Network Working Group
Internet-Draft
Intended status: Informational
Expires: May 3, 2018

A. Clemm
Futurewei Technologies, Inc.
L. Ciavaglia
Nokia
L. Granville
Federal University of Rio Grande do Sul (UFRGS)

October 30, 2017

Distinguishing Intent, Policy, and Service Models
draft-clemm-nmrg-dist-intent-00

Abstract

This document presents existing definitions of the Intent, Policy, and Service Models concepts, analyses their differences and commonalities, and how the concepts relate to one another. The document is intended to clarify the different concepts and converge towards a common and shared understanding, and then use this foundation to guide further definition of valid research and engineering problems and their solutions.

Table of Contents

1. Introduction

Traditionally in the IETF, interest with regard to management and operations has focused on the individual network features and devices. The emphasis has generally been on aspects that needed to be provided by a networking device. A prime example for this is SNMP-based management and the 200+ MIBs that have been defined by the IETF over the years. More recent examples include NETCONF, RESTCONF, and YANG data model definitions including aspects such as interface configuration, ACL configuration, or Syslog configuration. However, recent years have seen an increased interest in addressing also upper layers of management and managing end-to-end concepts. Examples include the definition of YANG models for network topology, or the explanation of service models in the context of service orchestration and controllers. In addition, this interest has been fueled by the discussion about how to manage autonomic networks as discussed in the ANIMA working group. Autonomic networks are driven by the need to lower operational expenses and make management exceptionally easy, putting it at odds with the need to manage the network one device and one feature at a time.

It has been recognized for a long time that comprehensive management solutions cannot operate only at the level of individual devices and low-level configurations. ITU-T's TMN model introduced a set of management layers as part of the TMN pyramid, consisting of network

element, network, service, and business management. High-level operational objectives would propagate in top-down fashion from upper to lower layers. The associated abstraction hierarchy was key to decompose management complexity into separate areas of concerns. This abstraction hierarchy was accompanied by an information hierarchy that concerned itself at the lowest level with device-specific information, but that would, at higher layers, include, for example, end-to-end service instances.

Accordingly, there is generally a recognition that to be able to manage networks efficiently end-to-end, management needs to be applied at higher levels of abstraction than that of low-level device details. Instead, it is required to take a more holistic view, in which device-specific details and data models are low-level artifacts that should ideally be derived from higher-level concepts. This can be achieved, for example, by higher-level systems that break down higher-level concepts (such as an instance of a service) into specific device configurations. Examples of such systems include SDN controllers or service provisioning systems. Potentially, this can be even done by intelligent devices themselves, for example, in case of autonomic nodes in an autonomic network. The goal here is that ultimately nodes are able to understand high-level concepts and automatically coordinate with other nodes to achieve the desired end-to-end behavior, without need for intermediate systems. While autonomic networks are intended to exhibit "self-management" properties, they still require input from an operator or outside system to provide operational guidance and information about the goals, purposes, and service instances that the network is to serve.

In the case of autonomic networks, the high-level guidance given to the network is commonly referred to as "Intent" [RFC7575]. The idea behind Intent is that a user provides guidance to the network, for example, communicate expectations regarding service level objectives and service instances or regarding operational goals, such as whether to optimize utilization or service levels in a certain operational context. At the same time, the user should have neither to revert to low-level configurations nor ideally have to learn a specific language of the network. Instead, the user should be able to simply express what the network should accomplish - to convey the user's intent. Ideally, the network would be able to infer the intent using a very high-level, natural language or conversational user interface. The autonomic network would be able to interpret this intent and break it down into low-level configurations as needed, even automatically coordinating between nodes to negotiate and tune behaviors. Intent would be conveyed to the autonomic network as a whole; propagation of intent among devices would occur automatically, as the user should not be concerned with individual nodes nor the instantiation of intent across these nodes.

The vision of giving the user the ability to communicate to the network in very simple, high-level terms what the network needs to provide and have the network do the rest seems like the Holy Grail of intuition and ease-of-use for how to interact with a network. Accordingly, the term "intent" has caught on rapidly and spread like wildfire, being rapidly adopted also by SDN controllers and by management solutions, all proclaiming the higher-level abstractions exposed of their own northbound interfaces as "intent". However, somewhat overlooked in all this is the fact that, as mentioned above, the concept of management or control abstraction hierarchies was not invented with Intent. This concept has been known for a long time and variations of this concept incarnated in different forms in the past. Specifically:

> Policy-based management has the goal of defining high-level policies that are subsequently translated and rendered into lower-level actions and parameter settings at devices. Policies are frequently defined in terms of rules, consisting of events (that trigger a rule), conditions (that are assessed when the rule is triggered), and actions (that are carried out when the condition holds). However, many different categories of policies and ways to define them exist, ranging from ACL-style matching rules to high-level declarative policy languages such as Ponder. [REFERENCES]
>
> Service models define end-to-end service instances, which are in turn mapped onto low-level configurations (often expressed via device data models) that are applied across devices and resources in the network.

This raises the question in which ways intent, policy, and service models are really different. Terms that are trending can become easily overloaded and may end up being used as synonyms for terms that had already been well-established before, not just for the new and differentiating concept for which they were introduced. In order to avoid this situation, this document aims to provide a clear distinction between these terms and what they represent. Specifically, it aims to answer the question whether "intent" is just a new term for "policy" (or "service model") or whether it represents something genuinely different.

2. Key Words

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Definitions and Acronyms

ACL: Access Control List

Intent: An abstract, high-level policy used to operate a network [RFC7575].

Policy: A rule, or set of rules, that governs the choices in behavior of a system.

PDP: Policy Decision Point

PEP: Policy Enforcement Point

Service Model: A model that represents a service that is provided by a network to a user.

4. Introduction of Concepts

The following subsections provide an overview of the concepts of service models, of policies respectively policy-based management, and of intent respectively intent-based management. While the descriptions are intentionally kept brief and do not provide detailed tutorials, they should convey the bigger picture of the purpose of each concept and provide a sense where those concepts are similar and where they differ. With this background, the differences between them are summarized in the subsequent section.

4.1. Service Models

A service model is a model that represents a service that is provided by a network to a user. An example of a service could be a Layer 3 VPN service, a Network Slice, or residential Internet access. Service models represent service instances as entities in their own right. Services have their own parameters, actions, and lifecycles. Typically, service instances can be bound to end users, who might be billed for the service.

Instantiating a service typically involves multiple aspects:

- Resources need to be allocated, such as IP addresses, interfaces, bandwidth, or memory.

- How to map services to the resources needs to be defined. Multiple mappings are often possible, which to select may depend on context (such as which type of access is available to connect the end user with the service).

- Bindings need to be maintained between upper- and lower-level objects.

They involve a system, such as a controller, that provides provisioning logic. Orchestration itself is conducted using a "push" model, in which the controller/manager initiates the operations as required, pushing down the specific configurations to the device. The device itself is typically agnostic to the service or the fact that its resources or configurations are part of a service/concept at a higher layer.

Instantiated service models map to instantiated lower-layer models. Examples include instances of paths, or instances of specific port configurations. The service model typically also models dependencies and layering of services over lower-layer networking resources that are used to provide services. This facilitates management by allowing to follow dependencies for troubleshooting activities, to perform impact analysis in which events in the network are assessed regarding their impact on services and customers Services are typically orchestrated or provisioned top-to-bottom, and to keep track of the assignment of network resources.

4.2. Policy and Policy-Based Management

Policy-based management (PBM) is a management paradigm that separates the rules that govern the behavior of a system from the functionality of the system. It promises to reduce maintenance costs of information and communication systems while improving flexibility and runtime adaptability. It is today present at the heart of a multitude of management architectures and paradigms including SLA-driven, Business-driven, autonomous, adaptive, and self-* management [Boutaba07]. The interested reader is asked to refer to the rich set of existing literature which includes this and many other references. In the following, we an only provide a much-abridged and distilled overview.

At the heart of policy-based management is the concept of a policy. Multiple definitions of policy exist: "Policies are rules governing the choices in behavior of a system" [Sloman94]. "Policy is a set of rules that are used to manage and control the changing and/or maintaining of the state of one or more managed objects" [Strassner03]. Common to most definitions is the definition of a policy as a "rule". Typically, rules follow consists of events (whose occurrence triggers a rule), conditions (that get assessed before any actions are actually "fired"), and actions that are carried out when the condition holds.

Policy-based management can be considered an imperative management paradigm: Policies specify precisely what needs to be done when. Using policies, management can in effect be defined as a set of simple control loops. This makes policy-based management a suitable technology to implement autonomic behavior that can exhibit self-* management properties including self-configuration, self-healing, self-optimization, and self-protection. In effect, policies define simple control loops typically used to define management as a set of simple control loops.

Policies typically involve a certain degree of abstraction in order to cope with heterogeneity of networking devices. Rather than having a device-specific policy that defines events, conditions, and actions in terms of device-specific commands, parameters, and data models, policy is defined at a higher-level of abstraction involving a canonical model of systems and devices to which the policy is to be applied. A policy agent on the device subsequently "renders" the policy, i.e., translates the canonical model into a device-specific representation. This concept allows to apply the same policy across a wide range of devices, which leads to operational scale and allows network operators and authors of policies to think in higher terms of abstraction than device specifics.

Policy-based management is typically "push-based": Policies are pushed onto devices where they are rendered and enforced. The push operations are conducted by a manager or controller, which is responsible for deploying policies across the network and monitor their proper operation. That said, other policy architectures are possible. For example, policy-based management can also include a pull-component in which the decision regarding which action to take is delegated to a so-called Policy Decision Point (PDP). This PDP can reside outside the managed device itself and has typically global visibility and context with which to make policy decisions. Whenever a network device observes an event that is associated with a policy, but lacks the full definition of the policy or the ability to reach a conclusion regarding the expected action, it reaches out to the PDP for a decision. Subsequently, the device carries out the decision as returned by the PDP - the device "enforces" the policy and hence acts as a PEP (Policy Enforcement Point). Either way, PBM architectures typically involve a central component from which policies are deployed across the network, and/or policy decisions served.

4.3. Intent and Intent-Based Management

In the context of Autonomic Networks, Intent is defined in as "an abstract, high-level policy used to operate a network". According to this definition, an intent is a specific type of policy. However, to avoid using "intent" simply as a synonym for "policy, a clearer

distinction needs to be introduced that distinguishes intent clearly from other types of policies.

Autonomic networks are expected to "self-manage" and operate with minimal outside intervention. However, autonomic networks are not clairvoyant and have no way of automatically knowing particular operational goals nor what instances of networking services to support. In other words, they do not know what the "intent" of the network provider is that gives the network the purpose of its being. This still needs to be communicated by what informally constitutes "intent".

More specifically, intent is a declaration of high-level operational goals or services that are to be provided by the network, without specifying how to achieve them. Those goals are defined in a manner that is purely declarative - they specify what to accomplish or what the desired outcome for the network operator is, not how to achieve it or even when to spring into action. In addition, Intent (at least in an Autonomic Network) should be rendered by network devices themselves, i.e., translated into device specific rules and courses of action. It should not be orchestrated or broken down by a higher-level, centralized system. Intent holds for the network as a whole, not individual devices, and is hence automatically disseminated across all devices in the network, which can themselves decide whether they need to act on it. This facilitates management even further, since it obviates the need for a higher-layer system to break down and decompose higher-level intent, and because there is no need to even discover and maintain an inventory of the network to be able to manage it.

Intent thus constititutes declarative policy with a network-wide scope. A human operator defines 'what' is expected, and the network computes a solution meeting the requirements. This computation can occur in distributed or even decentralized fashion by auonomic functions that reside on network nodes.

Other definitions of intent exist such as [ONF TR 523] and will be investigated in future revisions of this document. Likewise, some definitions of intent allow for the presence of a centralized function that renders the intent into lower-level policies or instructions and orchestrates them across the network. While to the end user the concept of "intent" appears the same regardless of its method of rendering, this interpretation opens a slippery slope of how to clearly distinguish "intent" from other higher-layer abstractions. Again, these notions will be further investigated in future revisions of this document and in collaboration with NMRG.

5. Distinguishing between Intent, Policy, and Service Models

What Intent, Policy, and Service Models all have in common is the fact that they involve a higher-layer of abstraction of a network that does not involve device-specifics, that generally transcends individual devices, and that makes the network easier to manage for applications and human users compared to having to manage the network one device at a time. Beyond that, differences emerge. Service models have less in common with policy and intent than policy and intent do with each other.

Summarized differences:

- Service model is a data model with dependencies onto lower models. It requires orchestration by a system; the logic to orchestrate/manage/provide the service model is not included as part of the model itself.

- Policy is a set of rules, typically modeled around a variation of events/conditions/actions, used to express simple control loops that can be rendered by devices themselves, without requiring intervention by outside system.

- Intent is a higher-level declarative policy that operates at the level of a network, not individual devices. It is used to define operational goals and desired outcomes without the need to enumerate specific events, conditions, and actions. Intent is rendered by the network itself; also the dissemination of intent across the network and any required coordination between nodes is resolved by the network itself without the need for outside systems.

6. IANA Considerations

Not applicable

7. Security Considerations

Not applicable

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7575] Behringer, M., Pritikin, M., Bjarnason, S., Clemm, A., Carpenter, B., Jiang, S., and L. Ciavaglia, "Autonomic Networking: Definitions and Design Goals", RFC 7575, DOI 10.17487/RFC7575, June 2015, <https://www.rfc-editor.org/info/rfc7575>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

8.2. Informative References

[Boutaba07]
Boutaba, R. and I. Aib, "Policy-Based Management: A Historical perspective. Journal of Network and Systems Management (JNSM), Springer, Vol. 15 (4).", December 2007.

[Sloman94]
Sloman, M., "Policy Driven Management for Distributed Systems. Journal of Network and Systems Management (JNSM), Springer, Vol. 2 (4).", December 1994.

[Strassner03]
Strassner, J., "Policy-Based Network Management. Elsevier.", 2003.

Authors' Addresses

Alexander Clemm
Futurewei Technologies, Inc.
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: ludwig@clemm.org

Laurent Ciavaglia
Nokia
Route de Villejust
Nozay 91460
FR

Email: laurent.ciavaglia@nokia.com

Lisandro Zambenedetti Granville
Federal University of Rio Grande do Sul (UFRGS)
Av. Bento Goncalves
Porto Alegre 9500
BR

Email: granville@inf.ufrgs.br