represent the structure of the managed system. We use the textual DSL framework called xtext, created by open Architecture Ware (available through Eclipse's Generative Modelling Technologies (GMT) initiative). The configuration DSL is generated from the model elements in the DEN-ng model that are marked as relevant for describing the structure of our particular managed network – this offline process is depicted at the top of Figure 6. The same technique used to create the configuration DSL and its editor were used to generate an event-condition-action (ECA) policy DSL that is based on the policy representation entities in the DEN-ng model. The ECA policy DSL has the following semantics: on the occurrence of a set of events, if the condition clause evaluates to true, then execute the action clause. Separation of the configuration and policy DSLs and their editors allows the policy editor to be used during system operation to create, modify, or withdraw policies.

A formal representation of the information model subsets used for generating the DSLs (configuration and policy) is required for automated reasoning in policy analysis – the OWL-based system ontology provides this ability. We use IBM's EMF Ontology Definition Metamodel (EODM) to produce an OWL representation of the identified subset of the DEN-ng information model. The resulting baseline ontology can be viewed within Eclipse with the IODT plug-in or saved and opened with the Protégé OWL editing tool. Our baseline system ontology has been edited to embody semantic information useful for analysis processes such as policy conflict detection.

The Policy Analyzer uses the object model and system ontology models to build a more complete understanding of the characteristics and behavior of policies and how they affect managed resources. To help provide this understanding we translate policies into a format suitable for deployment on a rule inference engine based Policy Decision Point. We use JBoss

Rules (based on the Drools rule engine), which uses a tailored object-oriented form of the Rete algorithm called Rete-OO for evaluating the rules. The Rete algorithm efficiently stores rules in memory in the form of a network so that it can take advantage of rule patterns to reduce the number of conditions that need to be evaluated. Policies are particularly amenable to Rete-based rule engines, as sets of deployed policies typically share event, condition, and even action parts.

As depicted in Figure 6, these components cooperate to realize FOCALE maintenance and adjustment control loops. For the maintenance loop a parser detects changes in the operational context of the network simulation (e.g. a bandwidth utilization threshold on a link being exceeded) and updates the object model accordingly. This object model change triggers evaluation of policies deployed in the JBoss rules engine; if the conditions of one or more policies are satisfied the appropriate actions are invoked on the simulated network via reconfiguration of simulation parameters (e.g., a low priority customer is denied access to a service, thereby reducing link bandwidth utilization to below the threshold value). For the adjustment loop human users can use the policy DSL editor to create/modify/withdraw policies, via an iterative process in which policies "proposed" by the user are analyzed by the Policy Analyzer for syntactic and semantic correctness, and potential for conflict with other deployed policies. This occurs at each step of their refinement down the policy continuum. If, at any stage, the problem cannot be resolved by the Policy Analyzer the user is informed of the problem and prompted to re-edit the policy. Once policy analysis is complete the created/modified policy is converted to JBoss format and deployed on the JBoss PDP. In this manner the operation of the maintenance control loop is adjusted.

6 Summary and Outlook

This paper has advocated the principle of *self-governance* as the basis for realizing communications networks that operate and are managed autonomically. We introduced the FOCALE autonomic network management architecture, which has a number of distinctive characteristics: (1) emphasis on the use of business goals (codified as policy rules) to determine how resources in the network should be collectively utilised to best deliver services to users; (2) context-aware policy management processes to adapt the management control loops used to ensure that system functionality adapts to meet changing user needs, business goals, and environmental conditions; and (3) a novel combination of information and data modeling, augmented by ontological data, to enable the system to *learn* and *reason* about itself and its environment. We emphasized the difficulties imposed by the inherently heterogeneous and interconnected nature of networked communications systems, and described how FOCALE's architectural components are designed to overcome these difficulties.

The grand vision of autonomic computing and autonomic networking is that of a completely self-managing infrastructure that can itself access, or generate, the knowledge it needs to allow it optimally react to changing operational context. Although this vision is very attractive, it is unlikely to ever be fully realized, especially in the context of the hugely complex and dynamic global communications infrastructure. In developing FOCALE we take a more pragmatic approach: rather then seeking to entirely eliminate human intervention from the management process, we seek to minimize it and to focus it more on business concerns then on low-level device configuration. In particular, we acknowledge that human intervention will sometimes be required for the refinement of policies down the policy continuum, and also to resolve policy conflicts never before encountered by the system. Of course, the degree to which this will happen is highly dependent on the completeness, correctness and timeliness of the knowledge embodied within the system's information, data and object models, and associated ontologies. We believe that DEN-ng is the most exhaustive and well-structured information model currently available, and as such provides a solid basis for realization of FOCALE.

In the paper we also briefly described our prototype realization of FOCALE. We are currently using this prototype to examine the tradeoffs between management sophistication and derived benefits as a function of network size, number of users, mix of traffic, and other factors. In parallel, we are working on replacing the simulated target environment with a small number of real routers – this process should provide valuable insights in to the practical implications of using information models and ontologies to automate the generation of CLI commands. Subsequently, we plan to explore integration of bio-inspired algorithms into FOCALE, and to demonstrate their added value via implementation in our prototype. Once sufficient experimental results are obtained, we plan on submitting our results to the Autonomic Communications Forum standards body.

Acknowledgements

We wish to acknowledge the valuable insights provided by Nazim Agoulmine in the development of FOCALE, and the work on prototype design and implementation carried out by Keara Barrett, Alan Davy, Steven Davy, and Elyes Lehtihet. This work has received support from Science Foundation Ireland under the "Autonomic Management of Communications Networks and Services" award (grant no. 04/IN3/I404C).

References

[1] M. Sloman, "Policy driven management for distributed systems", *J. Netw. Syst. Manag.*, vol. 2, no. 4, Dec. 1994, pp. 333-360.

[2] O. Babaoglu *et al.*, "Design patterns from biology for distributed computing," *ACM Trans. Auton. Adapt. Syst.*, vol. 1, no. 1, Sep. 2006, pp. 26-66.

[3] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, Jan. 2003, pp. 41- 50.

[4] J. Strassner, "Autonomic Networking – Theory and Practice" [tutorial], *Proc. 2004 IEEE/IFIP Network Operations and Management Symp. (NOMS 2004)*, IEEE, April 2004, pp. 927.

[5] J. O. Kephart, "Research challenges of autonomic computing," *Proc. 27th Int'l. Conf. on Software Engineering*, ACM Press, 2005, pp. 15-22.

[6] S. Dobson *et al.*, "A Survey of Autonomic Communications," *ACM Trans. Auton. Adapt. Syst.*, vol. 1, no. 2, Dec. 2006, pp. 223-259.

[7] D. Agrawal, K. W. Lee and J. Lobo, "Policy-based management of networked computing systems," *IEEE Commun. Mag.*, IEEE, Oct. 2005, vol. 43, no. 10, pp. 69-75.

[8] J. Strassner, "DEN-ng: achieving business driven network management," *Proc. 8th IEEE/IFIP Network Operations and Management Symp. (NOMS 2002)*, IEEE, Apr. 2002, pp. 753-766.

[9] K. Barrett, S. Davy, J. Strassner, B. Jennings, S. van der Meer and W. Donnelly, "A Model Based Approach for Policy Tool Generation and Policy Analysis," *Proc. 1st IEEE Int'l. Global Information Infrastructure Symp. 2007 (GIIS 2007)*, IEEE, 2007, pp. 99-106.

[10] Y. Kalfoglou and M. Schorlemmer, "Ontology Mapping: the state of the art," *Knowl. Eng. Rev.*, vol. 18, no. 1, 2003, pp. 1-31.