Policy-Based Architecture to Enable Autonomic Communications – A Position Paper

Steven Davy, Keara Barrett,
Sasitharan Balasubramaniam, Sven van der Meer,
Brendan Jennings
Telecommunication Software and Systems Group
Waterford Institute of Technology
Cork Road, Waterford, Ireland
{sdavy, kbarrett, sasib, vdmeer, bjennings}@tssg.org

John Strassner
Motorola Labs
Schaumburg, IL, USA
{john.strassner}@motorola.com

Abstract— **The increase in complexity of network management systems and a consequent lack of association to business requirements has driven the need for autonomic communications. By integrating context information, autonomic computing can provide more efficient means to counter technical problems found in complex network systems and at the same time address associated business requirements. In this paper, we propose an autonomic communications architecture that manages complexity through policy-based management where we incorporate a shared information model integrated with knowledge-based reasoning mechanisms to provide self-governance behavior.**

Keywords: Autonomic Communication, Shared Information and Data model, DEN-ng, Policy Framework, Analytic Hierarchy Process

I. INTRODUCTION

In recent years, there has been a significant increase in business, system, and operational complexity [14]. Fundamentally, complexity is an inherent problem of doing business. Two examples of increasing complexity are the manual management of network functionality (which is inherently error prone) and the probability of losing revenue if the operation of a business cannot dynamically adjust in response to the changing environment. Moreover, efforts such as Motorola's Seamless Mobility initiative [15] require the ability of a system to dynamically adjust the services that it provides based on a number of factors, including the introduction and removal of new and existing components, environmental conditions, security threats and permissions, and so forth. In short, the needs of the business (as codified by policies) must drive the resources and services that the network provides.

Autonomic computing [13] addresses managing and controlling complexity through the application of policy to ***govern*** adaptive behavior. This is based on three important concepts: (1) the sharing and reusing of common information and knowledge, (2) the application of machine learning and knowledge-based reasoning to guide the changes in behavior of the system, and (3) an extensible and flexible governance model that forms a closed control loop that learns from its decisions. Most of the current research work in autonomic computing is focused on the IT environment. Hence, much is not applicable to a wide range of applications. For example, current solutions for routing mechanisms used for ad hoc networks are only limited to static network layer routing mechanisms. Hence, they cannot be used to optimize Quality of Service (QoS) required by a given set of applications that all use the same (converged) or different (seamless mobility) network infrastructures. Autonomic communications can address the business as well as the technical problems inherent in network management by considering a variety of contextual information (e.g., the business requirements of service providers, user application and billing profile requirements, and the traffic conditioning capabilities of participating network interfaces) to provide a holistic solution that can adapt to changes in the technical and/or business environment.

This paper proposes an autonomic networking and communications architecture suitable for next generation networks. Our architecture is based on an information model that represents the managed environment and the entities within it. This information model includes a policy management model; hence, ***all*** managed entities can be managed using policy rules. This paper will focus on the policy framework as well as the functionality of their elements, which includes the capability to refine high-level business policies to low-level network element policies through the use of machine learning and knowledge-based reasoning mechanisms.

The structure of the paper is as follows. Section 2 presents the current work on autonomic computing and policy management as well as their limitations. Section 3 will describe our architecture, while section 4 will discuss the policy framework and its functionality. Section 5 will provide a scenario of our prototype and section 6 the conclusions.

II. RELATED WORK

We separate the related work into three sections: autonomic systems, policy management, and policy refinement.

A. Autonomic Enviroments

Prehofer and Bettstetter [1] have defined self-organization as a system-wide adaptive structure with no external or central dedicated control entity, where individual entities interact with

each other in a peer-to-peer fashion. They outlined four design paradigms for achieving self-organization, which includes defining local rules to achieve global properties by distributing responsibility among individual entities, incorporate current information to deduce new information, avoid maintaining long-lived state information, and use protocols that are adaptable to changes within the environment. Yagan and Tham [3] proposed a self-optimizing, self-healing architecture for QoS provisioning in Differentiated services. The architecture employs a model free Reinforcement Learning (RL) approach. However, test results have shown this methodology is not suitable for dynamic networks. IBM has defined an autonomic computing architecture [16], which focuses on enabling self-management functionality. This shifts the burden of managing systems from people to technologies based on an enterprise view of appropriate policies. The architecture is based on an intelligent control loop that monitors, analyzes, plans and executes based on a perception of the current environment. It describes an evolutionary approach wherein the ways in which an IT infrastructure evolves. Lanfranchi et al [2] proposed an autonomic monitoring system based on the resource model concept where the monitoring tool models and implements the aspect of the entities as objects, and predicts and corrects any abnormal behavior. However, the monitoring tool is limited to single machine fault detection rather than network system environments. At the same time, manual intervention is required in the design and deployment phase.

B. *Policy Management*

Policy-based management (PBM) is a concept developed originally for reducing the administrative complexity of reconfiguring the network whenever the behavior of the network needed to be changed. The manual process of reconfiguring the network is enormous, because of the vast amount of different devices that make up the network [13]. Policy-based management addresses this complexity using abstraction. Since there are multiple constituencies (e.g., business analysts, system architects and network engineers) that use policy, the notion of a "single" policy is erroneous. The policy continuum [17] was defined to address this concern. However, this does not solve the problem of detecting policy conflicts and resolving them. Early solutions for conflict detection were based on providing general rules, as in obligations and prohibitions. Meta-policies were suggested to resolve conflicts in distributed environments, including mechanisms for dynamic policy conflict detection and resolution. Dunlop et al [12] proposed the use of a policy type to explore the varying semantics associated with different policy conflicts. However, this database is statically defined offline, and is therefore centralized. Chadha et al [10] proposed a management architecture for mobile ad-hoc networks, based on the IETF PCIM policy model [18]. However, this model does not take into account context information that may improve policy decisions in different network environments. In addition, autonomic functionality could improve the ability of the network to function under harsh network conditions, but none is proposed. Zhuang et al [11] propose a policy-based management architecture to manage QoS in an integrated UMTS and WLAN environment. However, the SID policy model allows for better abstractions across divergent networks as it is a shared information and data model. The addition of autonomic functionality would improve divergent network crossover with the abilities of self-configuration and self-learning of network, service and user requirements.

C. *Policy Refinement*

While the importance of policy refinement for large distributed systems has been acknowledged, it is still a largely overlooked research domain. In 1999, researchers at Hewlett-Packard Laboratories published a report on a practical approach for policy refinement called the Policy Wizard Engine for Refinement or POWER prototype [7]. However, the POWER prototype requires significant interaction and system knowledge from a human operator to set up the refinement templates for a specific system and to translate the policy from one level of abstraction to another. This refinement protocol does not include analysis capabilities and is more of a guide for policy refinement than an automated policy refiner, which is not suitable for autonomic management. Bandara [8] describes a goal-based approach to policy refinement. The system uses event calculus in conjunction with abductive reasoning to derive the sequence of events that achieve the desired goal. However, the system is not suited for autonomics since it expects the users to provide a representation of the system description, in terms of the properties and behavior of the components and the goals that the system must satisfy. The user is also expected to define the pre and post conditions of the operations supported by components using state charts.

III. ARCHITECTURE

Our proposed autonomic communication architecture is illustrated in Figure 1. It is organized using four distinct architectural constructs: shared information, virtual software, infrastructure, and policy.

A. *Shared Information*

The Information model is based on the Shared Information/Data model (SID) of the TeleManagement Forum. The SID is a subset of the Directory Enabled Network – next generation (DEN-ng) initiative, which is a unified model that integrates the information model and associated policies [6]. Note that it uses abstraction mechanisms and software patterns to provide an extensible structure that can integrate other models or modeled information as appropriate. The SID is a key part of the NGOSS initiative; it represents information to be managed at each stage of development of a BSS/OSS. The SID is used as a reference model and common language for representing the business purpose and technology-neutral definition of the entities involved in the processes of a service provider. As such, it can be used to translate and refine high level business policies to low level network element policies, and supports the formation of the context model for the virtual software layer. DEN-ng defines the notion of a software contract, based on extensions to the Design by Contract paradigm of Bertrand Meyer. This codifies the interaction between entities exchanging information and communicating with each other.

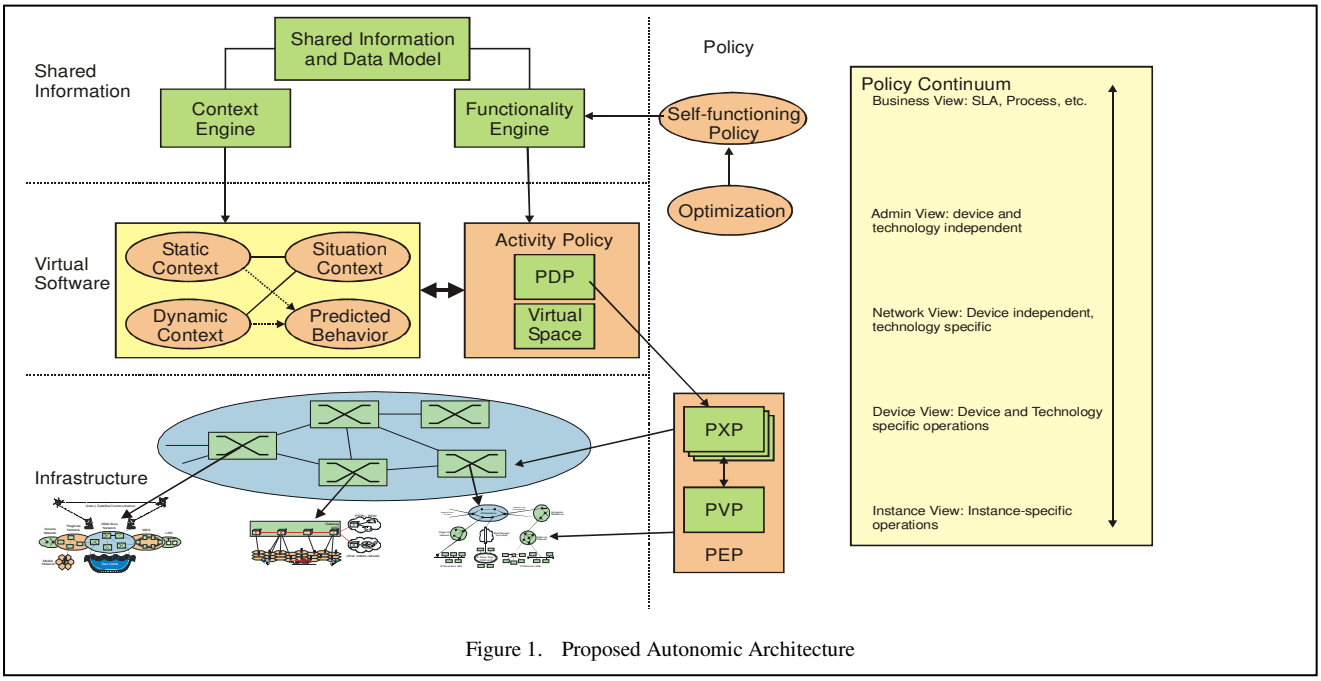

Figure 1. Proposed Autonomic Architecture

B. *Virtual Software*

The purpose of the virtual software is to support autonomic functionality for different heterogeneous networks and components. Autonomics achieves governance through (intelligent) decision-making. Hence, this software contains a model of the ***desired*** behavior as well as the ***deduced current behavior*** of the system or component being managed. In essence, a novel combination of information modeling and ontological engineering [19] enables us to attach semantics to facts; knowledge-based reasoning processes, such as abduction and machine learning, are then used to develop a representation of behavioral orchestration (including what actions to perform if a problem occurs). The context model holds information relevant to the current activities, including device- and vendor-specific managed elements and user profiles/preferences. The context of a situation can model both static and dynamic contextual environments. The predicted behavior context information is inferred from the static as well as the dynamic environment context information. Due to space limitations, we will not elaborate on our context model. The situation context information is then evaluated by the active working set of policies that are required in response to environmental, business, and/or technological changes. Changes from the environment are monitored by various sensors and agents.

C. *Infrastructure*

The infrastructure includes network elements and other computing devices as well as software that manages them. The type of protocols and adaptability mechanisms used depend on the current context as well as the type of behavior that is being orchestrated. For example, ad hoc networks may dynamically change or combine routing protocols (e.g., link state routing and distance vector routing), to suit the computing environment.

D. *Policy*

This component is based on the DEN-ng policy model, which enhances the IETF and DMTF policy architecture. It separates the functionality of a Policy Enforcement Point into a Policy Execution Point (PXP – for carrying out specified policy actions) and a Policy Verification Point (PVP – for ensuring that the policy actions executed correctly and, more importantly, did what was expected). A Policy Decision Point (PDP) distributes various levels of decision-making amount global and local scopes. A Policy Broker enables multiple Policy Servers (consisting of at least one PDP, PXP, and PVP) to negotiate and exchange policies. It is important to note that Policy doesn't belong to any one particular "layer", but rather spans and interconnects the other three components.

IV. POLICY FRAMEWORK

The policy framework is the foundation for decision-making in our autonomic communication environment. This section will describe the mechanisms for policy refinement, policy optimization techniques, and mechanisms for conflict detections.

A. *Policy Refinement*

Policy plays a central role in an autonomic architecture, as it formalizes the concept of decision-making. This means that the concepts and vocabulary for each constituency must be reconciled. Our policy framework uses the DEN-ng policy continuum to enable policies to be written for different constituencies using vocabularies native to each constituency. The resulting *set* of policies are then reconciled into a common policy language in the Policy Refinement module. There are several types of policies used in our autonomic framework. The policy continuum defines five sets of policies (business, system, network, device, and instance (low-level configuration)). Autonomic communications defines a sixth, called a *self-functioning policy*. The different levels of

abstraction of the Policy Continuum are refined using a combination of information modeling, ontological engineering, and knowledge-based reasoning for each level of the continuum. The information model is used for two distinct purposes: (1) to represent policies of all levels using the same set of building blocks, and (2) to explicitly define which set of managed entities are the subject and target of the policy. Ontological policy specifications are used to represent meanings and some semantics for policy as used by each constituency involved in the management of the entity and/or system. A dynamic mapping between the ontological aspects of each level of the policy continuum is then used to relate the different semantic information together. This gives the added advantage of alleviating system administration from requiring knowledge of policies specific to each constituency.

The self-functioning policies are responsible for performing functionality required in an autonomic communication environment. Kephart and Walsh [5] pointed out that the common definition of policies does not cover the spectrum of autonomics, and must be extended to define any formal behavior desired. We take a similar approach towards defining autonomic policies by employing optimization and knowledge-based reasoning techniques, and mapping these techniques to the transformation of policies at each level of the continuum. At the same time, a key requirement of autonomics is the ability for each network element to establish cooperative role with peer elements and work towards a common goal. In order to provide this mechanism, we define self-functioning negotiation processes between different architectural elements to reach the optimized goal specified by the self-functioning policies.

B. Policy Conflict and Detection

Since each element may contain multiple self-functioning policies, policy conflicts may arise. A policy conflict occurs when the conditions of two or more Policy Rules that apply to the same set of managed entities are simultaneously satisfied, but the actions of two or more of these Policy Rules conflict with each other. Therefore, a distributed approach towards providing a dynamic policy conflict detection and resolution process is essential. The approach we have taken is to aggregate the different policies by policy type, and determine the policy conflict probability between different policy types. This will provide a mechanism to sequence policies that may be triggered without leading to conflict situations. However, if the number of policy types increases, this calculation could become computationally difficult. Hence, we incorporated Reinforced Learning (RL) mechanisms to detect and learn the various conflicts that may occur between policies. The benefits of RL is its ability to support decision strategies for unknown systems with large dimensions that constantly change. This also enables new policies to be added safely. This is achieved by creating a virtual simulation space scoped by the network element, and examining how the new policy would execute in the virtual space. Our system provides the ability to self-learn the resulting behavior of this policy without having to deal with real conflicts. Once enough test results can ensure that the new policy has established its own definitive policy type and executes safely in simulations, the policy is then permitted to execute in the system.

V. SCENARIO

The proposed architecture is part of our ongoing work toward building an autonomic communication framework. We have developed a prototype to test the efficacy of the different architectural constructs for a service provider to test the offering of new service bundles. The scenario assumes a change in pricing structure from the service provider for an IP network that employs Differentiated Service (Diffserv) for QoS. In this scenario, the service provider has a total bandwidth capacity of 1Gb and offers three types of service classes: EF (Expedited Forwarding), AF_1 (Assured Forwarding), and AF_2. The types of applications for each traffic classes are as follows: class EF is suitable for high quality multimedia (256Kbps), class AF_1 is suitable for low quality multimedia (64Kbps), and AF_2 is for web applications (33Kbps). The scenario demonstrates the ability of the Policy Continuum to refine high-level business policies to control the changing of router QoS configurations by combining an optimization technique with the Information model. The approach we have taken to perform the optimization process is **Analytic Hierarchy Process (AHP)** [4]. The benefits of the AHP process are in its ability to vary weights between the importance of the objectives and also vary the weights between the choices with respect to the objectives. In this particular scenario, there are two objectives for the service provider. **Objective 1** achieves maximum resource for the traffic class, and **objective 2** maximizes the number of users for the traffic class. This advantage gives the ability to define the relational importance between different business objectives as well as the choices during the decision making process. The AHP calculation is a three-step process, and is as follows (due to space limitations, only a summary of the calculation procedure will be presented; for further information please refer to [4]).

Step 1) Calculating the objective weights: For *n* objectives, an *n x n* pair-wise comparison matrix is constructed. The value of an entry in the matrix will be determined by the importance between objectives, and is measured between 1-9. (This is part of the Business policy where the service provider will list the importance between each objective). After normalizing the matrix, where the weights are then calculated from the row average of the matrix. In this particular case, the corresponding weights for each object are: w_{obj1}=0.75, w_{obj2}=0.25.

Step 2) Calculate the choice score with respect to the objective. Construct a pair-wise comparison matrix for each objective and calculate the score of the choice (e.g. different services – EF, AF_1, AF_2) with respect to the objectives of the service provider. This will result in choice weights ($W_{EF\text{-}obj1}$, $W_{AF1\text{-}obj1}$, $W_{AF2\text{-}obj1}$), which are presented in Table 1 and 2.

Step 3) Determine the final traffic class score by calculating the sum of product of weights and choice score for each class.

The initial charges for each service was set at EF - \$5/Mb, AF_1 - \$3/Mb, and AF_2 - \$1/Mb. This resulted in the choice score shown in Table 1.

TABLE I. CHOICE SCORE FOR INITIAL PRICE STRUCTURE

	W_{EF}	W_{AF1}	W_{AF2}
Objective 1	0.68	0.25	0.07
Objective 2	0.64	0.18	0.17

After performing the calculation in step 2, this resulted in the final traffic class score of EF=0.67, AF_1=0.233, and AF_2=0.09. Calculating the ratio of 1 Gb bandwidth, this results in EF - 670Mb, AF_1 - 233 Mb, and AF_2- 90Mb. Once the bandwidth ratio is determined, the *self-configuring* policies are used to configure the network elements with the calculated bandwidth for each traffic class. After reviewing the performance for a period of time, it was determined that the multimedia application customers were under their desired target, but the web application customers were above their desired target. Therefore, the service provider decided to allocate more resources to AF_2 class by lowering the resources from the EF class. The service provider retained the importance between objectives and just changed the pricing structure. The service provider, however, noted that since the AF_2 class is attracting a large number of customers, they could increase the price for AF_2 to gain higher revenue. Therefore, the new price structure is set at EF - \$4/Mb, AF_1 - \$3/Mb, AF_2 - \$2/Mb. This triggers the *self-optimizing policy* to optimize the current bandwidth to suit the new price structure, which resulted in weights for score choice shown in Table 2.

TABLE II. CHOICE SCORE AFTER PRICING STRUCTURE CHANGE

	W_{EF}	W_{AF1}	W_{AF2}
Objective 1	0.54	0.29	0.16
Objective 2	0.68	0.2	0.12

The final class score after recalculation was EF=0.58, AF_1=0.27, and AF_2=0.15. From calculating the ratio of 1 Gb, the new bandwidth ratio that is configured at the routers are EF – 580 Mb, AF_1 – 270 Mb, and AF_2 – 150 Mb. Therefore, the scenario has demonstrated the capability of policy refinement to transform business policies to network level policies. This is achieved by embedding optimization techniques that accurately determines the correct bandwidth for each class with respect to pricing structure, importance of objectives, as well as the importance of each traffic class with respect to the objectives business policies.

VI. CONCLUSION

We have presented a policy-based autonomic framework suitable for next generation networks. The proposed solution employs the DEN-ng model, which provides an extensible means to represent both, the policy continuum as well as appropriate technology-neutral characteristics and behavior of managed entities, in a single unified framework. We have concentrated on the policy framework that is required to satisfy adaptive functionality (e.g. self-configuring, self-optimizing, self-healing, and self-protection). This policy framework provides the capability to refine high level business requirements into various forms suitable for different constituencies, including instance-specific policies to govern network element configuration changes. We have used a similar combination of information modeling, ontological engineering, and knowledge-based learning and reasoning techniques to perform conflict detection and resolution. Future work will concentrate on producing simulation and test results, along with refining our algorithms.

REFERENCES

[1] C. Prehofer, C. Bettstetter, "Self-Organization in Communication Networks: Principles and Paradigms", IEEE Communication Magazine, July 2005.

[2] G. Lanfranchi, P. Della Peruta, A. Perrone, D. Calvanese, "Toward a new landscape of systems management in an autonomic environment" IBM Systems journal, vol. 42. no. 1, 2003.

[3] D. Yagan, C.-K. Tham, "Self-Optimizing Architecture for QoS Provisioning in Differentiated Services", In Proceedings of the Second International Conference on Autonomic Computing (ICAC' 05), Seattle, WA, USA, June 2005.

[4] W. L. Winston, "Operations Research Applications and Algorithms", 2nd edition, Duxbury Press, California, USA 1991.

[5] J. O. Kephart, W. E. Walsh, "An Artificial Intelligence Perspective on Autonomic Computing Policies", In Proceedings of the 27th International Conference on Software Engineering, St. Louis, Missouri, USA, 2005.

[6] J. Strassner, "How Policy Empowers Business-Driven Device Management", In Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks (Policy' 02), Monterey, California, USA, June 2002.

[7] M. Casassa Mont, A. Baldwin, C. Goh, 1999, POWER Prototype: Towards Integrated Policy-Based Management.

[8] Arosha K. Bandara, Emil C. Lupu, Jonathan Moffett, Alessandra Russo, 2004, "A Goal-based Approach to Policy Refinement", 5th IEEE Workshop on Policies for Distributed Systems and Networks (Policy, 2004), IBM TJ Watson Research Centre, New York, USA.

[9] Shared Information/Data (SID) Model, Concepts, Principles and Domains, GB922 Release 4.5.

[10] R. Chadha, H. Cheng, Y.H. Chend, J. Chiang, A. Ghetie, G. Levin, H. Tanna, "Policy-Based Mobile Ad Hoc Network Management", In Proceedings of the Fifth IEEE Workshop on Policies for Distributed Systems and Networks (POLICY'04), New York, USA, June 2004.

[11] W. Zhang, Y.S. Gan, K.J. Loh, K.C. Chua, "Policy-Based QoS Management Architecture in an Integrated UMTS and WLAN Environment", IEEE Communications Magazine, November 2003

[12] N. Dunlop, J. Indulska, K. Raymond, "Dynamic Conflict Detection in Policy-Based Management Systems" In Proceedings of the 6th international Enterprise Distributed Object Computing Conference, IEEE Computer Society, Washington, DC, September 2002.

[13] Strassner, J., "Autonomic Networking – Theory and Practice", IEEE Tutorial, Dec 2004

[14] Kephart, J.O. and Chess, D.M., "The Vision of Autonomic Computing", Jan 2003, www.research.ibm.com/autonomic/research/papers/AC_Vision_Computer_Jan_2003.pdf

[15] http://www.motorola.com/content/0,,2364-4393,00.

[16] IBM, "An Architectural Blueprint for Autonomic Computing", April 2003

[17] Strassner, J. "Policy-Based Network Management", Morgan Kaufmann Publishers, ISBN 1-55860-859-1

[18] Moore, B., Ellesson, E., Strassner, J., Westerinen, A., "Policy Core Information Model – Version 1 Specification", RFC3060, February 2001

[19] Strassner, J. "Knowledge Management Issues for Autonomic Systems", TAKMA 2005.