

Network Working Group
Internet Draft
Intended status: Standard Track
Expires: November 2015

J. Bi
Tsinghua University
R. Tadepalli
Wipro Limited
M. Hayashi
KDDI R&D Labs. Inc.
May 7, 2015

DDC Service Policy YANG Data Model
draft-bi-supra-policy-model-02

Abstract

This document describes a YANG data model for traffic steering policies in Distributed Data Center (DDC) scenarios. The policy model is a specific data model for traffic steering using VPN technology. It helps the service management in Simplified Use of Policy Abstractions (SUPA) to model the policy (a set of constraints and rules) that defines how a VPN service is monitored by bandwidth and managed during its lifecycle. Two traffic steering applications have been provided such as traffic optimization with pass or bypass specific nodes or sites.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

This Internet-Draft will expire on October 25, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions used in this document	4
3. Network Configuration Model Overview	4
4. Network Policy Configuration Modules	4
4.1. Network Policy Model	5
4.2. Policy Applications in DDC services	7
4.2.1. Model for Traffic Steering Policy	9
4.2.2. Policy Based Traffic Steering Operation	17
5. Security Considerations	17
6. IANA Considerations	17
7. Contributor List	17
8. Acknowledgments	18
9. References	18
9.1. Normative References	18
9.2. Informative References	18

1. Introduction

In order to support the DDC service with VPN connection as well as new services, it brings new requirements for both network providers and service providers. Rapid uptake of new services requires dynamic service provisioning capabilities in the service management. This is achieved using policies that can be created by the operators once and the service management refers to these policies to infer how a given service needs to be provisioned considering the current state of the network.

In SUPA framework, network policy is a predefined rule or a set of rules that the service management use to map the service to the lower level network infrastructures. More specifically, based on different level of policy abstractions, there are a generic policy model on top and ECA policy, intent policy to handle service management. Note that, the generic policy is use case independent while the ECA policy has to be defined with objects from service model. In this way, an ECA policy defines:

An event or a set of events that trigger the evaluation of policy: This is the trigger for the service management application to evaluate if a policy needs to be applied. For example a user action to provision a new VPN service can be an event.

A set of conditions that need to be satisfied for the policy to be applicable: This enables service management to select the right policy by validating the conditions against the current network state.

A set of actions that should be triggered as part of the policy execution: This enables the service management to provision the service.

Meanwhile, DDC service which is mainly relied on VPN [RFC4110] needs policy based management and controlling capability from the service management systems to facilitate the service deployment both inter data centers and within data center.

This document introduces YANG [RFC6020] [RFC6021] data models for SUPA configuration. Such models can facilitate the standardization for the interface of SUPA, as they are compatible to a variety of protocols such as NETCONF [RFC6241] and [RESTCONF]. Please note that in the context of SUPA, the term "application" refers to an operational and management applications employed, and possibly implemented, by an operator. The policy model is based on the first example - DDC services.

With respect to the scope, defining an information model for the policy exchange between the policy manager and policy agent and a corresponding data model based on yang to support specific DDC service use case is initial goal of this document. The protocol specific aspects are deferred to respective implementations. Also certain foundational concepts of the model are intentionally left open to enable future extension. Here the traffic steering policy in DDC use case provides a concrete example for a specific network technology and service, as what constitutes a policy could itself vary depending on the context where it is used, e.g. there could

be tenant specific policies, site specific, network domain specific etc.

2. Conventions used in this document

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 [RFC2119]. In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying [RFC2119] significance.

3. Network Configuration Model Overview

Figure 1 illustrates the network configuration model which contains several modules for specific services such as VPN management. Basically, service model is to define the creation and configuration of the VPN service, while the policy model is more focused on the adjustment or optimization of the flow path during the lifecycle of the VPN service based on the predefined policy.

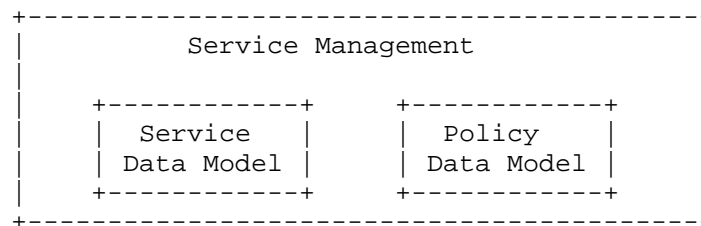


Figure 1: Overview of configuration model structure

4. Network Policy Configuration Modules

In this section, a policy model is defined with an application for traffic steering between data centers are provided to illustrate how to use the policy model. The policy model and policy configuration are based on a set of specific network services and the framework of SUPA [SUPA-framework]. On the other hand, the policy model should be working on the orchestration level which is above network element and below OSS level based on the YANG model classification in [draft-bogdanovic-netmod-yang-model classification-02]

4.1. Network Policy Model

A Policy Rule can be expressed in different ways and its content is separated from its representation. Therefore, this object takes different forms, depending on the level of abstraction desired. For example, an event-condition-action policy is expressed as a tuple of three statements. Within SUPA scope, the service management system takes the E-C-A fashioned policy model to handle the VPN management in the data center use case to improve bandwidth utilization and facilitate service deployment.

Event: a significant occurrence in time that triggers the evaluation of the condition of the policy rule

Condition: a set of tests that determine if the actions of the policy rule should be executed or not

Action: a set of operations that should be performed if the condition is true

Note that event, condition, and action can each be made up of Boolean clauses

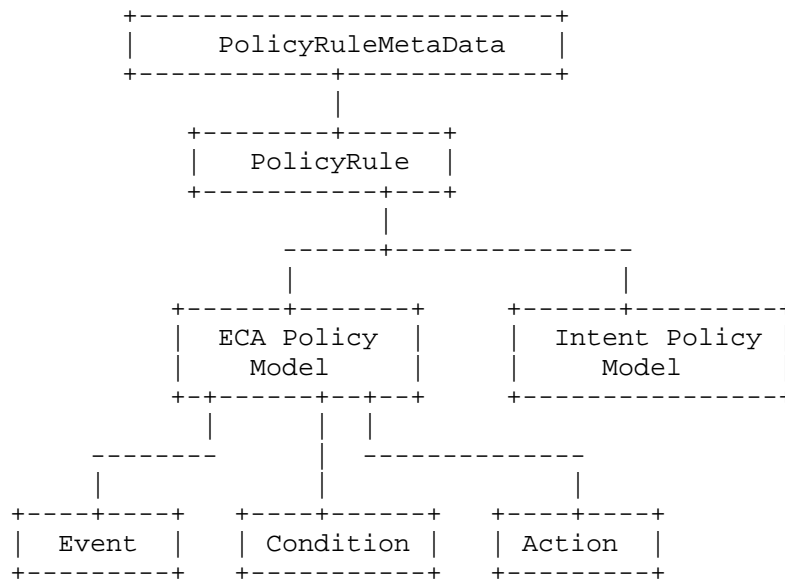


Figure 2: Overview of ECA policy model

```

module: ietf-supa-policy
  +--rw supa-policy
    +--rw supa-policy-name?          string
    +--rw supa-policy-type?          enumeration
    +--rw applicable-service-type?    enumeration
    +--rw supa-policy-priority?       uint8
    +--rw supa-policy-validity-period
      | +--rw start?                yang:date-and-time
      | +--rw end?                  yang:date-and-time
      | +--rw duration?             uint32
      | +--rw periodicity?          enumeration
    +--rw supa-policy-metadata?       uint32
    +--rw supa-policy-atomic
      | +--rw supa-ECA-policy-rule
      |   +--rw policy-rule-name?    string
      |   +--rw policy-rule-type?    enumeration
      |   +--rw policy-rule-deploy-status? enumeration
      |   +--rw policy-rule-exec-status? enumeration
      |   +--rw has-policy-events?    boolean
      |   +--rw has-policy-conditions? boolean
      |   +--rw has-policy-actions?   boolean
    +--rw supa-policy-statement
      +--rw event-clause
      | +--rw policy-variable?      string
      +--rw condition-clause
      | +--rw policy-variable?      string
      | +--rw policy-operator?      string
      | +--rw policy-value?         string
      +--rw action-clause
      | +--rw policy-action?        string

```

As shown in the YANG tree of the ECA policy model, it is basically following the generic policy model from [draft-strassner-supa-generic-policy-model-00] to define the policy model structure of the ECA policy.

The top level class "supa-policy": it has some basic attributes such as name and type. "applicable-service-type" is related the service which define the service being supported by the policy model. "supa-policy-priority" and "supa-policy-validity-period" define when and how often the policy will be executed, which is designed to handle some time based management policy. E.g. "perform LOAD BALANCING at 2am every Thursday" such policy will

introduce the timing issue. And finally, "supa-policy-atomic" and "supa-policy-statement" are two major components of the policy model.

"supa-policy-atomic" is a type of Policy Container which here contains the ECA model policy model. In addition to the "name" and "type" attributes, "policy-rule-deploy-status" and "policy-rule-exec-status" describe the deploy status and execute status of the ECA policy respectively. The most important components are "has-policy-events", "has-policy-conditions" and "has-policy-actions" which denote whether the policy rule has an "event", "condition" and "action" clause. Note that each definition of the clause is in the "supa-policy-statement"

"supa-policy-statement" separates the representation of a SUPA Policy from its implementation. Its subclasses enable the developer to define a SUPA Policy as either a completely reusable set of SUPA Policy objects or as an efficient encoding made up of attributes. "event-clause" defines the policy event the system monitors. If the predefined event is evaluated as TRUE, then the condition will be evaluated. "condition-clause" defines the condition to trigger the corresponding action. It should be a three-tuple clause with {PolicyVariable, PolicyOperator, PolicyValue}. E.g., in "bandwidth utilization > 80%", "bandwidth utilization" is the variable, ">" is the operator and "80%" is the value. "action-clause" defines the consequence action as the "condition-clause" is evaluated as TRUE.

4.2. Policy Applications in DDC services

More specifically, for the networking system an E-C-A based policy model can cover use cases as follows:

Network Policy

```
-event:
  -- bandwidth usage
  -- port N status
  -- TTL value
-condition :
  -- bandwidth usage > x
  -- port N is up
  -- TTL < y
-action:
  -- adjust flow
  -- use backup link
  -- retry
```

For the first one, if the bandwidth usage of a contain link is above threshold, the flow will be adjusted. The event is the link bandwidth usage, the condition is bandwidth usage above the x and the action is to adjust the flow.

The second one is, if the port N is up, the system will use backup link.

The third one is, if the TTL is below y, retry.

The following describes SUPA policy model designed for DDC services use case [SUPA-DDC] to do the traffic steering among DCs. [SUPA-DDC] took a large-scale Internet Data Center (IDC) operator as an example to describe what SUPA needs to do in VPN-based traffic steering. Here the ECA policy model only focus on the policy based traffic steering application.

The traffic steering policy is used in dynamical link configuration in DDC services [SUPA-DDC]. The service management can dynamically adjust the traffic flow in the links based on traffic steering policy. The policy model specifies some high level requirements to the links, like routing strategy. For example in this specific traffic steering policy model, if the link bandwidth usage is above certain threshold, the monitored flow will be forced to routed to the third place to bypass the overloaded node or site.

Module "ietf-sup-policy" defines E-C-A based policy model to describe the policy management in DDC service. In effect, the module can be expanded with more operation services for DDC services. The ECA model here with the DDC traffic steering application is an instantiation and inheritance of the generic policy model with ECA policy rule.


```

module: ietf-supa-policy
  +--rw supa-ddc-policy
    +--rw adjust-flow-path-policy
      +--rw adjust-flow-path* [vpn-name]
        +--rw vpn-name          string
        +--rw vpn-type?         enumeration
        +--rw flow-name?        string
        +--rw traffic-steering-policy-rule
          +--rw policy-rule-deploy-status? enumeration
          +--rw policy-rule-exec-status?  enumeration
          +--rw policy-event
            | +--rw bandwidth?  string
          +--rw policy-condition
            | +--rw bandwidth-utilization? enumeration
            | +--rw operator?    enumeration
            | +--rw value?      uint32
          +--rw policy-action-adjust-path
            +--rw constraint-nodes
              | +--rw constraint-node* [nodeId]
              |   +--rw nodeId          string
              |   +--ro constraint-type? enumeration
              |   +--rw sequence?      uint32
            +--rw constraint-sites
              +--rw constraint-site* [siteId]
                +--rw siteId          string
                +--ro constraint-type? enumeration
                +--rw sequence?      uint32

```

4.2.1. Model for Traffic Steering Policy

```

<CODE BEGINS>
module ietf-supa-policy {
  namespace "urn:ietf:params:xml:ns:yang:ietf-supa-policy";
  // replace with IANA namespace when assigned
  prefix policy;

  import ietf-inet-types {
    prefix inet;
  }

  organization "IETF";
  contact
    "Editor: Jun Bi
     junbi@tsinghua.edu.cn";

  description
    "This YANG module defines a component that describing

```

the ddc policy model for monitoring and optimizing tenant's DC (data center) services that are deployed between multiple data centers.

Terms and Acronyms

DDC: Distributed Data Center

L2VPN: Layer 2 Virtual Private Network

L3VPN: Layer 3 Virtual Private Network";

```
revision 2015-05-06 {
  description
    "Revised revision.";
  reference
    " Network Policy YANG Data Model ";
}
```

```
container supa-ddc-policy{
  description
    "Distributed Data Center Service Operation Data";
```

```
  container adjust-flow-path-policy {
    description
      "To improve the bandwidth utilization (or reduce the cost,
      or other reason) and mitigate traffic congestion,management
      system/ application requires controller to adjust certain
      flows to pass/bypass certain nodes(or links), when, e.g.,
      bandwidth utilization exceed certain threshold. Vpn name,
      vpn type, adjusted flow and specified nodes (that the flow
      should pass) should be told to controller. So that the
      controller can configure the network elements to change the
      VRF table and routing table. If the link bandwidth usage is
      above certain threshold, the monitored flow will be forced
      to routed to the third place to bypass the overloaded node
      or site.";
```

```
  list adjust-flow-path {
    key "vpn-name";
    description
      "The list of VPN and flow that need to be adjusted in
      specific paths. So as to optimize traffic in the links
      that are between data centers.";
    leaf vpn-name {
      type string;
      mandatory true;
      description
        "Indicates the name of VPN that the adjusted flow
        belongs to. A VPN instance is identified by vpn-name.
        It should be one of the created VPN instance names";
```

```
    }
    leaf vpn-type {
      type enumeration {
        enum L2VPN {
          description "L2VPN";
        }
        enum L3VPN {
          description "L3VPN";
        }
      }
      description "Indicates the type of VPN instance that the
        adjusted flow belongs to. L2VPN or L3VPN";
    }
    leaf flow-name {
      type string;
      description "The name of the adjusted flow. So as to tell
        the Controller which flow should be adjusted";
    }
  }
  container traffic-steering-policy-rule{

    leaf policy-rule-deploy-status {
      description "It defines the current deployment status
        of this policy rule. Both operational and test mode values
        are included in its definition.";
      type enumeration {
        enum 0{
          description "undefined";
        }
        enum 1{
          description "deployed and enabled";
        }
        enum 2{
          description "deployed and in test";
        }
        enum 3{
          description "deployed but not enabled";
        }
        enum 4{
          description "ready to be deployed";
        }
        enum 5{
          description "not deployed";
        }
      }
    }
  }
```

```
leaf policy-rule-exec-status {
  description "It defines the current execution status
of this policy rule. Both operational and test mode values
are included in its definition";
  type enumeration {
    enum 0{
      description "undefined";
    }
    enum 1{
      description "executed and SUCCEEDED (operational
mode)";
    }
    enum 2{
      description "executed and FAILED (operational
mode)";
    }
    enum 3{
      description "currently executing (operational
mode)";
    }
    enum 4{
      description "executed and SUCCEEDED (test mode)";
    }
    enum 5{
      description "executed and FAILED (test mode)";
    }
    enum 6{
      description "currently executing (test mode)";
    }
  }
}

container policy-event{
  description "The Policy defines an ECA type policy
model for the service management of traffic optimization
between DCs. If the predefined event is monitored, then
the controller will evaluate if the condition is matched.
If the condition is matched, the corresponding action will
be performed. Here in this case, bandwidth is the 'event'
to monitor in the traffic steering policy case";

  leaf bandwidth {
    type string;
  }
}

container policy-condition {
```

```

description "There are several types of 'condition'
match, here it is above/below";
leaf bandwidth-utilization {
  type enumeration {
    enum utilization {
      description "In this case, the bandwidth
utilization is the 'event'. The controller will
monitor the bandwidth utilization if it is above
the threshold which is the link utilization ratio,
0-100%";
    }
    enum BW {
      description " In this case, the bandwidth usage
is the 'event'. The controller will monitor the
bandwidth usage if it is above the threshold which
is bandwidth value, 2G,10G e.g.";
    }
  }
}

leaf operator{
  type enumeration {
    enum above {
      description "If the value of the monitored event
is above the threshold, do the action. E.g., in
this case, if the bandwidth utilization is above
70%, 'above' is the 'match-type' here.";
    }
    enum below {
      description "If the value of the monitored event
is below the threshold, do the action. E.g., in this
case, if the bandwidth usage is below 2Gb, 'below'
is the 'match-type' here.";
    }
  }
}

leaf value {
  description "The target value of the monitored
'event' or the threshold to trigger the action. E.g.,
in this case, if the bandwidth utilization is above 70%,
'70%' is the 'value' here. If the bandwidth usage is
above 2Gb, here '2Gb' is the 'value' here";
  type uint32;
}

container policy-action-adjust-path {

```

description "This is the corresponding 'action' when the condition has been triggered. E.g., if the link occupied ratio is above 70%, adjust the path of the flow. Here the 'adjust-path' is the action of the ECA policy. If the 'event' is monitored, and the 'condition' is triggered, the corresponding 'action', rerouted the flow path, will be performed.";

```
container constraint-nodes {
  description "The generated 'action' of the ECA
  policy. It is a description of the constraints on the
  nodes those need to be adjusted. Here the 'action' only
  describe the constraint not the real operation. And
  then, the controller will compute a new route based on
  the constraint. More specifically, the constraint-nodes
  are a set of nodes that specify the constraints by the
  up-level. Each node in the list will be passed/bypassed
  when computing the new route.";
```

```
list constraint-node {
```

```
  max-elements unbounded;
```

```
  min-elements 0;
```

```
  description "There can be a set of nodes to be
  constrained. Each node in the list will be passed/
  bypassed when computing the new route.";
```

```
  key nodeId;
```

```
  description " List of nodes that the adjusted
  flow needs to pass or bypass so as to adjust the flow
  path between data centers. E.g. if the event and
  condition has been triggered that the vpn flow will be
  adjusted. The constraint-nodes specify how to adjust
  the flow and which nodes will be adjusted. Each node
  in the list will be told to controller to pass/bypass
  when computing the new route.";
```

```
  leaf nodeId {
```

```
    description "The node needs to be constrained
    that describe which nodes will be adjusted in the
    path. The node ID is the identifier of the node that
    the controller will take into account when computing
    the new route.";
```

```
    config true;
```

```
    type string {
```

```
      length "0..64";
```

```
      pattern "([^\?]*)";
```

```
    }
```

```
  }
```

```
  leaf constraint-type {
```

```
    description "The node can be bypassed or
    passed.";
```

```
    config false;
```

```

    type enumeration {
        enum bypass {
            value 0;
            description "The node will be bypassed.
            E.g., if the corresponding action of the ECA
            policy is to bypass the busy node 1, 2 and 3,
            then the node-list will be nodeId: 1, 2 and 3,
            while the 'constraint-type' of each node is
            'bypass' which denote all these nodes should be
            bypassed when computing the new route. Then the
            controller will compute the new route based on
            the constraint to compute a new path without
            node 1, 2
            and 3.";
        }
        enum pass {
            value 1;
            description "The node will be passed. E.g., if
            the corresponding action of the ECA policy is to
            pass the light loaded nodes 1, 2 and 3, then the
            node-list will be nodeId: 1, 2 and 3, while the
            'constraint-type' of each node is 'pass' which
            denote all these nodes should be passed when
            computing the new route. Then the controller
            will compute the new route based on the
            constraint to compute a new path without node
            1, 2 and 3.";
        }
    }
}
leaf sequence {
    description "Constraint node sequence. The
    corresponding action can also specify the sequence
    of the node list that should be passed/bypassed.
    E.g., if the 'action' is to adjust the flow to a
    new path with node 1, 2 and 3, while the pass the
    node 1 first, then 3, finally 2. Here the sequence
    of the node-list will be 1-3-2. Then the controller
    will compute the new path based on the sequence.";
    config true;
    default 1;
    type uint32 {
        range "0..128";
    }
}
}
}
container constraint-sites {
    list constraint-site {

        max-elements unbounded;
        min-elements 0;
        description ".";
        key siteId;
    }
}

```

```

description " List of sites that the adjusted
flow needs to pass or bypass So as to adjust the flow
path between data centers. The site can be a group of
nodes at different granularities for example data
centers.";

description " List of sites that the adjusted
flow needs to pass or bypass So as to adjust the flow
path between data centers.";

leaf siteId {
  description "The site needs to be adjusted in
the flow path. The site";
  config true;
  type string {
    length "0..64";
    pattern "([^\?]*)";
  }
}
leaf constraint-type {

  description "The constraint type of the site
to specify whether the node site will be passed or
bypassed";
  config false;

  type enumeration {
    enum bypass {
      value 0;
      description "The site will be bypassed";
    }
    enum pass {
      value 1;
      description "The site will be passed";
    }
  }
}

leaf sequence {
  description "constraint site sequence";
  config true;
  default 1;
  type uint32 {
    range "0..128";
  }
}
}
}
}
}

```



```
    }  
  }  
}  
}  
<CODE ENDS>
```

4.2.2. Policy Based Traffic Steering Operation

The associated action generated by the ECA model is sent to the controller such as the list of constraint nodes which denotes the constraint of the adjusted flow path. E.g. the node C will be bypassed.

Based on the steering information, the Network Manager/Controller generates a path which meets the requirements: e.g., the computed path is (A, D, E, B). Network Manager/Controller also has to configure each device on the new path, not only the devices specified by the configuration such as node D, but also the devices in the underlying network which must be reconfigured, such as node E. The topology information is also necessary when Network Manager/Controller decides which device ought to be configured.

5. Security Considerations

TBD

6. IANA Considerations

This document has no actions for IANA.

7. Contributor List

Andy Bierman

YumaWorks, Inc.

Email: andy@yumaworks.com

8. Acknowledgments

This document has benefited from reviews, suggestions, comments and proposed text provided by the following members, listed in alphabetical order: Jing Huang, Junru Lin, Felix Lu, Juergen Schoenwaelder, Yiyong Zha, and Min Zha.

9. References

9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.
- [RFC6021] Schoenwaelder, J., "Common YANG Data Types", RFC 6021, October 2010.
- [RFC3272] Awduche, D., Chiu, A., Elwalid, A., Widjaja, I., and X. Xiao, "Overview and Principles of Internet Traffic Engineering", RFC 3272, May 2002.

9.2. Informative References

- [SUPA-framework] C. Zhou, L. M. Contreras, Q. Sun, and P. Yegani, " The Framework of Shared Unified Policy Automation (SUPA) ", IETF Internet draft, draft-zhou-supa-framework, January 2015.
- [SUPA-problem-statement] G. Karagiannis, Q. Sun, Luis M. Contreras, P. Yegani, and JF Tremblay, "Problem Statement for Shared Unified Policy Automation (SUPA)", IETF Internet draft, draft-karagiannis-supa-problem-statement, January 2015.
- [SUPA-DDC] Y. Cheng, and JF. Tremblay, "Use Cases for Distributed Data Center Applications in SUPA", IETF Internet draft, draft-cheng-supa-ddc-use-cases, January 2015.
- [RESTCONF] Bierman, A., Bjorklund, M., Watsen, K., and R. Fernando, "RESTCONF Protocol", draft-ietf-netconf-restconf (work in progress), July 2014.

[POLICY MODEL] Z. Wang, L. Dunbar, Q. Wu, "Network Policy YANG Data Model" draft-wang-netmod-yang-policy-dm, January 2015.

Authors' Addresses

Jun Bi
Tsinghua University
Network Research Center, Tsinghua University
Beijing 100084, China
Email: junbi@tsinghua.edu.cn

Raghavendra Tadepalli
Wipro Limited
Email: raghav.rao@wipro.com

Michiaki Hayashi
KDDI R&D Labs. Inc.
2-1-15 Ohara, Fujimino, Saitama, Japan. 356-8502
Email: mc-hayashi@kddilabs.jp

