Internet Engineering Task Force
Internet Draft
Intended status: Informational

C. Li
China Telecom
Y. Cheng
China Unicom
J. Strassner
O.Havel
W.Xu
Huawei Technologies

October 22, 2018

Expires: April 2019

Intent Classification
draft-draft-li-intent-classification-01

Status of this Memo

Copyright Notice

Abstract

RFC 7575 [RFC7575] defines Intent as an abstract high-level policy used to operate the network. Intent management system includes an interface for users to input requests and an engine to translate the intents into the network configuration and manage their lifecycle. Up to now, there is no commonly agreed definition, interface or model of intent.

This document discusses what intent means to different stakeholders, describes different ways to classify intent, and an associated taxonomy of this classification.

Table of Contents

1. Introduction

Different SDOs (such as [ANIMA][ONF]) have proposed intent as a declarative interface for defining a set of network operations to execute.

Although there is no common definition or model of intent which are agreed by all SDOs, there are several shared principles:

- intent should be declarative, using and depending on as few deployment details as possible and focusing on what and not how

- intent should provide an easy-to-use interface, and use terminology and concepts familiar to its target audience

- intent should be vendor-independent and portable across platforms

- the intent framework should be able to detect and resolve conflicts between multiple intents

SDOs have different perspectives on what intent is, what set of actors it is intended to serve, and how it should be used. This document provides several dimensions to classify intents.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Acronyms

CLI: Command Line Interface

SDO: Standards Development Organisation

SUPA: Simplified Use of Policy Abstractions

VPN: Virtual Private Network

4. Abstract intent requirements

In order to understand the different intent requirements that would drive intent classification, we first need to understand what intent means for different intent users.

4.1. What is Intent

The term Intent has become very widely used in the industry for different purposes, sometimes it is not even in agreement with SDO shared principles mentioned in the Introduction. Different stakeholders consider an intent to be an ECA policy, a GBP policy, a business policy, a network service, a customer service, a network configuration, application / application group policy, any operator/administrator task, network troubleshooting / diagnostics / test, a new app, a marketing term for existing management/orchestration capabilities, etc. Their intent is sometimes technical, non-technical, abstract or technology specific. For some stakeholders, intent is a subset of these and for other stakeholders intent is all of these. It has in some cases become a term to replace a very generic 'service' or 'policy' terminology.

While it is easier for those familiar with different standards to understand what service, CFS, RFS, resource, policy continuum, ECA policy, declarative policy, abstract policy or intent policy is, it may be more difficult for the wider audience. Intent is very often just a synonym for policy. Those familiar with policies understand the difference between a business, intent, declarative, imperative and ECA policy. But maybe the wider audience does not understand the difference and sometimes equates the policy to an ECA policy.

Therefore, it is important to start a discussion in the industry about what intent is for different solutions and intent users. It is also imperative to try to propose some intent categories / classifications that could be understood by a wider audience. This would help us define intent interfaces, DSLs and models.

4.2. Intent Solutions & Intent Users

Different Solutions and Actors have different requirements, expectations and priorities for intent driven networking. They require different intent types and have different use cases. Some users are more technical and require intents that expose more technical information. Other users do not understand networks and require intents that shield them from different networking concepts and technologies.

4.3. Current Problems & Requirements

Network APIs and CLIs are too complex due to the fact that they expose technologies & topologies. App developers and end-users do not want to set IP Addresses, VLANs, subnets, ports, etc. Operators and administrators would also benefit from the simpler interfaces, like:

- Allow Customer Site A to be connected to Internet via Network B

- Allow User A to access all internal resources, except the Server B

- Allow User B to access Internet via Corporate Network A

- Move all Users from Corporate Network A to the Corporate Network B

- Request Gold VPN service between my sites A, B and C

- Provide CE Redundancy for all Customer Sites

- Add Access Rules to my Service

Networks are complex, with many different protocols and encapsulations. Some basic questions are not easy to answer:

- Can User A talk to User B?

- Can Host A talk to Host B?

- Are there any loops in my network?

- Are Network A and Network B connected?

- Can User A listen to communications between Users B & C?

Operators and Administrators manually troubleshoot and fix their networks and services. They instead want:

- a reliable network that is self-configured and self-assured based on the intent

- to be notified about the problem before the user is aware

- automation of network/service recovery based on intent (self-healing, self-optimization)

- to get suggestions about correction/optimization steps based on experience (historical data & behaviour)

Therefore, Operators and Administrators want to:

- simplify and automate network operations

- simplify definitions of network services

- provide simple customer APIs for Value Added Services (operators)

- be informed if the network or service is not behaving as requested

- enable automatic optimization and correction for selected scenarios

- have systems that learn from historic information and behaviour

End-Users cannot build their own services and policies without becoming technical experts and they must perform manual maintenance actions. Application developers and end-users/subscribers want to be able to:

- build their own network services with their own policies via simple interfaces, without becoming networking experts

- have their network services up and running based on intent and automation only, without any manual actions or maintenance

4.4. Intent Types that need to be supported

The following intent types need to be supported, in order to address the requirements from different solutions and intent users:

- Customer network service intent

- Network resource management

- Cloud and cloud resource management

- Network Policy intent

- Task based intents

- System policies intents

5. The Policy Continuum

The Policy Continuum defines the set of actors that will create, read, use, and manage policy. Each set of actors has their own terminology and concepts that they are familiar with. This captures the fact that business people do not want to use CLI, and network operations center personnel do not want to use non-technical languages.

6. Functional Characteristics and Behavior

Intent can be used to operate immediately on a target (much like issuing a command), or whenever it is appropriate (e.g., in response to an event). In either case, intent has a number of behaviors that serve to further organize its purpose, as described by the following subsections.

6.1. Persistence

Intents can be classified into transient/persistent intents.

If intent is transient, it has no lifecycle management. As soon as the specified operation is successfully carried out, the intent is finished, and can no longer affect the target object.

If the intent is persistent, it has lifecycle management. Once the intent is successfully activated and deployed, the system will keep all relevant intents active until they are deactivated or removed.

6.2. Granularity

Intents can have different granularities: high granularity, low granularity and anything in between.

High granularity intents are more complex to design but are the most valuable. Intent translation, intent conflict resolution and intent verification are very complex and require advanced algorithms. Examples: e2e network service, like customer network service over physical & virtual network, over access, metro, dc and wan with all related QoS, security and application policies.

Low granularity intents, like some path checks (can A talk to B) or individual network service/network/application/user policies, are the least complex. Their intent translation, intent conflict resolution and intent verification are much simpler than for high granularity intents.

6.3. Abstracting Intent Operation

The modeling of Policies can be abstracting using the following three-tuple:

{Context, Capabilities, Constraints}

Context grounds the policy, and determines if it is relevant or not for the current situation. Capabilities describe the functionality that the policy can perform. Capabilities take different forms, depending on the expressivity of the policy as well as the programming paradigm(s) used. Constraints define any restrictions on the capabilities to be used for that particular context. Metadata can be optionally attached to each of the elements of the three-tuple, and may be used to describe how the policy should be used and how it operates, as well as prescribe any operational dependencies that must be taken into account.

Put another way:

- o Context selects policies based on applicability

- o Capabilities describe the functionality provided by the policy

- o Constraints restrict the capabilities offered and/or the behavior of the policy

Hence, the difference between imperative, declarative, and other types of policies lies in how the elements of this three-tuple are used according to that particular programming paradigm. This is how [SUPA] was designed: a Policy is a container that aggregates a set of statements.

6.4. Policy Subjects and Policy Targets

Policy subject is the actor that performs the action specified in the policy. It can be the intent management system which executes the policy. Policy target is a set of managed objects which may be affected in the policy enforcement.

6.5. Policy Scope

Policies used to manage the behavior of objects that they are applied to (e.g., the target of the policy).

It is useful to differentiate between the following categories of targets:

- Policies defined for the Customer or End-User

- Policies defined for the management system to act on objects in the domain that the management system controls

- Policies defined for the management system to act on objects in one or more domains that the management system does not directly control

The different origins and views of these three categories of actors lead to the following important differences:

- Network Knowledge. This area is explored using three exemplary actors that have different knowledge of the network.

Customers and end-users do not necessarily know the functional and operational details of the network that they are using. Furthermore, most of the actors in this category lack skills to understand such details; in fact, such knowledge is typically not relevant to their job. In addition, the network may not expose these details to its users. This class of actor focuses on the applications that they run, and uses services offered by the network. Hence, they want to specify policies that provide consistent behavior according to their business needs. They do not have to worry about how the policies are deployed onto the underlying network, and especially, whether the policies need to be translated to different forms to enable network elements to understand them.

Application developers work in a set of abstractions defined by their application and programming environment(s). For example, many application developers think in terms of objects (for example, a VPN). While this makes sense to the application developer, most network devices do not have a VPN object per se; rather, the VPN is formed through a set of configuration statements for that device in concert with configuration statements for the other devices that together make up the VPN. Hence, the view of application developers matches the services provided by the network, but may not directly correspond to other views of other actors.

Management personnel, such as network Administrators, have complete knowledge of the underlying network. However, they may not understand the details of the applications and services of Customers and End-Users.

- o Automation. In theory, intents from both end-user and management system can be automated. In practice, most intents from end-user are created manually according to business request. End-users do not create or alter intents unless there is change in business. Intents from management systems can be created or altered to reflect with network policy change. For example, end-users create intents to set up paths between hosts, while the management system creates an intent to set a global link utilization limit.

7. IANA Considerations

This document includes no request to IANA.

8. Security Considerations

This document does not have any Security Considerations.

9. IANA Considerations

This document includes no request to IANA.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[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

[SUPA] Strassner, J., "Simplified Use of Policy Abstractions", 2017, <https://datatracker.ietf.org/doc/draft-ietf-supa-generic-policy-info-model/?include_text=1>.

10.2. Informative References

[ANIMA] Du, Z., "ANIMA Intent Policy and Format", 2017, <https://datatracker.ietf.org/doc/draft-du-anima-an-intent/>.

[ONF] ONF, "Intent Definition Principles", 2017, <https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR-523_Intent_Definition_Principles.pdf>.

[ONOS] ONOS, "ONOS Intent Framework", 2017, <https://wiki.onosproject.org/display/ONOS/Intent+Framework/>.

11. Acknowledgments

The authors would like to thank Will (Shucheng) Liu and Xiaolin Song for their comments to this document.

Authors' Addresses

Chen Li
China Telecom
No.118 Xizhimennei street, Xicheng District
Beijing 100035
P.R. China
Email: lichen.bri@chinatelecom.cn

Ying Cheng
China Unicom
No.21 Financial Street, XiCheng District
Beijing 100033
P.R. China
Email: chengying10@chinaunicom.cn

John Strassner
Huawei Technologies
2330 Central Expressway
Santa Clara 95138
Email: john.sc.strassner@huawei.com

Olga Havel
Huawei Technologies
Email: olga.havel@huawei.com

Weiping Xu
Huawei Technologies
Bantian, Longgang District
shenzhen 518129
P.R. China

Email: xuweiping@huawei.com