End-to-End Model Driven Policy Based Network Management

Dave Raymer, John Strassner
Motorola Labs
Ft. Worth, Texas, USA
Schaumburg, IL, USA
{David.Raymer, John.Strassner}@motorola.com

Elyes Lehtihet, Sven van der Meer
Telecommunications Software and Systems Group, Waterford Institute of Technology
Waterford, Ireland
{elehtihet, vdmeer}@tssg.org

Abstract

The continued movement towards converged networks changes the focus to building application services that enable customers to move between different types of service providers based on their needs. Policy management becomes paramount for the rapid deployment and management of these application services. This paper presents the concept of a policy continuum and discusses the importance of modeling and natural languages in the presence of the policy continuum, resulting in a novel architecture suitable for autonomic computing.

1 Introduction

The focus of policy-based network management (PBNM) has been on the development of models and languages for the representation and specification of policy. While this is important, it is not sufficient to move PBNM out of the laboratory and research environments. This is because widespread commercial deployment of PBNM is dependent on the integration of a consistent and coherent approach to policy that extends from the business support systems through the network to the subscriber units at the very edges of the network. Providers require an end-to-end PBNM (E2E-PBNM) approach in order to ensure that their business policies and needs will be supported and met by the network and the applications and services it supports.

In order to enable the pervasive deployment and utilization of E2E-PBNM, a number of obstacles must be overcome. These obstacles are:

1. Definition of different constituencies that need to work together to realize E2EPBNM.
2. Creation of an information model that enables the modeling of all aspects of the network, its users, and its operational environment.
3. The creation of a means for different constituencies to use the information model to define and express policies in terms of the relevant stakeholders.
4. The ability to rapidly move from models to software in a rapid (preferably automated) fashion that provides for application, service, and network management and control.

These problems must be solved in concert, as opposed to separately, in order to develop an E2E-PBNM architecture that can scale to meet the needs of a Service Provider or large Enterprise. Then, we can consider future applications of E2E-PBNM, such as for autonomic computing [1]. For example, how can policy control the reconfiguration of a network element in the face of changing functionality of the network element, changing demands of its users, and changing environmental conditions?

The rest of this paper is organized as follows. First, we address the four shortcomings addressed above; then we apply them to a specific solution. Section 2 describes the concept of different constituencies, along with a solution to enable them working together while retaining their individual concepts and terms. Section 3 describes our approach to represent network, user, and other entities using policy via the DEN-ng model. Section 4 defines a stratified language-based approach to enable different constituencies to use the DEN-ng policy model. Section 5 describes our variation of Model-Driven Architecture, which we call MDX. Section 6 shows how policy can manage network reconfiguration and it provides a high-level description of the resulting architecture. Finally, we discuss conclusions in section 7 and provide references in the last section.

2 The Policy Continuum

The policy continuum was developed in order to enable multiple constituencies, which have different concepts and terminologies, to co-define and -develop policies. This approach was presented in [6] and [5].

Most systems define the notion of a policy as a single entity. This is incorrect. For example, there are policies to represent business rules, policies to control customer rebates, and even policies to control configuring device features. These policies use different grammars to express their function, and are used by different constituencies. However, they can in reality be different views of the same policy:

- The services and resources that a business provides to its customers are governed by one or more policies
- Services and resources are tied to products; hence the customer gets a rebate determined by policy if the provided services and/or resources do not meet their agreed, contracted needs
- Reconfiguration is used to adjust the availability and performance of services and resources.

One example of the policy continuum is shown in Figure 1 below.

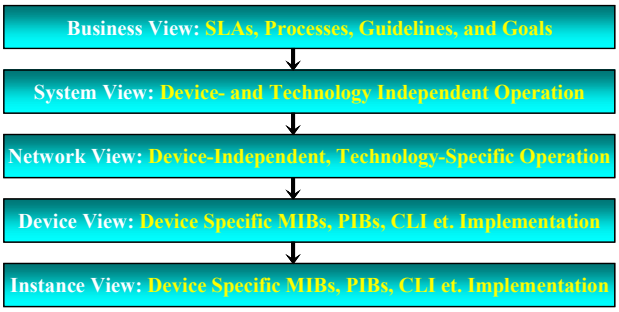

Figure 1. The Policy Continuum

Each level of the policy continuum is optimized for a different type of constituency that needs and/or uses information of a specific nature. For example, the business user wants Service Level Agreement (SLA) information, and isn't interested in the type of queuing that will be used to forward traffic. Conversely, the administrator of the network may want to develop specific commands to program the device, and may need to have a completely different representation of the policy.

DEN-ng defines a set of *views* to support the needs of different constituencies in the policy continuum. This is similar to the RM-ODP concept of viewpoints [4], except that DEN-ng defines a set of views that are strongly related to each other. This enables the needs of different constituencies to be associated with each other. For instance, a business rule can be translated into command changes to govern changes in a device configuration. The four DEN-ng views are identical to the NGOSS views [3], which enables NGOSS to be used to help attain our goals. The DEN-ng information model [7], [5] is not a single model, but rather a set of models, one per view. Each view has its own grammar and terminology, which enables each view to express the needs of a particular set of constituencies through specific semantics and structure. This enables policy to be treated as a continuum, where policies in different views are related to each other through model mappings [5] of the DEN-ng information model views. A model mapping is a translation from one type of model to another type of model. A model mapping changes the representation and/or level of abstraction used in one model to another representation and/or level of abstraction in another model. This provides a layered set of policies with different levels of abstractions, and model mappings to translate between them.

3 The DEN-ng Information Model

The network management community needs to create an information model that encompasses the entire network and its operational environment. Next generation network (NGN) management and other applications, such as autonomic computing [2] will be even more complex. One of the principal functions to model is ***change***: the functionality of a device, the needs of its users, and its operational environment.

This has been a very difficult goal to achieve. Four problems with current management models are: (1) they cannot express similar concepts used in different implementations, (2) the policy model must be independent of content, (3) they are *current state* models (models that are made up of entities that represent the current state of a managed object), and (4) they are difficult to extend to accommodate new technologies, devices, vendor-specific information, and other factors.

The DEN-ng object-oriented information model provides a cohesive, comprehensive and extensible means to categorize and represent things of interest in a managed environment, including users, policies, processes, routers, services, and anything else that needs to be represented in a common way to facilitate its representation and management. The DEN-ng information model defines the static and dynamic characteristics and behavior of these managed entities as independent of any specific type of repository, software usage, or access protocol. Note that the explicit use of *dynamic* models differentiates it from other current management efforts.

DEN-ng uses dynamic models to represent the *life cycle* of managed elements. Many different stakeholders are required to work together to build a product. However, they all have different perspectives on how the product works. This means that one concept might mean different things to different people. For example, when a business analyst looks at an SLA, that person thinks of contractual obligations and different options for realizing revenue. In con-

trast, the network technician responsible for implementing the SLA is at a loss, since the SLA does not contain the technical specifications needed by the network engineer to configure the device to support the services that are being sold. Thus, the SLA must be translated to a form that contains information suitable for the network engineer. DEN-ng made the decision that instead of trying to build a single *"Üßer-model"* that was capable of representing these different concerns, it would instead build a set of models, each focused on a different constituency, in the form of four views (business, system, implementation, and deployment).

4 The Policy Language Requirements

There are two aspects to the language problem that must be dealt with in the E2E-PBNM space. The first of these is the creation of one or more languages that can be easily and effectively used to express policy by individuals knowledgeable with regard to the business drivers and needs of a provider. Note that for maximum acceptance by the business community, this *business policy language* should enable the user to define policies using a (possibly restricted) form of natural language. The second issue is whether to use distinctly separate languages for each subsequent level within the policy continuum or dialects of the business policy language. This is an issue, because using such a language makes it difficult to translate into vendor-specific commands to reconfigure a device.

The approach outlined in this paper recommends using a base business level policy language with a number of dialects to represent policy at each layer of the policy continuum.

The business level policy language (BLPL) must be useable by business policy authors (BPAs) that are not knowledgeable with regards to the concepts of PBNM. For example, if the task of the BPA is to define the characteristics of three service offerings, called Gold, Silver, and Bronze, then the BPA should be able to do this without having to know the details of the network devices that will be used to support these services. Additionally, the BLPL must be capable of expressing policy with high levels of both specificity and simplicity.

A BLPL that is not capable of the definition of concise and precise policy will not provide the necessary level of information to enable transformations into a language, or dialect of a base language, with meaning at successively lower layers of the policy continuum.

A *business level policy language* is therefore defined as a restricted natural language based on English that is extended in terminology and expressions to enable the specification of network management policy for an end-to-end ICT network.

5 Model-Driven Everything

The OMG's Model-driven Architecture (MDA) and the DEN-ng policy continuum share some common characteristics. These similarities between the OMG's vision and our program made MDA a promising candidate to guide the design and deployment of E2E PBNM solutions.

Model-driven design, especially for network management, is driven by the semantic information available in a heterogeneous environment, e.g., having multiple constituencies that use different concepts and terminologies to define and develop policies. A "simple" MDA approach, suitable for the development of e.g. banking applications, is not sufficient.

Instead, the model-driven design needs to extend existing MDA tools towards a framework that allows:

- Define and test models for each constituency, e.g. supporting different views with specific languages and map changes automatically between the views.
- Develop a framework for processing the policy language, which means to
  - Import and explore different source of knowledge (platform independent / specific models).
  - Implement the OMG QVT (Query View and Transformation) on our language representation, and generate platform independent / specific models
- Design pattern transformations from our language to different technologies
  - Mapping from the language toward different technologies (XML Schema, DB Schema, Any format) and the tool toward different implementations (Java, C#, Any language)

6 E2E PBNM Architecture

The E2E PBNM Architecture is a component based architecture that is compatible with the TeleManagement Forum NGOSS™ Technology Neutral Architecture. The E2E PBNM Architecture adds a number of fundamental framework services to those described in [6]. These services provide integration of and access to the information contained in the ontology, knowledge and data stores, as well as the management and distribution of policy.

As can be seen in Figure 2, the E2E PBNM architecture is intended to integrate into existing network infrastructure; this provides the capability for the enterprise network operator to deploy the E2E PBNM solution in a phased pragmatic fashion. The Policy Application shown in Figure 2 is responsible for translating from vendor-specific to

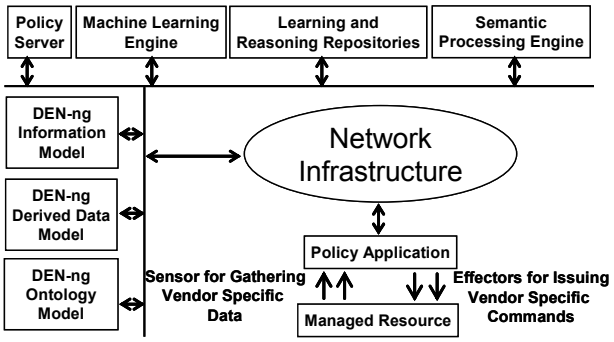

Figure 2. E2E PBM Architecture

technology-neutral forms of data, harmonizing data from multiple sources, operating on those data using elements of the policy language, and maintaining a closed loop system. A brief explanation of how this is done is as follows:

- We assume that the network is made up of heterogeneous data that may be in incompatible formats and have differing semantics; therefore, we need to harmonize the data into a common form so that it can be used by the rest of the system.
- A basic autonomic control loop consists of the actions
  - observe sensor information
  - determine the actual state of the managed element
  - compare the actual state of the managed element to its desired state
  - define an optimal path of state transitions that move the current state of the managed element back to its desired state
  - monitor the results and prove that the current state of the managed element is indeed what is expected at each reconfiguration operation.
- In order to carry out the above control loop actions, we need different type of data
  - Data models contain instance and vendor specific subsets of the information model, and hence provide data "out of context"
  - The information model contains the overall set of relationships that we know about, and hence provides the context for information from a specific data model
  - Ontologies augment information in the models with additional meaning and relationships
- We then have different applications that use these data to maintain the state of the managed element
  - Machine learning is used to constantly observe data and behavior and develop intelligence
  - Reasoning algorithms are used to post hypotheses as to why data was received, guide the system in gathering new data to support (or disprove) the hypotheses and again, increase knowledge in the system
  - The policy server guides each of the components shown in Figure 2
  - Policy plays a key role in this approach: **it controls whether an individual state transition is allowed at a given point in time and context**

7 Conclusions

This paper has presented a novel policy-based architecture for end-to-end network management. Future work will concentrate on stressing the above architecture, as well as incorporating additional reasoning capabilities (in the form of reinforcement and concept learning, as well as abductive and inductive reasoning algorithms) to give the autonomic system greater understanding of its environment. As this future work grows, we expect the role of policy to be even more pronounced.

References

[1] J. Strassner, *"Autonomic Networking - Theory and Practice"*, IM 2005 Tutorial, May 2005

[2] J. O. Kephart and D. M. Chess, *"The Vision of Autonomic Computing"*, IEEE Computer 36, No. 1, 41-50 (2003).

[3] TeleManagement Forum, *"GB927: The NGOSS^{TM} Lifecycle and Methodology"*, Nov 2004.

[4] Open Distributed Processing Reference Model - Foundations, ISO/IEC 10746-2, 1996.

[5] J. Strassner. *"Policy-based Network Management: Solutions for the Next Generation"*. Morgan-Kaufman Publishers. ISBN 1-55-859-1, Sep 2003.

[6] TeleManagement Forum, *"TMF 053: The NGOSS^{TM} Technology Neutral Architecture"* and its Addenda, 2000-2005.

[7] DEN-ng forms the foundation of the TeleManagement Forum's Shared Information and Data Model, and is partially incorporated in the following specifications: GB922 main, GB922 Addenda 1A, 1BI, 1P, 1-POL, 1R, 4SO, 4S-QoS, 5RO, 5PR, and 5LR.