Network Working Group
Internet-Draft
Intended status: Informational
Expires: March 19, 2021

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

September 15, 2020

Intent-Based Networking - Concepts and Definitions
draft-irtf-nmrg-ibn-concepts-definitions-02

Abstract

Intent and Intent-Based Networking (IBN) are taking the industry by storm. At the same time, those terms are used loosely and often inconsistently, in many cases overlapping and confused with other concepts such as "Policy". This document clarifies the concept of "Intent" and provides an overview of functionality that is associated with it. The goal is to contribute towards a common and shared understanding of terms, concepts, and functionality that can be used as foundation to guide further definition of associated research and engineering problems and their solutions.

Table of Contents

1. Introduction

Traditionally in the IETF, interest regarding management and operations has focused on individual network and device features. Standardization emphasis has generally been put on management instrumentation that needed to be provided to a networking device. A

prime example of this is SNMP-based management and the 200+ MIBs that have been defined by the IETF over the years. More recent examples include YANG data model definitions for aspects such as interface configuration, ACL configuration, or Syslog configuration.

There is a sense and reality that in modern network environments managing networks by configuring myriads of "nerd knobs" on a device-by-device basis is no longer sustainable. Significant challenges arise with keeping device configurations not only consistent across a network, but consistent with the needs of services and service features they are supposed to enable. Adaptability to changes at scale is a fundamental property of a well-designed IBN system, that requires the ability to consume and process analytics that is context/intent aware at near real-time speeds. At the same time, operations need to be streamlined and automated wherever possible to not only lower operational expenses, but also allow for rapid reconfiguration of networks at sub-second time scales and to ensure that networks are delivering their functionality as expected.

Accordingly, the IETF has begun to address end-to-end management aspects that go beyond the realm of individual devices in isolation. Examples include the definition of YANG models for network topology [RFC8345] or the introduction of service models used by service orchestration systems and controllers [RFC8309]. Much 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 desire to lower operational expenses and make the management of the network as a whole more straightforward, putting it at odds with the need to manage the network one device and one feature at a time. However, 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.

This vision has since caught on with the industry in a big way, leading to a significant number of solutions that offer "Intent-based management" that promise network providers to manage networks holistically at a higher level of abstraction and as a system that happens to consist of interconnected components, as opposed to a set of independent devices (that happen to be interconnected). Those offerings include IBN systems (offering full a life-cycle of intent), SDN controllers (offering a single point of control and administration for a network), and network management and Operations Support Systems (OSS).

However, 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. In this sense, the vision of "Intent" is not entirely new. In the past, ITU-T's model of a Telecommunications Management Network, TMN, introduced a set of management layers that defined a management hierarchy, consisting of network element, network, service, and business management. High-level operational objectives would propagate in a top-down fashion from upper to lower layers. The associated abstraction hierarchy was crucial 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. Similarly, the concept of "Policy-based Network Management (PBNM)" has, for a long time, touted the ability to allow users to manage networks by specifying high-level management policies, with policy systems automatically "rendering" those policies, i.e., breaking them down into low-level configurations and control logic.

What has been missing, however, is putting these concepts into a more current context and updating them to account for current technology trends. This document clarifies the concepts behind intent. It differentiates it from related concepts. It also provides an overview of first-order principles of Intent-Based Networking as well as associated functionality. The goal is to contribute to a common and shared understanding that can be used as a foundation to articulate research and engineering problems in the area of Intent-Based Networking. It should be noted that the articulation of those problems is beyond this document's scope.

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

API: Application Programming Interface

Intent: A set of operational goals that a network should meet and outcomes that a network is supposed to deliver, defined in a declarative manner without specifying how to achieve or implement them.

IBA: Intent-Based Analytics - Analytics that are defined and derived from users' intent and used to validate the intended state.

IBN: Intent-Based Network, a network that can be managed using intent.

IBS: Intent-Based System, a system that supports management functions that can be guided using intent.

Policy: A 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.

SSoT: Single Source of Truth - A functional block in an IBN system that normalizes users' intent and serves as the single source of data for the lower layers.

4. Introduction of Concepts

The following section provides an overview of the concept of Intent and Intent-Based Management. It also provides an overview of the related concepts of service models, and of policies respectively Policy-Based Network Management, and explains how they relate to Intent and Intent-Based Management.

4.1. Intent and Intent-Based Management

The term "Intent" was first introduced in the context of Autonomic Networks, where it is defined as "an abstract, high-level policy used to operate a network" [RFC7575]. According to this definition, an Intent is a specific type of policy, provided by a user to provide guidance to the Autonomic Network that would otherwise operate without human intervention. However, to avoid using "Intent" simply as a synonym for "Policy", a distinction needs to be introduced that differentiates Intent clearly from other types of policies.

For one, while Intent-Based Management aims to lead towards networks that are dramatically simpler to manage and operate requiring only minimal outside intervention, the concept of "Intent" is not limited to autonomic networks, but applies to any network. Networks, even when considered "autonomic", 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 operational goals that a network should meet and outcomes that the network is supposed to deliver, without specifying how to achieve them. Those goals and outcomes are defined in a manner that is purely declarative - they specify what to accomplish, not how to achieve it. "Intent" thus applies several important concepts simultaneously:

- It provides data abstraction: Users and operators do not need to be concerned with low-level device configuration and nerd knobs.

- It provides functional abstraction from particular management and control logic: Users and operators do not need to be concerned even with how to achieve a given Intent. What is specified is a desired outcome, with the Intent-based system automatically figuring out a course of action (e.g., a set of rules (thus, a set of rules is not part of an intent but rather derived from that intent), an algorithm) for how to achieve the outcome.

The following are some examples of intent:

- "Steer networking traffic originating from endpoints in one geography away from a second geography, unless the destination lies in that second geography."

- "Avoid routing networking traffic originating from a given set of endpoints (or associated with a given customer) through a particular vendor's equipment, even if this occurs at the expense of reduced service levels."

- "Maximize network utilization even if it means trading off service levels (such as latency, loss), unless service levels have deteriorated 20% or more from their historic mean."

- "VPN service must have path protection at all times for all paths."

- "Generate in-situ OAM data and network telemetry across for later offline analysis whenever significant fluctuations in latency across a path are observed."

In an autonomic network, intent should be rendered by the network itself, i.e., translated into device-specific rules and courses of

action. Ideally, it should not even be orchestrated or broken down by a higher-level, centralized system, but by the network devices themselves using a combination of distributed algorithms and local device abstraction. In this idealized vision, because intent holds for the network as a whole, intent should ideally be automatically disseminated across all devices in the network, which can themselves decide whether they need to act on it.

However, such decentralization will not be practical in all cases. Certain functions will need to be at least conceptually centralized. For example, users may require a single conceptual point of interaction with the network. Likewise, the vast majority of network devices may be intent-agnostic and focus only (for example) on the actual forwarding of packets. This implies that certain intent functionality needs to be provided by functions that are specialized for that purpose (which depending on scenario may be hosted on dedicated systems, or cohosted with other networking functions). For example, functionality to translate intent into courses of actions and algorithms to achieve desired outcomes may need to be provided by such specialized functions. Of course, to avoid single points of failure, the implementation and hosting of those functions may still itself be distributed, even if conceptually centralized.

Accordingly, an intent-based network is a network that can be managed using intent. This means, it is able to recognize and ingest intent of an operator, or user, and configure and adapt itself autonomously according to the user intent, achieving an intended outcome (i.e., a desired state or behavior) without requiring the user to specify the detailed technical steps for how to achieve the outcome. Instead, the intent-based network will be able to figure out on its own how to achieve the outcome.

Other definitions of intent exist, such as [TR523]. Intent there is simply defined as a declarative interface that is typically provided by a controller. It implies the presence of a centralized function that renders the intent into lower-level policies or instructions and orchestrates them across the network. While this is certainly one way of implementation, the definition presented here is narrower in the sense that it emphasizes the importance of managing the network by specifying desired outcomes without the specific steps to be taken in order to achieve the outcome. A controller API that simply provides a network-level of abstraction would not necessarily qualify as intent. Likewise, ingestion and recognition of intent may not necessarily occur via a traditional API, but may involve other types of human-machine interactions.

4.2. Related Concepts

4.2.1. Service Models

A service model is a model that represents a service that is provided by a network to a user. Per [RFC8309], a service model describes a service and its parameters in a portable/implementation-agnostic way that can be used independently of the equipment and operating environment on which the service is realized. Two subcategories are distinguished: a "Customer Service Model" describes an instance of a service as provided to a customer, possibly associated with a service order. A "Service Delivery Model" describes how a service is instantiated over existing networking infrastructure.

An example of a service could be a Layer 3 VPN service [RFC8299], 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 life-cycles. Typically, service instances can be bound to end-users, who might be billed for the service.

Instantiating a service typically involves multiple aspects:

- A user (or northbound system) needs to define and/or request a service to be instantiated.

- Resources need to be allocated, such as IP addresses, AS numbers, VLAN or VxLAN pools, 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.

- Once instantiated, the service needs to be validated and assured to ensure that the network indeed delivers the service as requested.

They involve a system, such as a controller, that provides provisioning logic. This includes breaking down high-level abstractions into lower-level device abstractions, identifying and allocating system resources, and orchestrating individual provisioning steps. Orchestration operations are generally conducted using a "push" model in which the controller/manager initiates the operations as required, then pushes down the specific configurations

to the device. In addition to instantiating and creating new instances of a service, updating, modifying, and decommissioning services need to be also supported. The device itself typically remains 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 network and device 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 and provisioned top-to-bottom, which also facilitates keeping track of the assignment of network resources. Service models might also be associated with other data that does not concern the network but provides business context. This includes things such as customer data (such as billing information), service orders and service catalogs, tariffs, service contracts, and Service Level Agreements (SLAs), including contractual agreements regarding remediation actions.

[I-D.ietf-teas-te-service-mapping-yang] is an example of a data model that provides a mapping for customer service models (e.g., the L3VPN Service Model) to Traffic Engineering (TE) models (e.g., the TE Tunnel or the Abstraction and Control of Traffic Engineered Networks Virtual Network model)

Like intent, service models provide higher layers of abstraction. Service models are often also complemented with mappings that capture dependencies between service and device or network configurations. Unlike intent, service models do not allow to define a desired "outcome" that would be automatically maintained by the intent system. Instead, the management of service models requires the development of sophisticated algorithms and control logic by network providers or system integrators.

4.2.2. Policy and Policy-Based Network Management

Policy-Based Network Management (PBNM) 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 present today 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 will 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 the 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, the definition of a rule consists of an event (whose occurrence triggers a rule), a set of conditions (which get assessed and which must be true before any actions are actually "fired"), and finally a set of one or more actions that are carried out when the condition holds.

Policy-based management can be considered an imperative management paradigm: Policies precisely specified what needs to be done when and in which circumstance. By 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 management as a set of simple control loops.

Policies typically involve a certain degree of abstraction in order to cope with the 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, a 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 a controller or the device subsequently "renders" the policy, i.e., translates the canonical model into a device-specific representation. This concept allows applying the same policy across a wide range of devices without needing to define multiple variants. In other words - policy definition is de-coupled from policy instantiation and policy enforcement. This enables operational scale and allows network operators and authors of policies to think in higher terms of abstraction than device specifics and be able to reuse the same, high-level definition across different networking domains, WAN, DC, or public cloud.

PBNM 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 being 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 (reached, for example, by deciding on an action based on various conditions). 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, PBNM architectures typically involve a central component from which policies are deployed across the network, and/or policy decisions served.

Like Intent, policies provide a higher layer of abstraction. Policy systems are also able to capture dynamic aspects of the system under management through the specification of rules that allow defining various triggers for specific courses of actions. Unlike intent, the definition of those rules (and courses of actions) still needs to be articulated by users. Since the intent is unknown, conflict resolution within or between policies requires interactions with a user or some kind of logic that resides outside of PBM. In that sense, policy constitutes a lower level of abstraction than intent, and it is conceivable for Intent-Based Systems to generate policies that are subsequently deployed by a PBM, allowing PBM to support Intent-Based Networking.

4.2.3. 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:

- A service model is a data model that is used to describe instances of services that are provided to customers. A service model has dependencies on lower-level models (device and network models) when describing how the service is mapped onto underlying network and IT infrastructure. Instantiating a service model requires

orchestration by a system; the logic for how to orchestrate/manage/provide the service model, and how to map it onto underlying resources, 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, without requiring intervention by the outside system. Policy lets users define what to do under what circumstances, but it does not specify the desired outcome.

- Intent is a high-level, declarative goal that operates at the level of a network and services it provides, not individual devices. It is used to define outcomes and high-level operational goals, without specifying how those outcomes and should be achieved or how goals should specifically be satisfied, and without the need to enumerate specific events, conditions, and actions. Which algorithm or rules to apply can be automatically "learned/derived from intent" by the intent system. In the context of autonomic networking, intent is ideally rendered by the network itself; also, the dissemination of intent across the network and any required coordination between nodes is resolved by the network without the need for external systems.

One analogy to capture the difference between policy and intent systems is that of Expert Systems and Learning Systems in the field of Artificial Intelligence. Expert Systems operate on knowledge bases with rules that are supplied by users, analogous to policy systems whose polies are supplied by users. They are able to make automatic inferences based on those rules, but are not able to "learn" new rules on their own. Learning Systems (popularized by deep learning and neural networks), on the other hand, are able to learn without depending on user programming or articulation of rules. However, they do require a learning or training phase, and explanations of actions that the system actually takes provide a different set of challenges. Analogous to intent-based systems, users focus on what they would like the learning system to accomplish, but not how to do it.

5. Principles

The following main operating principles allow characterizing the intent-based/-driven/-defined nature of a system.

1. Single Source of Truth (SSoT) and Single Version/View of Truth (SVoT). The SSoT is an essential component of an intent-based system as it enables several important operations. The set of validated intent expressions is the system's SSoT. SSoT and the

records of the operational states enable comparing the intended state and actual state of the system and determining drift between them. SSoT and the drift information provide the basis for corrective actions. If the intent-based is equipped with prediction capabilities or means, it can further develop strategies to anticipate, plan, and pro-actively act on the diverging trends with the aim to minimize their impact. Beyond providing a means for consistent system operation, SSoT also allows for better traceability to validate if/how the initial intent and associated business goals have been properly met, to evaluate the impacts of changes in the intent parameters and impacts and effects of the events occurring in the system. Single Version (or View) of Truth derives from the SSoT and can be used to perform other operations such as query, poll, or filter the measured and correlated information to create so-called "views". These views can serve the operators and/or the users of the intent-based system. To create intents as single sources of truth, the intent-based system must follow well-specified and well-documented processes and models. In other contexts [Lenrow15], SSoT is also referred to as the invariance of the intent.

2. One-touch but not one-shot. In an ideal intent-based system, the user expresses its intents in one form or another, and then the system takes over all subsequent operations (one-touch). A zero-touch approach could also be imagined in the case where the intent-based system has the capabilities or means to recognize intentions in any form of data. However, the zero- or one-touch approach should not be mistaken the fact that reaching the state of a well-formed and valid intent expression is not a one-shot process. On the contrary, the interfacing between the user and the intent-based system could be designed as an interactive and iterative process. Depending on the level of abstraction, the intent expressions will initially contain more or less implicit parts, and unprecise or unknown parameters and constraints. The role of the intent-based system is to parse, understand, and refine the intent expression to reach a well-formed and valid intent expression that can be further used by the system for the fulfillment and assurance operations. An intent refinement process could use a combination of iterative steps involving the user to validate the proposed refined intent and to ask the user for clarifications in case some parameters or variables could not be deduced or learned by the means of the system itself. In addition, the Intent-Based System will need to moderate between conflicting intent, helping users to properly choose between intent alternatives that may have different ramifications.

3. Autonomy and Supervision. A desirable goal for an intent-based system is to offer a high degree of flexibility and freedom on both the user side and system side, e.g., by giving the user the ability to express intents using its own terms, by supporting different forms of expression of intents and being capable of refining the intent expressions to well-formed and exploitable expressions. The dual principle of autonomy and supervision allows to operate a system that will have the necessary levels of autonomy to conduct its tasks and operations without requiring intervention of the user and taking its own decisions (within its areas of concern and span of control) as how to perform and meet the user expectations in terms of performance and quality, while at the same time providing the proper level of supervision to satisfy the user requirements for reporting and escalation of relevant information.

4. Learning. An intent-based system is a learning system. By contrast to the imperative type of system, such as Event-Condition-Action policy rules, where the user defines beforehand the expected behavior of the system to various events and conditions, in an intent-based system, the user only declares what the system should achieve and not how to achieve these goals. There is thus a transfer of reasoning/rationality from the human (domain knowledge) to the system. This transfer of cognitive capability also implies the availability in the intent-based system of capabilities or means for learning, reasoning, and knowledge representation and management. The learning abilities of an intent-based systems can apply to different tasks such as optimization of the intent rendering or intent refinement processes. The fact that an intent-based system is a continuously evolving system creates the condition for continuous learning and optimization. Other cognitive capabilities such as planning can also be leveraged in an intent-based system to anticipate or forecast future system state and response to changes in intents or network conditions and thus elaboration of plans to accommodate the changes while preserving system stability and efficiency in a trade-off with cost and robustness of operations. Cope with unawareness of users (smart recommendations).

5. Capability exposure. Capability exposure consists in the need for expressive network capabilities, requirements, and constraints to be able to compose/decompose intents and map the user's expectations to the system capabilities.

6. Abstract and outcome-driven. Users do not need to be concerned with how intent is achieved and are empowered to think in terms of outcomes. In addition, they do can refer to concepts at a

higher level of abstractions, independent e.g. of vendor-specific renderings.

The described principles are perhaps the most prominent, but they are not an exhaustive list. There are additional aspects to consider, such as:

- Intent targets are not individual devices but typically aggregations (such as groups of devices adhering to a common criteria, such as devices of a particular role) or abstractions (such as service types, service instances, topologies)

- Abstraction and inherent virtualization: agnostic to implementation details

- Human-tailored network interaction: IBN SHOULD speak the language of the user as opposed to requiring the user to speak the language of the device/network

- Explainability as an important IBN function, detection and IBN-aided resolution of conflicting intent, reconciliation of what the user wants and what the network can actually do

- Inherent support, verification, and assurance of trust

All of these principles and considerations have implications on the design of intent-based systems and their supporting architecture and need to be considered when deriving functional and operational requirements.

6. Intent-Based Networking - Functionality

Intent-Based Networking involves a wide variety of functions which can be roughly divided into two categories:

- Intent Fulfillment provides functions and interfaces that allow users to communicate intent to the network, and that perform the necessary actions to ensure that intent is achieved. This includes algorithms to determine proper courses of action and functions that learn to optimize outcomes over time. In addition, it also includes more traditional functions such as any required orchestration of coordinated configuration operations across the network and rendering of higher-level abstractions into lower-level parameters and control knobs.

- Intent Assurance provides functions and interfaces that allow users to validate and monitor that the network is indeed adhering to and complying with intent. This is necessary to assess the

effectiveness of actions taken as part of fulfillment, providing important feedback that allows those functions to be trained or tuned over time to optimize outcomes. In addition, Intent Assurance is necessary to address "intent drift". Intent drift occurs when a system originally meets the intent, but over time gradually allows its behavior to change or be affected until it no longer does, or does so in a less effective manner.

The following sections provide a more comprehensive overview of those functions.

6.1. Intent Fulfillment

Intent fulfillment is concerned with the functions that take intent from its origination by a user (generally, an administrator of the responsible organization) to its realization in the network.

6.1.1. Intent Ingestion and Interaction with Users

The first set of functions is concerned with "ingesting" intent, i.e. obtaining intent through interactions with users. They provide functions that recognize intent from interaction with the user as well as functions that allow users to refine their intent and articulate it in such ways so that it becomes actionable by an Intent-Based System. Typically, those functions go beyond a traditional API, although they may include APIs provided for interactions with other machines. They may support unconventional human-machine interactions, in which a human will not simply give simple commands, but which may involve a human-machine dialog to provide clarifications, to explain ramifications and trade-offs, and to facilitate refinements. The goal is ultimately to make intent-based systems as easy and natural to use as possible, allowing the user to interact with the Intent-Based System in ways that does not involve a steep learning curve forcing the user to learn the "language" of the system

6.1.2. Intent Translation

A second set of functions needs to translate user intent into courses of actions and requests to take against the network, which will be meaningful to network configuration and provisioning systems. These functions lie at the core of Intent-Based Systems, bridging the gap between interaction with users on one hand and the traditional management and operations side that will need to orchestrate provisioning and configuration across the network.

Beyond merely breaking down a higher layer of abstraction (intent) into a lower layer of abstraction (policies, device configuration),

Intent Translation functions can be complemented with functions and algorithms that perform optimizations and that are able to learn and improve over time in order to result in the best outcomes, specifically in cases where multiple ways of achieving those outcomes are conceivable. For example, satisfying an intent may involve computation of paths and other parameters that need will need to be configured across the network. Heuristics and algorithms to do so may evolve over time to optimize outcomes which may depend a myriad of dynamic network conditions and context.

6.1.3. Intent Orchestration

A third set of functions deals with the actual configuration and provisioning steps that need to be orchestrated across the network and that were determined by the previous intent translation step.

6.2. Intent Assurance

Assurance is concerned with the functions that are necessary to ensure that the network indeed complies with the desired intent once it has been fulfilled.

6.2.1. Monitoring

A first set of assurance functions monitors and observes the network and its exhibited behavior. This includes all the usual assurance functions such as monitoring the network for events and performance outliers, performing measurements to assess service levels that are being delivered, generating and collecting telemetry data. Monitoring and observation are required as basis for the next set of functions that assess whether the observed behavior is in fact in compliance with the behavior that is expected based on the intent.

6.2.2. Intent Compliance Assessment

At the core of Intent Assurance are functions that compare the actual network behavior that is being monitored and observed with the intended behavior that is expected per the intent. These functions continuously assess and validate whether the observation indicates compliance with intent. This includes assessing the effectiveness of intent fulfillment actions, including verifying that the actions had the desired effect and assessing the magnitude of the effect as applicable. It can also include functions that perform analysis and aggregation of raw observation data. The results of the assessment can be fed back to facilitate learning functions that optimize outcomes.

Intent compliance assessment also includes assessing whether intent drift occurs over time. Intent drift can be caused by control plane or lower-level management operations that inadvertently cause behavior changes which conflict with intent which was orchestrated earlier. Intent-Based Systems and Networks need to be able to detect when such drift occurs or is about to occur.

6.2.3. Intent Compliance Actions

When intent drift occurs or network behavior is inconsistent with desired intent, functions that are able to trigger corrective actions are needed. This includes actions needed to resolve intent drift and bring the network back into compliance. Alternatively and where necessary, reporting functions need to be triggered that alert operators and provide them with the necessary information and tools to react appropriately, e.g. by helping them articulate modifications to the original intent to moderate between conflicting concerns.

6.2.4. Abstraction, Aggregation, Reporting

The outcome of Intent Assurance needs to be reported back to the user in ways that allows the user to relate the outcomes to their intent. This requires a set of functions that are able to analyze, aggregate, and abstract the results of the observations accordingly. In many cases, lower-level concepts such as detailed performance statistics and observations related to low-level settings need to be "up-leveled" to concepts the user can relate to and take action on.

The required aggregation and analysis functionality needs to be complemented with functions that report intent compliance status and provide adequate summarization and visualization to the user.

7. Life-cycle

Intent is subject to a life-cycle: it comes into being, may undergo changes over the course of time, and may at some point be retracted. This life-cycle is closely tied to various interconnection functions that are associated with the intent concept.

Figure 1 depicts an intent life-cycle and its main functions. The functions were introduced in Section 6 and are divided into two functional (horizontal) planes, reflecting the distinction between fulfillment and assurance. In addition, they are divided into three (vertical) spaces.

The spaces indicate the different perspectives and interactions with different roles that are involved in addressing the functions:

- The user space involves the functions that interface the network and intent-based system with the human user. It involves the functions that allow users to articulate and the intent-based system to recognize that intent. It also involves the functions that report back the status of the network relative to the intent and that allow users to assess whether their intent has the desired effect.

- The translation or Intent-Based System (IBS) space involves the functions that bridge the gap between intent users and network operations. This includes the functions used to translate an intent into a course of action, the algorithms used to plan and optimize those courses of actions also in consideration of feedback, the functions to analyze and abstract observations to validate compliance with the intent and take corrective actions as necessary.

- The Network Operations space, finally, involves the traditional orchestration, configuration, monitoring, and measurement functions, which are used to effectuate the rendered intent and observe its effects on the network.

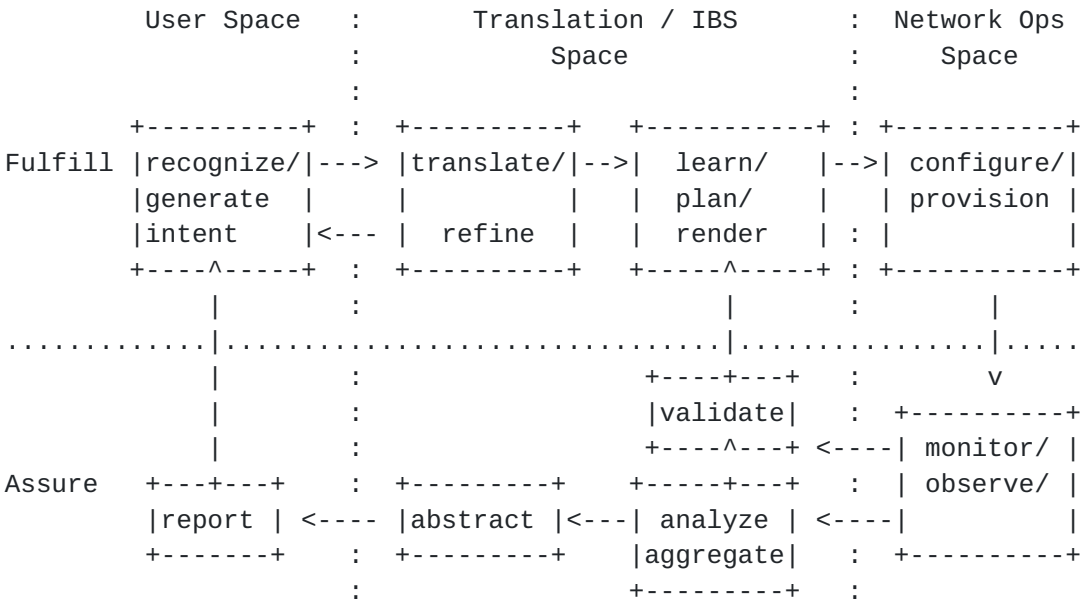

Figure 1: Intent Life-cycle

When carefully inspecting the diagram, it becomes apparent that the intent life-cycle, in fact, involves two cycles, or loops:

- The "inner" intent control loop between IBS and Network Operations space is completely autonomic and does not involve any human in the loop. It involves automatic analysis and validation of intent based on observations from the network operations space. Those observations are fed into the function that plans the rendering of networking intent in order to make adjustments as needed in the configuration of the network.

- The "outer" intent control loop involves the user space and includes the user taking action and adjusting their intent based on feedback from the IBS.

8. Additional Considerations

Given the popularity of the term "intent", its use could be broadened to encompass also known or related concepts, resulting in "intent-washing" that paints those concepts in a new light by simply applying new intent terminology to them. However, in some cases, this actually makes sense not just a marketing ploy but as a way to better relate previously existing and new concepts.

In that sense and regards to intent, it make sense to distinguish various subcategories of intent as follows:

- Operational Intent: defines intent related to operational goals of an operator; corresponds to the original "intent" term and the concepts defined in this document.

- Rule Intent: a synonym for policy rules regarding what to do when certain events occur.

- Service intent: a synonym for customer service model [RFC8309].

- Flow Intent: A synonym for a Service Level Objective for a given flow.

A comprehensive set of classifications of different concepts and categories of intent will be described in a separate document.

9. IANA Considerations

Not applicable

10. Security Considerations

This document describes concepts and definitions of Intent-based Networking. As such, the below security considerations remain high level, i.e. in the form of principles, guidelines or requirements.

More detailed security considerations will be described in the documents that specify the architecture and functionality.

Security in Intent-based Networking can apply to different facets:

- Securing the intent-based system itself.
- Mitigating the effects of erroneous, harmful or compromised intents.
- Expressing security policies or security-related parameters with intents.

Securing the intent-based system aims at making the intent-based system operationally secure by implementing security mechanisms and applying security best practices. In the context of Intent-based Networking, such mechanisms and practices may consist in intent verification and validation; operations on intents by authenticated and authorized users only; protection against or detection of tampered intents. Such mechanisms may also include the introduction of multiple levels of intent. For example, intent related to securing the network should occur at a "deeper" level that overrides other levels of intent if necessary, and that is not subject to modification through regular operations but through ones that are specifically secured. Use of additional mechanisms such as explanation components that describe the security ramifications and trade-off should be considered as well.

Mitigating the effects of erroneous or compromised intents aims at making the intent-based system operationally safe by providing checkpoint and safeguard mechanisms and operating principles. In the context of Intent-based Networking, such mechanisms and principles may consist in the ability to automatically detect unintended, detrimental or abnormal behavior; the ability to automatically (and gracefully) rollback or fallback to a previous "safe" state; the ability to prevent or contain error amplification (due to the combination of higher degree of automation and the intrinsic higher degree of freedom, ambiguity, and implicit conveyed by intents); dynamic levels of supervision and reporting to make the user aware of the right information, at the right time with the right level of context. Erroneous or harmful intents may inadvertently propagate and compromise security. In addition, compromised intents, for example intent forged by an inside attacker, may sabotage or harm the network resources and make them vulnerable to further, larger attacks, e.g. by defeating certain security mechanisms.

Expressing security policies or security-related parameters with intents consists in using the intent formalism (a high-level,

declarative abstraction), or part(s) of an intent statement to define security-related aspects such as data governance, level(s) of confidentiality in data exchange, level(s) of availability of system resources, of protection in forwarding paths, of isolation in processing functions, level(s) of encryption, authorized entities for given operations, etc.

The development and introduction of Intent-Based Networking in operational environments will certainly create new security concerns. Such security concerns have to be anticipated at the design and specification time. However, Intent-Based Networking may also be used as an enabler for better security. For instance, security and privacy rules could be expressed in more human-friendly and generic way and be less technology-specific and less complex, leading to fewer low-level configuration mistakes. The detection of threats or attacks could also be made more simple and comprehensive thanks to conflict detection at higher-level or at coarser granularity

More thorough security analyses should be conducted as our understanding of Intent-Based Networking technology matures.

11. References

11.1. 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>.

[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>.

11.2. 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.

[I-D.ietf-teas-te-service-mapping-yang]
Lee, Y., Dhody, D., Fioccola, G., WU, Q., Ceccarelli, D., and J. Tantsura, "Traffic Engineering (TE) and Service Mapping Yang Model", draft-ietf-teas-te-service-mapping-yang-04 (work in progress), July 2020.

[Lenrow15]
Lenrow, D., "Intent As The Common Interface to Network Resources, Intent Based Network Summit 2015 ONF Boulder: IntentNBI", February 2015.

[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>.

[RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299, DOI 10.17487/RFC8299, January 2018, <https://www.rfc-editor.org/info/rfc8299>.

[RFC8309] Wu, Q., Liu, W., and A. Farrel, "Service Models Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018, <https://www.rfc-editor.org/info/rfc8309>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <https://www.rfc-editor.org/info/rfc8345>.

[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.

[TR523] Foundation, O. N., "Intent NBI - Definition and Principles. ONF TR-523.", October 2016.

Authors' Addresses

Alexander Clemm
Futurewei
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

Jeff Tantsura
Apstra, Inc.

Email: jefftant.ietf@gmail.com