OPSAWG Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2017

N. Sambo
M. Dallaglio
P. Castoldi
Scuola Superiore Sant'Anna
G. Fioccola
A. Di Giglio
Telecom Italia
F. Cugini
CNIT
G. Bernini
P. Giardina
Nextworks

June 27, 2017

Extending YANG for events, actions, and finite state machine
draft-sambo-opsawg-ccamp-supa-ext-yang-fsm-00

Abstract

Network operators and service providers are facing the challenge of deployment of systems from different vendors while looking for a trade-off among transmission performance, network device reuse, and capital expenditure without the need of being tied to single vendor equipment. The deployment and operation of more dynamic and programmable transport optical network infrastructures can be driven by adopting model-driven and software-defined control and management paradigms. In this context, YANG enables to compile a set of consistent vendor-neutral data models for optical networks and components based on actual operational needs emerging from heterogeneous use cases. This document extends YANG from data to functional modeling in order to describe events, operations, and finite state machine of YANG-defined network elements. The proposed models can be applied in the context of optical networks to pre-instruct data plane devices (e.g., an optical transponder) on the actions to be performed (e.g., code adaptation) in case some events, such as physical layer degradations, occur.

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). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on December 29, 2017.

Copyright Notice

Copyright (c) 2017 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

Networks are evolving toward more programmability, flexibility, and multi-vendor interoperability. Multi-vendor interoperability can be applied in the context of nodes, i.e. a node composed of components provided by different vendors (named white box) is assembled under the same control system. This way, operators can optimize costs and network performance without the need of being tied to single vendor equipment. NETCONF protocol RFC6241 [RFC6241] based on YANG data modeling language RFC6020 [RFC6020] is emerging as a candidate Software Defined Networking (SDN) enabled protocol. First, NETCONF supports both control and management functionalities, thus permits high programmability. Then, YANG enables data modeling in a vendor-neutral way. Some recent works have provided YANG models to describe attributes of links (e.g., identification), nodes (e.g., connectivity matrix), media channels, and transponders (e.g., supported forward error correction - FEC) of networks ([I-D.ietf-i2rs-yang-network-topo] [I-D.vergara-ccamp-flexigrid-yang] [I-D.zhang-ccamp-l1-topo-yang]), also including optical technologies. Such draft mainly refers to elastic optical networks (EONs), i.e. optical networks based on flexible grid where circuits with different bandwidth requirements are switched. EONs are expected to employ flexible transponders, i.e. transponders supporting multiple bit rates, multiple modulation formats, and multiple codes. Such transponders permits the (re-) configuration of the bit rate value based on traffic requirements, as well as the configuration of the modulation format and code based on the physical characteristics of a path (e.g., quadrature phase shift keying is more robust than 16 quadrature amplitude modulation). This document extends YANG from data to functional modeling in order to describe events, operations, and finite state machine of YANG-defined network elements. Such models can be applied to a case of transponder reconfiguration in EONs. In particular, the model enables a centralized remote network controller (managed by a network operator) to instruct a transponder controller about the actions to perform when certain events (e.g., failures) occur. The actions to be taken and the events can be re-programmed on the device.

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 [RFC2119].

3. Terminology

ABNO: Application-Based Network Operations

BER: Bit Error Rate

EON: Elastic Optical Network

FEC: Forward Error Correction

FSM: Finite State Machine

NETCONF: Network Configuration Protocol

OAM: Operation Administration and Maintenance

SDN: Software Defined Network

YANG: Yet Another Network Generator

4. Example of application

Flexible transponders enable several settings of transmission parameters' configuration, through the support of multiple modulation formats and forward error correction (FEC) schemes. This way, transmission parameters can be (re-)configured based on the physical layer conditions. The YANG model presented in this draft enables to pre-program reconfiguration settings of data plane devices in case of failures or physical layer degradations. In particular, soft failures are assumed. Soft failures imply transmission performance degradation, in turns a bit error rate (BER) increase, e.g. due to the ageing of some network devices. Without loosing generality, the ABNO architecture is assumed for the control and management of EONs (RFC7491 [RFC7491]). Considering the state of the art, when pre-FEC BER passes above a predefined threshold, it is expected that an alarm is sent to the OAM Handler, which communicates with the ABNO controller that may trigger an SDN controller (that could be the Provisioning Manager of ABNO RFC7491 [RFC7491]) for computing new transmission parameters. The involved ABNO modules are shown in the simplified ABNO architecture of Fig. 1. Then, transponders are reconfigured. When alarms related to several connections impacted by the soft failure are generated, this procedure may be particularly time consuming. The related workflow for transponder reconfiguration is shown in Fig. 2. The proposed model enables an SDN controller to instruct the transponder about reconfiguration of new transmission parameters values if a soft failure occurs. This can be done before the failure occurs (e.g., during the connection instantiation phase or during the connection service), so that data plane devices can

promptly reconfigure themselves without querying the SDN controller to trigger an on-demand recovery. This is expected to speed up the recovery process from soft failures. The related flow chart is shown in Fig. 3.

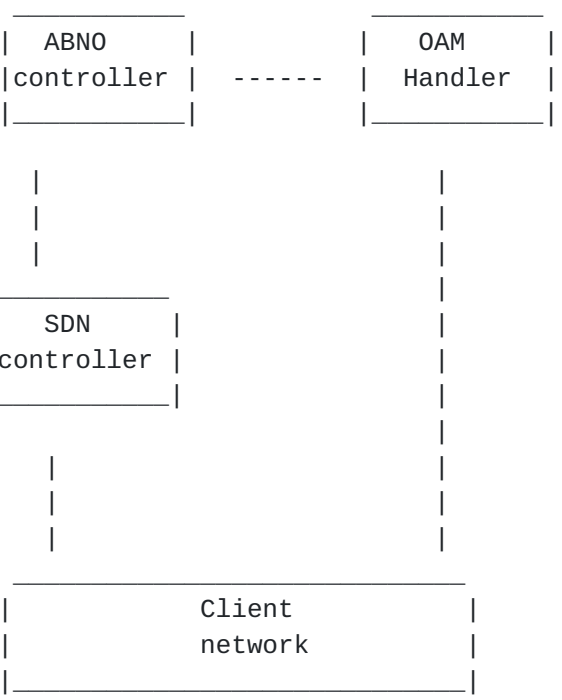

Figure 1: Assumed ABNO functional modules

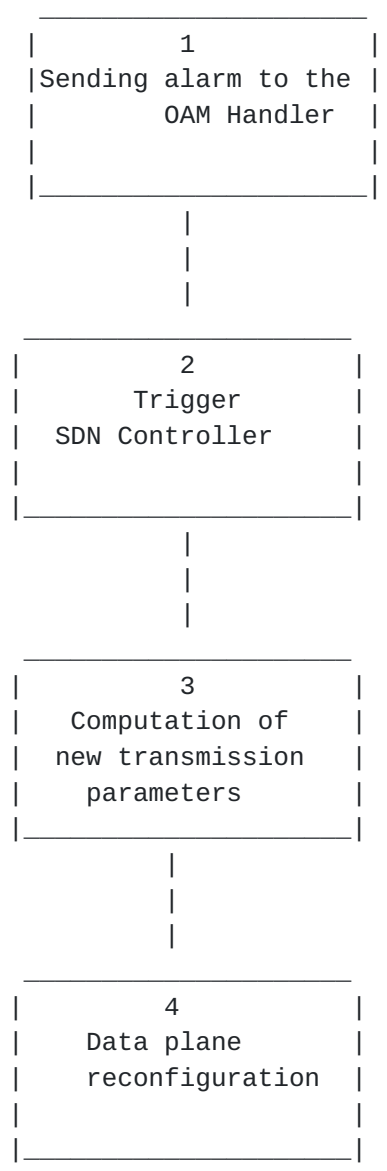

Figure 2: Flow chart of the expected state-of-the-art approach

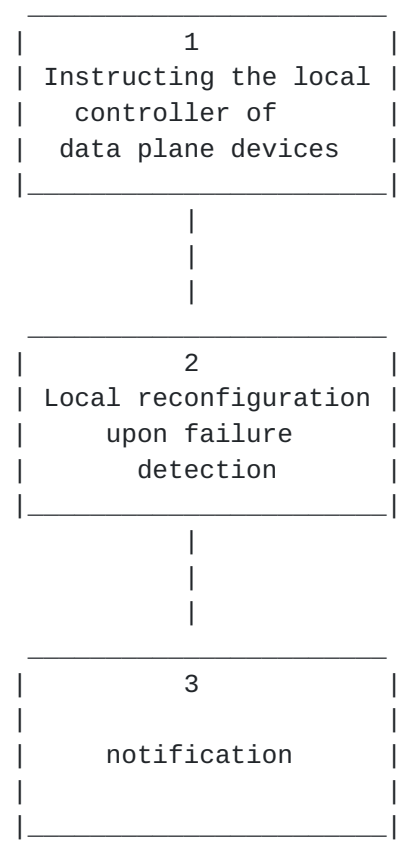

Figure 3: Flow chart of the approach exploiting YANG models in this draft

5. Extending YANG for events and reactions

The model extends YANG to define a list of events associated with specific reactions. The related code and tree are shown in the Appendix.

<event>: this element defines an event and it is composed by a set of leaves' attributes as follows.
<name>: this attribute defines the name of the event.
<type>: this attribute defines the type of the event from a pool of possible event types predefined inside the YANG model. Together with the <name> attribute, it uniquely identifies the event.
<description>: this optional attribute is a "string" describing the event
<filters>: this leaf is a list that enhances the description of an event. Given that an event does not necessary means a particular degradation or faults, this list can be used to define thresholds to express a measure

```
            of the event.
          <filter>: this leaf of <filters> defines a threshold to
                    characterize the event.
          <filter-id>: this leaf of <filters> define the
                    identifier number associated with the <filter>
                    attribute.
      <reaction>: this attribute defines a list of operations to take
            if the event occurs.
            <operations>: this list defines the set of operations
                    that have to be taken if the event occurs.
                      <id>: this leaf of <operations> defines the
                            identifier number of an operation.
                      <type>: this leaf of <operations> defines
                            the type of an operation.
                      <simple>: this leaf defines (differently
                            from <conditional> detailed below) an
                            operation that has to be directly
                            executed.
                             <execute>: this attribute recalls an
                                    RPC encapsulating the effective
                                    task (operation) to be executed
                                    by the data plane hardware.
                             <next-operation>: this attribute
                                    defines the identification
                                    number of a next operation that
                                    has to be taken.
                      <conditional>: this leaf enables a check
                            ("true" or "false") to be verified
                            before executing the operation. Based
                            on the check, the proper attributes
                            <execute> and <next-operation> are
                            considered.
                             <statement>: this leaf of
             <conditional> defines the
                                   condition to be verified before
                                   executing the operation.
                             <true>: this leaf of <conditional>
                                    defines a result of the check
                                    associated to <statement>.
                                    Proper <execute> and
                                    <next-operation> attributes are
                                    associated with this result of
                                    the check.
                             <false>: this leaf of <conditional>
                                    defines a result of the check
                                    associated to <statement>.
                                    Proper <execute> and
                                    <next-operation> attributes are
```

associated with this result of
the check.

6. Extending YANG for finite state machine (FSM)

This model extends the one of the events and reactions by adding the state information and state transition. More precisely, the model defines a list of states associated with events. Each state has a description attribute and it is identified through an id. Each state includes a list of events as defined in the event model, with the additional next-state attribute, which points to the next state. The related code and tree are shown in the Appendix.

<current-state>: it defines the current state of the FSM.
<states>: this element defines the FSM as follows.
<state>: this list defines all the FSM states.
<id>: this leaf attribute of <state> defines the identifier of the state
<name>: this leaf attribute of <state> defines the name of the state
<description>: this leaf is a "string" describing the state
<events>: this attribute is the one described in the previous section. In particular, this attribute defines a list of events that may induce a transition to another state in the FSM.
<next-state>: this attribute is included in the model <events> and defines the next state of FSM when an operation is executed.

7. Implementation for the considered use case of application

The models defined in this document are an extension of YANG through functions, events, and FSM, besides data modeling. These models can be used to enable a centralized network controller, managed by a network operator, to instruct data plane hardware on its reconfiguration if some events, such as a failure or physical layer degradation, occur. As an example, an optical signal impacted by a soft failure (i.e., a physical layer degradation inducing a pre forward error correction bit error rate increase - pre-FEC) can be maintained by adapting the FEC of the signal itself. This action to be taken and, more in general operations to be executed depending on critical events, can be (re-) programmed on the transponder by (re-) sending a NETCONF <edit-config> message to the device controller including a FSM defined by the YANG model. Such a system has the main goal to speed up the reaction of the network to certain events/

faults and to alleviate the workload of the centralized controller. The speed up derives from the fact that the centralized controller is able to pre-compute and pre-configure on the network devices the actions to take when an event occurs taking into account a global view and knowledge of the network. In this way, the device is already aware of the actions to be locally applied to reconfigure a connection, avoiding to inform the controller and to wait for the response indicating what to do. Consequently, part of the workload is also removed from the centralized controller. When the reaction is successfully completed in the data plane, the centralized controller can be notified about the faults and the taken action. A flexible transponder supporting two FEC types, 7% and 20%, is considered. A two-states FSM is also assumed. The states have <name> attribute set to "Steady" and "Fec-Baud-Adapt", respectively. In the "Steady" state, the signal is in a healthy condition, adopting a 7% FEC, with a pre-FEC BER below an assigned threshold of 9 x 10-4. A transition from this state can be triggered by the event with <name>=BER_CHANGE and <filter-type>=9 x 10-4, thus expressing a change of the pre-FEC BER above the threshold. In case the pre-FEC BER exceeds 9 x 10-4 due to a soft failure, the state machine evolves to the "Fec-Baud-Adapt" state and an adaptation to a more robust FEC of 20% (executed by the attribute <execute>) is performed. The system can return to the "Steady" state if the pre-FEC BER goes below another pre-defined threshold and the FEC is reconfigured to 7%.

8. Appendix

This appendix reports the YANG models code and the related tree.

8.1. YANG model for events and actions - Tree

```
   +--rw events
         +--rw event [name type]
            +--rw name           string
            +--rw type           event-type
            +--rw description?   string
            +--rw filters
            |  +--rw filter [filter-id]
            |     +--rw filter-id    yp:filter-id
            +--rw reaction
               +--rw operation [id]
                  +--rw id             event-id-type
                  +--rw type           enumeration
                  +--rw conditional
                  |  +--rw statement    string
                  |  +--rw true
                  |  |  +--rw execute
                  |  |  +--rw next-operation?   event-id-type
                  |  +--rw false
                  |     +--rw execute
                  |     +--rw next-operation?   event-id-type
                  +--rw simple
                     +--rw execute
                     +--rw next-operation?   event-id-type
```

8.2. YANG model for FSM - Tree

```
      +--rw current-state?   leafref
      +--rw states
         +--rw state [id]
            +--rw id             state-id-type
            +--rw name             string
            +--rw description?   string
            +--rw events
               +--rw event [name type]
                  +--rw name           string
                  +--rw type           event-type
                  +--rw description?   string
                  +--rw filters
                  |  +--rw filter [filter-id]
                  |     +--rw filter-id    yp:filter-id
                  +--rw reaction
                     +--rw operation [id]
                        +--rw id             event-id-type
                        +--rw type           enumeration
                        +--rw conditional
                        |  +--rw statement    string
                        |  +--rw true
                        |  |  +--rw execute
                        |  |  +--rw next-operation?   event-id-type
                        |  |  +--rw next-state?       leafref
                        |  +--rw false
                        |     +--rw execute
                        |     +--rw next-operation?   event-id-type
                        |     +--rw next-state?       leafref
                        +--rw simple
                           +--rw execute
                           +--rw next-operation?   event-id-type
                           +--rw next-state?       leafref
```

8.3. YANG model for events and actions - Code

```
 module events {
   namespace "http://sssup.it/events";
   prefix ev;

   import ietf-yang-push {
     prefix yp;
   }

   organization
     "Scuola Superiore Sant'Anna Network and Services Laboratory";

   contact
     " Editor: Matteo Dallaglio
```

```
               <mailto:m.dallaglio@sssup.it>
    ";

   description
     "This module contains a YANG definitions of events and generic
      reactions.";

   revision 2016-03-15 {
     description "Initial Revision.";
     reference
       "RFC xxxx: A YANG data model for the description of events and
        reactions";
   }

   // identity statements

   identity EVENT {
       description "Base for all types of event";
   }

   identity ON_CHANGE {
       base EVENT;
       description
         "The event when the database changes.";
   }

   // typedef statements

   typedef event-type {
     type identityref {
       base EVENT;
     }
   }

   typedef event-id-type {
     type uint32;
   }


   // grouping statements
   grouping operation-block {
     leaf id {
       type event-id-type;
     }
     leaf type {
       type enumeration {
         enum CONDITIONAL_OP;
         enum SIMPLE_OP;
```

```
      }
      mandatory true;
    }

    grouping execution-top {
      anyxml execute {
        description "Represent the action to perform";
      }
      leaf next-operation {
        type event-id-type;
        description "the id of the next operation to execute";
      }
    }

    container conditional {
      when "../type = 'CONDITIONAL_OP'";
      leaf statement {
        type string;
        mandatory true;
        description
          "The statement to be evaluated before execution.
          E.g. if a=b";
      }
      container true {
        uses execution-top;
      }
      container false {
        uses execution-top;
      }
    }

    container simple {
      when "../type = 'SIMPLE_OP'";
      description
        "Simple execution of an action without checking any condition";
      uses execution-top;
    }
  }

  grouping operation-top {
    list operation {
      key "id";
      ordered-by user;
      uses operation-block;
    }
  }

  grouping on-change {
```

```
    description
      "Event occuring when a modification of one or more
       objects occurs";

    container filters {
      description
        "This container contains a list of configurable filters
         that can be applied to subscriptions.  This facilitates
         the reuse of complex filters once defined.";
      list filter {
        key "filter-id";
        description
          "A list of configurable filters that can be applied to
           subscriptions.";
        leaf filter-id {
          type yp:filter-id;
          description
            "An identifier to differentiate between filters.";
        }
        uses yp:datatree-filter;
      }
    }
  }

  grouping event-top {
    leaf name {
      type string;
      mandatory true;
    }

    leaf type {
      type event-type;
      mandatory true;
    }

    leaf description {
      type string;
    }

    // list of all possible events
    uses on-change {
      when "type = 'ON_CHANGE'";
    }

    container reaction {
      uses operation-top;
    }
  }
```

```
   grouping events-top {
     container events {
       list event {
         key "name type";
         uses event-top;
       }
     }
   }


   // data definition statements

   uses events-top;

   // extension statements

   // feature statements

   // augment statements

   // rpc statements

   // notification statements


 }//module events
```

8.4. YANG model for FSM - Code

```
  module finite-state-machine {
    namespace "http://sssup.it/fsm";
    prefix fsm;

    import events {
      prefix ev;
    }

    organization
      "Scuola Superiore Sant'Anna Network and Services Laboratory";

    contact
      " Editor: Matteo Dallaglio
                <mailto:m.dallaglio@sssup.it>
      ";

    description
      "This module contains a YANG definitions of a generic finite state
```

```
       machine.";

    revision 2016-03-15 {
      description "Initial Revision.";
      reference
        "RFC xxxx:";
    }

    // identity statements

    // typedef statements

    typedef state-id-type {
      type uint32;
    }


    // grouping statements
    grouping state-top {
      leaf id {
        type state-id-type;
      }

      leaf name {
        type string;
      }

      leaf description {
        type string;
      }


      grouping next-state-top {
        leaf next-state {
            type leafref {
              path "../../../../../../../../../states/state/id";
            }
            description "Id of the next state";
          }
      }

      uses ev:events-top {
        augment "events/event/reaction/operation/conditional/true" {
          uses next-state-top;
        }
        augment "events/event/reaction/operation/conditional/false" {
          uses next-state-top;
        }
```

```
        augment "events/event/reaction/operation/simple" {
          //uses next-state-top;
          leaf next-state {
            type leafref {
              path "../../../../../../../../states/state/id";
            }
            description "Id of the next state";
          }
        }
      }

    }



    grouping states-top {
      leaf current-state {
        type leafref {
          path "../states/state/id";
        }
      }

      container states {
        list state {
          key "id";
          uses state-top;
        }
      }
    }


    // data definition statements



    uses states-top;

    // extension statements

    // feature statements

    // augment statements.

    // rpc statements

    // notification statements
```

```
}//module fsm
```

8.5. Example of values for the YANG model

FIELD NAME	YANG DATA TYPE	VALUE
Current State	leafref	"an existing state id in the FSM"
State		
id	uint32	1
name	string	Steady
description	string	"whatever string"
event		
name	string	"whatever string"
type	enum	BER_CHANGE
description	string	"whatever string"
filter		
filter-id	uint32	2
filter-type	anyxml or xpath	BER>0.0009
reaction		
id	uint32	3
type	enum	SIMPLE
statement	string	"whatever string"
execute	anyxml	"this recalls an RPC where the FEC value is expressed"
next-operation	uint32	NULL
next-state	leafref	"an existing state id in the FSM"

9. Acknowledgements

This work has been partially supported by the European Commission through the H2020 ORCHESTRA (Optical peRformanCe monitoring enabling dynamic networks using a Holistic cross-layEr, Self-configurable Truly flexible approach, grant agreement no: H2020-645360) project. The views expressed here are those of the authors only. The European Commission is not liable for any use that may be made of the information in this document.

10. Security Considerations

TBD

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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

[RFC7491] King, D. and A. Farrel, "A PCE-Based Architecture for Application-Based Network Operations", RFC 7491, DOI 10.17487/RFC7491, March 2015, <http://www.rfc-editor.org/info/rfc7491>.

11.2. Informative References

[I-D.ietf-i2rs-yang-network-topo] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A Data Model for Network Topologies", draft-ietf-i2rs-yang-network-topo-13 (work in progress), June 2017.

[I-D.vergara-ccamp-flexigrid-yang] Madrid, U., Perdices, D., Lopezalvarez, V., Dios, O., King, D., Lee, Y., and G. Galimberti, "YANG data model for Flexi-Grid Optical Networks", draft-vergara-ccamp-flexigrid-yang-04 (work in progress), March 2017.

[I-D.zhang-ccamp-l1-topo-yang] zhenghaomian@huawei.com, z., Fan, Z., Sharma, A., and X. Liu, "A YANG Data Model for Optical Transport Network Topology", draft-zhang-ccamp-l1-topo-yang-07 (work in progress), April 2017.

Authors' Addresses

Nicola Sambo
Scuola Superiore Sant'Anna
Via Moruzzi 1
Pisa 56124
Italy

Email: nicola.sambo@sssup.it

Matteo Dallaglio
Scuola Superiore Sant'Anna
Via Moruzzi 1
Pisa 56124
Italy

Email: matteo.dallaglio@sssup.it

Piero Castoldi
Scuola Superiore Sant'Anna
Via Moruzzi 1
Pisa 56124
Italy

Email: piero.castoldi@sssup.it

Giuseppe Fioccola
Telecom Italia
Via Reiss Romoli, 274
Torino 10148
Italy

Email: giuseppe.fioccola@telecomitalia.it

Andrea Di Giglio
Telecom Italia
Via Reiss Romoli, 274
Torino 10148
Italy

Email: andrea.digiglio@telecomitalia.it

Filippo Cugini
CNIT
Via Moruzzi 1
Pisa 56124
Italy

Email: filippo.cugini@cnit.it

Giacomo Bernini
Nextworks
Via Livornese 1027
Pisa 56122
Italy

Email: g.bernini@nextworks.it

Pietro G. Giardina
Nextworks
Via Livornese 1027
Pisa 56122
Italy

Email: p.giardina@nextworks.it