GWD-Draft
NSI-WG
nsi-wg@ogf.org

John MacAuley, SURFnet

July 7, 2013
Updated: July 23, 2013

NSI-CS Service Decoupling

Status of This Document

Grid Working Document (GWD)

Document Change History

Date	Change
June 28, 2013	Initial draft of service decoupling proposal.
July 5, 2013	Added serviceType elelment to identify Service Description template.
July 7, 2013	Removed serviceAttributes element and included external namespace service elements in body of criteria element.
July 23, 2013	Added decoupled service error proposal and reservation modification handling.

Abstract

In NSI CS version 2.0 focused was placed on a number of improvements, including two-phase reserve/modify, better support for service activation, more complete state machine definition, and clearing up ambiguities around STP definition. All these improvements were an important step along the path to deploying production services, however, little focus was actually placed on the definition of the service itself. This proposal discusses the current service offering supported by NSI CS version 2.0, and a modification to allow for more flexibility in the protocol, allowing for multiple new services to be offered without the need to update the existing base NSI CS protocol. In addition, we resurrect the Service Description concept and bring it fully into the protocol specification.

Contents

John MacAuley, SURFnet

1 Introduction

Discuss the advantages of service decoupling from the core NSI-CS protocol.

2 Current NSI-CS service mechanism

In NSI CS version 1.x and 2.0 only unidirectional and bidirectional point-to-point services are offered as part of the protocol. In fact, the NSI working group violated most common protocol design principles by hard coding the service offering directly into the core protocol specification, when in fact, the NSI CS protocol was meant to support many service types. We will now discuss where we have made these easy to correct mistakes.

The base NSI CS *reserve* operation, as shown in Figure 1 below, is extremely generic from a service point of view, containing only base identification and descriptive information. The specifics of the service reservation request are held within the criteria element.

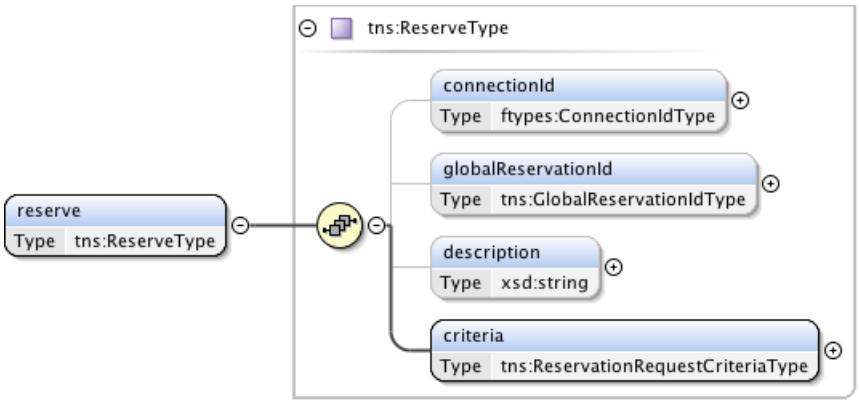


Figure 1 - Reserve request operation.

Exploding the criteria element in Figure 2 we can see the details of the service request. The ***version*** attribute represents the version of the reservation and is not related to the type of service being requested. The ***schedule*** element contains the reservation start and end time criteria, providing a generic scheduling capability independent of the service type being offered. The ***serviceAttributes*** element is also a generic structure allowing for the specification of parameters applying to the service reservation. At the moment, the ***serviceAttributes*** allows for generic type/value strings, or inclusion of elements from external namespaces. Where we run into a problem is with the ***bandwidth*** and ***path*** criteria elements.

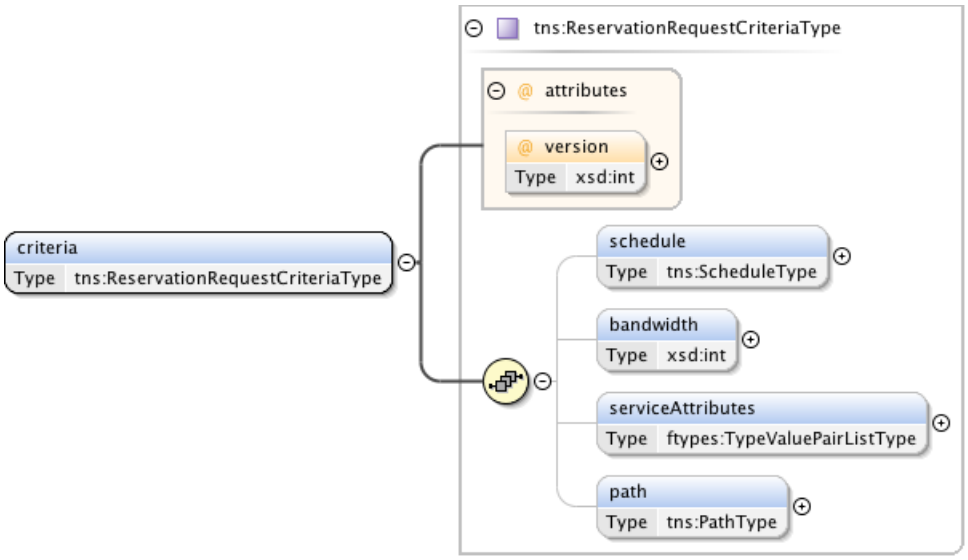


Figure 2 – Service criteria element.

As currently defined, ***bandwidth*** and ***path*** restrict a reservation request to either a unidirectional or symmetric bidirectional point-to-point service. Figure 3 shows details of the reservation ***path*** object.

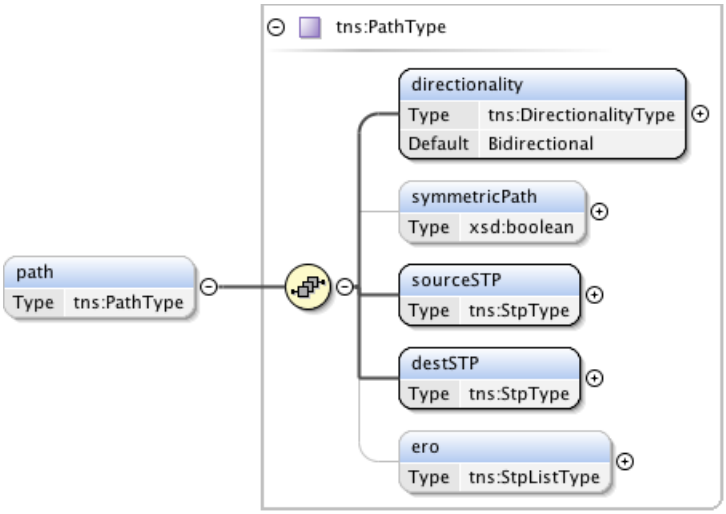


Figure 3 – Service path element.

It should be noted that the ***symmetricPath*** element applies to only the bidirectional service specification, and indicates whether unidirectional components of the bidirectional service must follow the same path (true) or are free to take diverse paths (false).[1]

As we can see from the current definition, any time we would like to add new services, or extend/correct an issue in an existing service, we need to modify the core NSI CS protocol

[1] Do not confuse this attribute with bandwidth symmetry as only symmetric bidirectional services are supported.

definition. This is an extremely expensive proposition for NSI as a standard body, as well as NSA implementations for the churn in the core protocol. Unfortunately, we have built what could be termed a brittle protocol with respect to offered services.

3 Decoupled services

Our primary goal should be to remove the dependencies of service specification from the core NSI CS protocol. This will allow the existing NSI CS protocol to remain stable while permitting changes to the services offered by NSA within the network. Fortunately, with XML and a small change to the structure of our existing NSI CS protocol we can achieve this exact behavior.

3.1 Using XML schema ANY element

The key to this change is the use the XML ***ANY*** mechanism similar to the ***serviceAttributes*** element within the reservation criteria. This element has already been used by some NSA implementations to specify additional service parameters within their client requests. For example, SURFnet uses these parameters to allow customers to request Layer 1 sub network protection for their NSI-CS services. The following XML snippet would appear in the ***criteria*** element of an NSI CS v2.0 reservation request:[2]

```
<serviceAttributes>
   <surf:sNCP xmlns:surf="http://schemas.surfnet.nl/nsi/2013/04/services">Protected</surf:sNCP>
</serviceAttributes>
```

In this case, we are taking advantage of the flexible ***ANY*** element feature within XML that permits inclusion of elements from external namespaces, or in this specific case, we are seeing an element called “sNCP” from the SURFnet schema namespace. The SURFnet NSA knows how to interpret elements from their services namespace, and therefore, can provide additional capabilities outside the core NSI CS protocol. No extensions to the core NSI protocol were required.

3.2 Restructuring *criteria* element

The proposal is to remove the point-to-point specific ***bandwidth***, ***path*** elements from the ***criteria*** element and place them into a separate service specific schema definition with its own dedicated namespace, and add an ***ANY*** child element to the ***criteria*** element to allow generic inclusion of external service specifications. With this new ***ANY*** child element we also remove the need for the ***serviceAttributes*** element as the capability is incorporated into the base ***criteria*** element. In addition to the service specification decoupling, we introduce a new string element called ***serviceType***, which will be described in the next section.

Incorporating these described changes, the new ***criteria*** element would be simplified to what is now shown below in Figure 4.

[2] This assumes the discussed simplification of the ***ServiceAttributesType***.

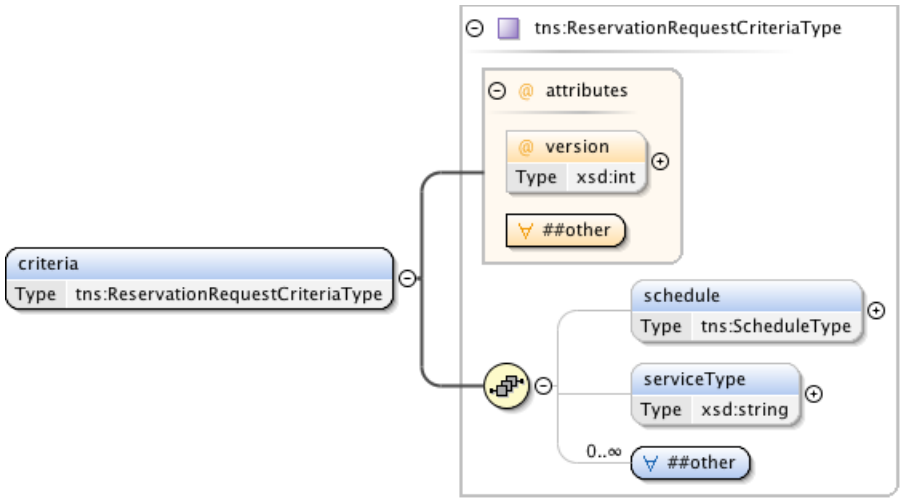
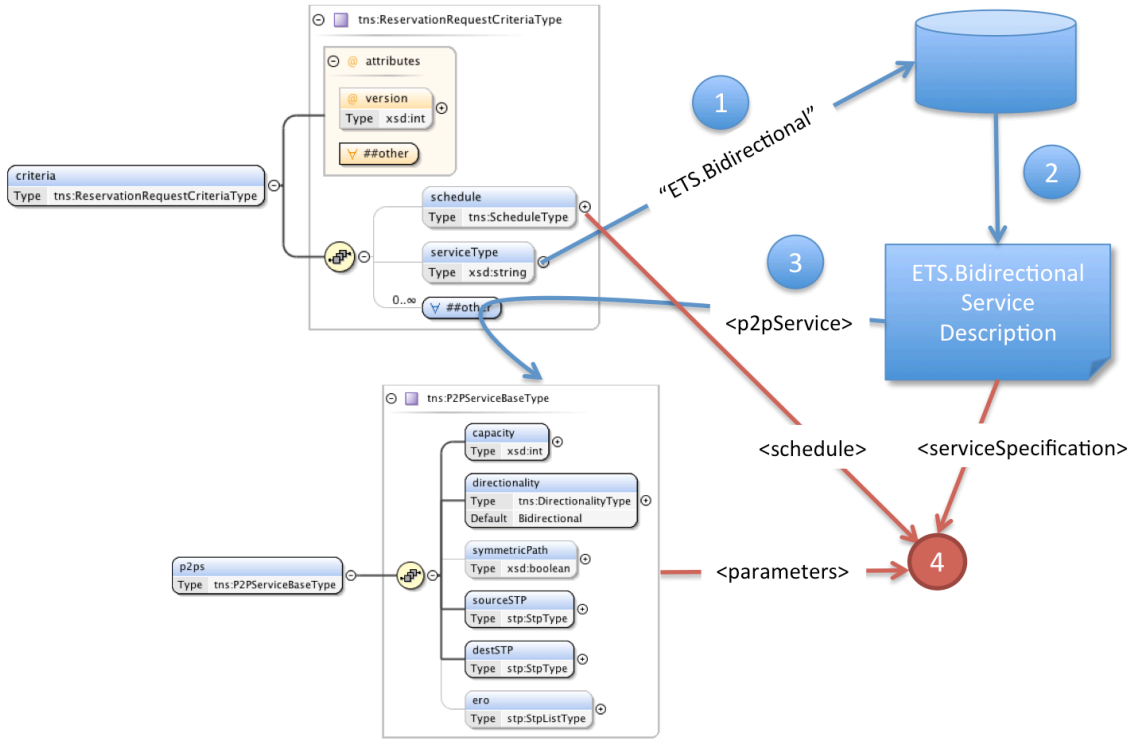


Figure 4 – New reservation ***criteria*** element.

3.3 The *serviceType* element

The ***serviceType*** element will relay the specific service type being requested in the reservation. This service type string maps to a specific Service Description template defined by the network providers describing the type of service offered, parameters supported in a reservation request (mandatory and optional), defaults for parameters if not specified (as well as maximums and minimums), and other attributes relating to the service offering. The NSA in turn uses this information to determine the specific service parameters carried in the ***criteria*** element required to specify the requested service.

The Service Description template is an important component in the solution, linking the opaque information carried in the NSI CS protocol to the concrete parameters needed to satisfy a specific service request. Figure 5 is a pictorial view of how the ***serviceType*** maps through the Service Description template, identifying the mandatory and optional service elements needed to satisfy the request.



1. Extract ***serviceType*** value from incoming reservation request and lookup Service Description corresponding to ***serviceType***.
2. Use Service Description to determine the service elements needed for the specific service requested and any other service related parameters.
3. Extract specific services elements from ***criteria*** as described in Service Description.
4. Process service request using supplied service parameters and service template information.

Figure 5 – Using ***serviceType*** to determine required and optional service elements.

3.4 Service specific schema

An interesting side effect of this service decoupling is the core NSI-CS protocol specification is simplified with the removal of the service specific types that now go into a new service schema. Figure 6 shows a repackaged ***p2ps*** element that will be included in ***criteria*** when a generic point-to-point service is requested (notice ***bandwidth*** has been renamed to ***capacity***).

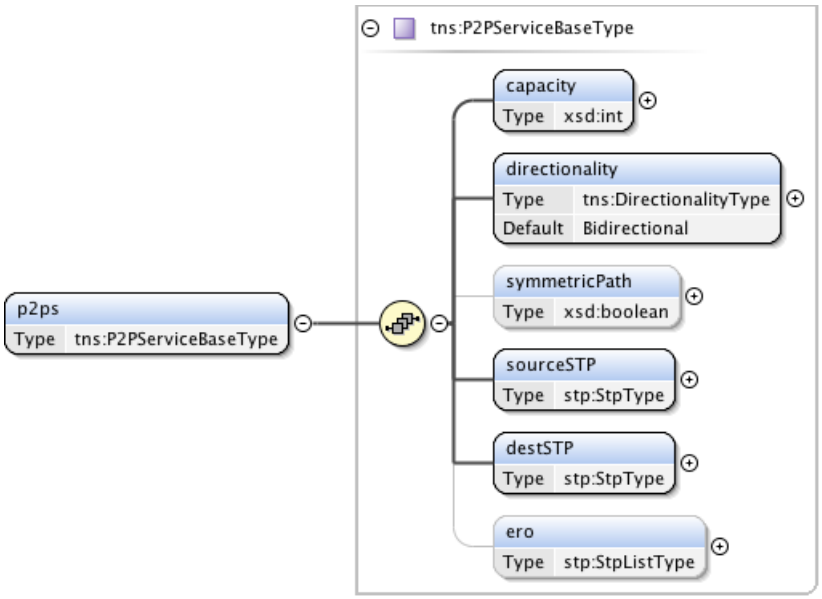


Figure 6 – A point-to-point service definition.

Below we see an example *reserve* request XML message for a bidirectional service as currently defined in NSI CS version 2.0. Notice the ***bandwidth*** and ***path*** elements are members of the ***criteria*** element. Also notice that ***serviceAttributes*** contains the SURFnet specific ***sNCP*** element as an example of including from an external namespace.

```
<nsi:reserve
  xmlns:nsi="http://schemas.ogf.org/nsi/2013/04/connection/types"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:surf="http://schemas.surfnet.nl/nsi/2013/04/services">

  <connectionId>urn:uuid:4b4a71d0-3c71-47cf-a646-beacb14a4c72</connectionId>
  <globalReservationId>urn:uuid:83fe4f36-5b38-41b6-bc46-a362a06a54ee</globalReservationId>
  <description> My example reservation with existing NSI CS 2.0 schema</description>
  <criteria version="1">
    <schedule>
      <startTime>2013-09-30T09:30:10Z</startTime>
      <endTime>2013-09-30T10:30:10Z</endTime>
    </schedule>
    <bandwidth>1000</bandwidth>
    <serviceAttributes>
      <attribute>
        <surf:sNCP>Protected</surf:sNCP>
      </attribute>
    </serviceAttributes>
    <path>
      <directionality>Bidirectional</directionality>
      <symmetricPath>true</symmetricPath>
      <sourceSTP>
        <networkId>urn:ogf:network:netherlight.net:2012</networkId>
        <localId> urn:ogf:network:netherlight.net:2012:uvalight-netherlight</localId>
        <labels>
          <attribute type="vlan">
            <value>1901</value>
          </attribute>
```

```
        </labels>
      </sourceSTP>
      <destSTP>
        <networkId>urn:ogf:network:netherlight.net:2012</networkId>
        <localId> urn:ogf:network:netherlight.net:2012:netherlight-czechlight</localId>
        <labels>
          <attribute type="vlan">
            <value>1901</value>
          </attribute>
        </labels>
      </destSTP>
    </path>
  </criteria>
</nsi:reserve>
```

Below we have the proposed restructured *reserve* request XML message:

```
<nsi:reserve xmlns:nsi="http://schemas.ogf.org/nsi/2013/04/connection/types"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:p2p="http://schemas.ogf.org/nsi/2013/04/services/point2point"
  xmlns:surf="http://schemas.surfnet.nl/nsi/2013/04/services">

  <connectionId>urn:uuid:4b4a71d0-3c71-47cf-a646-beacb14a4c72</connectionId>
  <globalReservationId>urn:uuid:83fe4f36-5b38-41b6-bc46-a362a06a54ee</globalReservationId>
  <description>My example reservation using decoupled service schema</description>
  <criteria version="1">
    <schedule>
      <startTime>2013-09-30T09:30:10Z</startTime>
      <endTime>2013-09-30T10:30:10Z</endTime>
    </schedule>
    <p2p:p2ps>
      <capacity>1000</capacity>
      <directionality>Bidirectional</directionality>
      <symmetricPath>true</symmetricPath>
      <sourceSTP>
        <networkId>urn:ogf:network:netherlight.net:2012</networkId>
        <localId> urn:ogf:network:netherlight.net:2012:uvalight-netherlight</localId>
        <labels>
          <attribute type="vlan">
            <value>1901</value>
          </attribute>
        </labels>
      </sourceSTP>
      <destSTP>
        <networkId>urn:ogf:network:netherlight.net:2012</networkId>
        <localId> urn:ogf:network:netherlight.net:2012:netherlight-czechlight</localId>
        <labels>
          <attribute type="vlan">
            <value>1901</value>
          </attribute>
        </labels>
      </destSTP>
    </p2p:p2ps>
    <surf:sNCP>Protected</surf:sNCP>
  </criteria>
</nsi:reserve>
```

These messages look quite similar, but there are some key differences:

- The ***p2p*** namespace is defined in the ***reserve*** element using a unique URL defining the service XSD document. We have encapsulated all types needed for this service in that XSD document.

```
xmlns:p2p="http://schemas.ogf.org/nsi/2013/04/services/point2point"
```

- The ***capacity*** and ***path*** elements members are now part of the ***p2ps*** element included within the ***criteria*** element. These attributes are part of the service specification, and therefore, separate from the core reservation criteria. We have effectively decoupled the attributes of a service from the core NSI protocol.

- The ***serviceType*** element is added to identify the desired service requested and will identify the specific service elements carried in ***criteria***.

- The ***serviceAttributes*** element was removed and the ANY functionality placed directly into the criteria element to simplify specification of these external namespace elements.

- Multiple service attributes can be specified as show by inclusion of both the ***p2ps*** and SURFnet specific ***sNCP*** elements.

In addition to these changes, the restructuring of the ***criteria*** element would be done in the *reserveConfirmed* and *query* messages.

3.5 Reservation modification

For a base point-to-point service specification we support the modification of ***schedule*** (start or end time), as well as the ***capacity*** of the service. The ***schedule*** element is within the core ***criteria*** element, and remains as is, specifying a change in the combination of ***startTime*** and ***endTime*** as desired. For the external service schema, only the elements to be modified are included in the request. These will be defined as separate elements within their schema definition for inclusion as modifiable items.

Below is an example *reserve* modification request XML message where we are requesting a modification to the ***capacity*** parameter of the reservation:

```
<nsi:reserve xmlns:nsi="http://schemas.ogf.org/nsi/2013/04/connection/types"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:p2p="http://schemas.ogf.org/nsi/2013/04/services/point2point"
  xmlns:surf="http://schemas.surfnet.nl/nsi/2013/04/services">

  <connectionId>urn:uuid:4b4a71d0-3c71-47cf-a646-beacb14a4c72</connectionId>
  <criteria version="2">
    <p2p:capacity>500</p2p:capacity>
  </criteria>
</nsi:reserve>
```

3.6 Service error handling

The NSI-CS protocol utilizes the ***ServiceExceptionType*** structure to commonly convey error information in both *SOAP faults* and operation *Failed* messages. The structure is extremely

flexible and able to handle both simple high-level error information, as well as detailed errors down to the individual attribute value causing a problem. The current ***ServiceExceptionType*** is show in Figure 7.

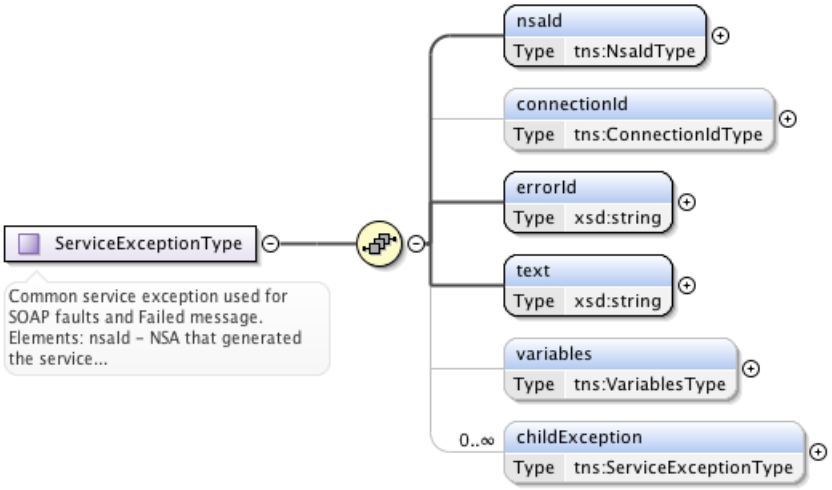


Figure 7 – Current ***ServiceExceptionType*** definition.

Parameter	M/O	Description
nsaId	M	NSA that generated the service exception.
connectionId	O	The connectionId associated with the reservation impacted by this error.
errorId	M	Error identifier uniquely identifying each known fault within the protocol.
text	M	User-friendly message text describing the error.
variables	O	An optional collection of type/value pairs providing additional information relating to the error.
childException	O	Hierarchical list of service exceptions capturing failures within the request tree.

NSI-CS uses a hierarchal error code structure to group related error codes together under a common parent error code value. For example, the following table demonstrates the generic **TOPOLOGY_ERROR** parent code and four more specific children topology related error codes.

text	errorId	Description
TOPOLOGY_ERROR	00400	Parent error classification.
UNKNOWN_STP	00401	Could not find STP in topology database.
STP_RESOLUTION_ERROR	00402	Could not resolve STP to a managing NSA.
NO_PATH_FOUND	00403	Path computation failed to resolve route for reservation.
VLANID_INTERCANGE_NOT_SUPPORTED	00404	VLAN interchange not supported for requested path.

Table 1 – Topology related error codes.

Below we see an example XML segment for the ***serviceException*** element generated when an invalid STP was specified in the ***sourceSTP*** element of the reserve request operation.

```
<serviceException>
  <nsaId>urn:ogf:network:netherlight.net:2012:nsa</nsaId>
  <connectionId>urn:uuid:4b4a71d0-3c71-47cf-a646-beacb14a4c72</connectionId >
  <errorId>00401</errorId>
  <text>UNKNOWN_STP</text>
  <variables>
```

```
        <variable type="sourceSTP.localId">
          <value>urn:ogf:network:netherlight.net:2012:stp:ams-uva-83</value>
        </variable>
      </variables>
    </serviceException>
```

In this example we see the ***nsaId*** of the NSA generating the error, the ***connectionId*** associated with the error, the ***errorId*** and ***text*** for an unknown STP, and the ***variables*** section that contains the parameter causing the issue.

Unfortunately, as can be seen with the **UNKNOWN_STP**, **STP_RESOLUTION_ERROR**, and **VLANID_INTERCANGE_NOT_SUPPORTED** error codes, we have service specific error information in the current error-handling framework. The concept of STP and VLANID are currently specific to the point-to-point Ethernet service specification. As new services are offered, and existing ones modified, these errors will need to be modified or extended. This is not a desirable feature of the existing error handling strategy. For this reason we need to provide a decoupled service specific error solution.

In Figure 8 we provide a solution to the problem by extending the existing ***ServiceExceptionType*** to include an optional ***serviceError*** element.

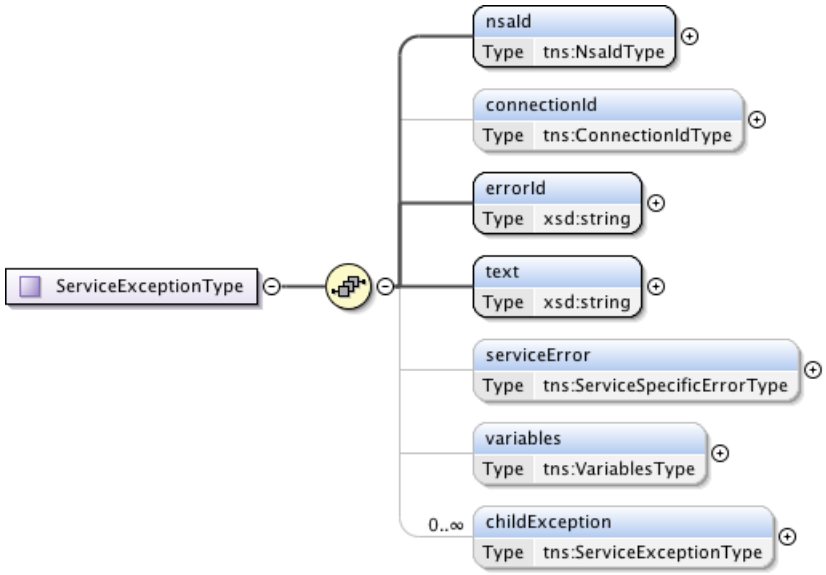


Figure 8 – Service decoupled error ***ServiceExceptionType*** definition.

The ***serviceError*** element is only included when there is a service specific error generated by an NSA. As seen in Figure 9, this new element contains the service specific ***serviceType*** element mapping to the service description used for the service request[3], a service specific ***serviceErrorId*** defined in the service specification, and a user readable ***serviceErrorText*** element.

[3] The ***serviceType*** is included since the original ***serviceType*** specified in the *reserve* request may have been re-mapped into a different ***serviceType*** when sent to a child NSA.

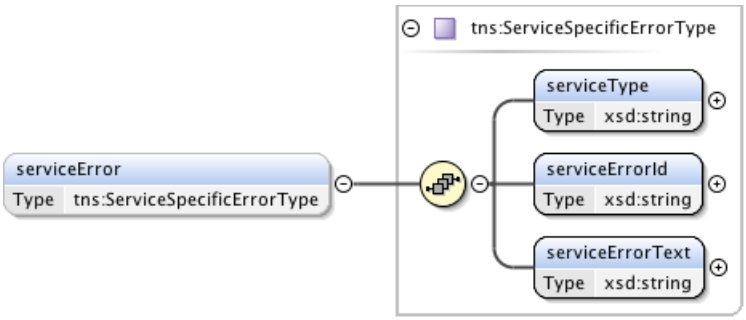


Figure 9 – New ***ServiceSpecificErrorType*** definition.

We also modify the existing error codes to remove service specific values, and introduce a generic **SERVICE_ERROR** parent code to indicate the existence of a service specific error. Standard errors can be used when applicable, but the service specific errors are available if service specific information relating to the failure must be conveyed to the parent NSA.

text	errorId	Description
SERVICE_ERROR	00700	Parent error classification for a service specific error.

When new services are defined three pieces of information must now be generated:

1. Service specific XML schema if not reusing an existing schema.
2. Service description document.
3. Service error code definitions.

For example, we can repackage the existing service errors into a service specific list. It is recommended that a similar hierarchy to the existing errors be maintained for consistency, including parent classifications. The table below is an example of a service specific error list repackaging the existing NSI-CS point-to-point service errors. Those errors considered of the TOPOLOGY_ERROR classification get a number under the 00400 grouping, and likewise, those considered RESOURCE_ERROR related get a number 00600 under the 00400 grouping. Any service specific errors that do not fall under the existing allocated hierarchy can be assigned a number under the 00700 grouping.

text	errorId	Description
UNKNOWN_STP	00401	Could not find STP in topology database.
STP_RESOLUTION_ERROR	00402	Could not resolve STP to a managing NSA.
VLANID_INTERCANGE_NOT_SUPPORTED	00404	VLAN interchange not supported for requested path.
STP_UNAVALABLE	00601	Specified STP already in use.
BANDWIDTH_UNAVAILABLE	00602	Insufficient bandwidth available for reservation.

Table 2 – NSI-CS point-to-point service specific errors.

Using the previous service error example we would now repackage a service specific error as follows:

```
<serviceException>
  <nsaId>urn:ogf:network:netherlight.net:2012:nsa/</nsaId>
  <connectionId>urn:uuid:4b4a71d0-3c71-47cf-a646-beacb14a4c72</connectionId >
  <errorId>00700</errorId>
  <text>SERVICE_ERROR</text>
  <serviceError>
    <serviceType>ETS.SURFnet</serviceType>
    <serviceErrorId>00401</serviceErrorId>
    <serviceErrorText>UNKNOWN_STP</serviceErrorText>
  </serviceError>
```

```
    <variables>
      <variable type="sourceSTP">
        <value>urn:ogf:network:netherlight.net:2012:stp:ams-uva-83</value>
      </variable>
    </variables>
  </serviceException>
```

4 Summary

The proposed decoupling changes are an improvement over the current tight coupling of the point-to-point service definition within the NSI CS version 2.0 protocol specification. Refactoring the service elements into an external namespace and including through an ***ANY*** definition within the ***criteria*** element does not change the behavioral aspects of the protocol, nor the information carried in the messages, as this is just a syntactic change to where the data is carried. There will be minimal impact current NSA implementation of NSI CS version 2.0.

5 Contributors

The following people contributed to the content of this document:

John MacAuley
Network Services, SURFnet
The Netherlands
Email: john@surfnet.nl

6 Acknowledgments

The author would like to thank the color blue for all your wonderful contributions to the world.

7 Intellectual Property Statement

8 Disclaimer

9 Full Copyright Notice

10 Appendix – NSI-CS Point-to-Point Service Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    The OGF takes no position regarding the validity or scope of any intellectual property
or other rights that might be claimed to pertain to the implementation or use of the
technology described in this document or the extent to which any license under such rights
might or might not be available; neither does it represent that it has made any effort to
identify any such rights.  Copies of claims of rights made available for publication and
any assurances of licenses to be made available, or the result of an attempt made to
obtain a general license or permission for the use of such proprietary rights by
implementers or users of this specification can be obtained from the OGF Secretariat.

    The OGF invites any interested party to bring to its attention any copyrights, patents
or patent applications, or other proprietary rights which may cover technology that may be
required to practice this recommendation.  Please address the information to the OGF
Executive Director.

    This document and the information contained herein is provided on an "As Is" basis and
the OGF disclaims all warranties, express or implied, including but not limited to any
warranty that the use of the information herein will not infringe any rights or any
implied warranties of merchantability or fitness for a particular purpose.

    Copyright (C) Open Grid Forum (2009-2012). All Rights Reserved.

    This document and translations of it may be copied and furnished to others, and
derivative works that comment on or otherwise explain it or assist in its implementation
may be prepared, copied, published and distributed, in whole or in part, without
restriction of any kind, provided that the above copyright notice and this paragraph are
included on all such copies and derivative works. However, this document itself may not be
modified in any way, such as by removing the copyright notice or references to the OGF or
other organizations, except as needed for the purpose of developing Grid Recommendations
in which case the procedures for copyrights defined in the OGF Document process must be
followed, or as required to translate it into languages other than English.

    The limited permissions granted above are perpetual and will not be revoked by the OGF
or its successors or assignees.

    Open Grid Forum NSI Connection Services Protocol v2.0
-->
<xsd:schema targetNamespace="http://schemas.ogf.org/nsi/2013/04/services/point2point"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:ftypes="http://schemas.ogf.org/nsi/2013/04/framework/types"
    xmlns:types="http://schemas.ogf.org/nsi/2013/04/services/types"
    xmlns:tns="http://schemas.ogf.org/nsi/2013/04/services/point2point">

    <xsd:annotation>
        <xsd:documentation xml:lang="en">
        </xsd:documentation>
```

```
    </xsd:annotation>

    <!-- Import the common NSI framework types. -->
    <xsd:import namespace="http://schemas.ogf.org/nsi/2013/04/framework/types"
        schemaLocation="ogf_nsi_framework_types_v2_0.xsd"/>

    <!-- Import the common NSI framework types. -->
    <xsd:import namespace="http://schemas.ogf.org/nsi/2013/04/services/types"
        schemaLocation="ogf_nsi_services_types_v2_0.xsd"/>

    <xsd:element name="p2ps" type="tns:P2PServiceBaseType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                This Point-to-Point Service element is used to specify a generic
                point-to-point service request in the NSI CS protocol.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:element>

    <xsd:element name="ets" type="tns:EthernetBaseType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                This Ethernet Transport Service element is used to specify a
                point-to-point Ethernet service request in the NSI CS protocol.
            </xsd:documentation>
        </xsd:annotation>
    </xsd:element>

    <!-- These are the modifiable parameters within this service. -->

    <xsd:element name="capacity" type="xsd:int" />

    <xsd:complexType name="P2PServiceBaseType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                Type defining a generic point-to-point service specification.
                At the moment this type supports a unidirectional or symmetric
                bidirectional service.

                Elements:

                capacity - Capacity of the service in Mb/s.

                directionality - The (uni or bi) directionality of the service.

                symmetricPath - An indication that both directions of a bidirectional
                circuit must fallow the same path.  Only applicable when
                directionality is "Bidirectional".  If not specified then value
                is assumed to be false.

                sourceSTP - Source STP of the service.

                destSTP - Destination STP of the service.

                ero - Hop-by-hop ordered list of STP from sourceSTP to
                destSTP. List does not include sourceSTP and destSTP.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="capacity"        type="xsd:int" />
            <xsd:element name="directionality"  type="tns:DirectionalityType"
default="Bidirectional" />
            <xsd:element name="symmetricPath"   type="xsd:boolean" minOccurs="0" />
            <xsd:element name="sourceSTP"       type="types:StpType" />
            <xsd:element name="destSTP"         type="types:StpType" />
            <xsd:element name="ero"             type="types:StpListType" minOccurs="0" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="EthernetBaseType">
        <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">
                Point-to-Point Ethernet service definition.

                Elements:

                capacity - Capacity of the service in Mb/s including ethernet
                framing headers.

                directionality - The (uni or bi) directionality of the service.

                symmetricPath - An indication that both directions of a bidirectional
                circuit must fallow the same path.  Only applicable when
                directionality is "Bidirectional".  If not specified then value
                is assumed to be false.

                sourceSTP - Source STP of the service.

                destSTP - Destination STP of the service.

                ero - Hop-by-hop ordered list of STP from sourceSTP to
                destSTP. List does not include sourceSTP and destSTP.

                mtu - Specifies the maximum transmission unit size in bits.

                burstsize - Specifies the maximum number of bits that can be
                send to the interface before the sender must wait before
                sending again.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:complexContent>
            <xsd:extension  base="tns:P2PServiceBaseType">
                <xsd:sequence>
                    <xsd:element name="mtu"       type="xsd:int" minOccurs="0" />
                    <xsd:element name="burstsize" type="xsd:int" minOccurs="0" />
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

    <xsd:simpleType name="DirectionalityType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                The directionality of the requested data service.  Possible values
                are "Bidirectional" for a bidirectional data service, and
                "Unidirectional" for a unidirectional data service.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="Bidirectional" />
            <xsd:enumeration value="Unidirectional" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:schema>
```

11 Appendix – NSI-CS Service Types Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    The OGF takes no position regarding the validity or scope of any intellectual property
or other rights that might be claimed to pertain to the implementation or use of the
technology described in this document or the extent to which any license under such rights
might or might not be available; neither does it represent that it has made any effort to
identify any such rights.  Copies of claims of rights made available for publication and
any assurances of licenses to be made available, or the result of an attempt made to
obtain a general license or permission for the use of such proprietary rights by
implementers or users of this specification can be obtained from the OGF Secretariat.

    The OGF invites any interested party to bring to its attention any copyrights, patents
or patent applications, or other proprietary rights which may cover technology that may be
```

```
required to practice this recommendation.  Please address the information to the OGF
Executive Director.

    This document and the information contained herein is provided on an "As Is" basis and
the OGF disclaims all warranties, express or implied, including but not limited to any
warranty that the use of the information herein will not infringe any rights or any
implied warranties of merchantability or fitness for a particular purpose.

    Copyright (C) Open Grid Forum (2009-2012). All Rights Reserved.

    This document and translations of it may be copied and furnished to others, and
derivative works that comment on or otherwise explain it or assist in its implementation
may be prepared, copied, published and distributed, in whole or in part, without
restriction of any kind, provided that the above copyright notice and this paragraph are
included on all such copies and derivative works. However, this document itself may not be
modified in any way, such as by removing the copyright notice or references to the OGF or
other organizations, except as needed for the purpose of developing Grid Recommendations
in which case the procedures for copyrights defined in the OGF Document process must be
followed, or as required to translate it into languages other than English.

    The limited permissions granted above are perpetual and will not be revoked by the OGF
or its successors or assignees.

    Open Grid Forum NSI Connection Services Protocol v2.0
-->
<xsd:schema targetNamespace="http://schemas.ogf.org/nsi/2013/04/services/types"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:ftypes="http://schemas.ogf.org/nsi/2013/04/framework/types"
    xmlns:tns="http://schemas.ogf.org/nsi/2013/04/services/types">

    <xsd:annotation>
        <xsd:documentation xml:lang="en">
        </xsd:documentation>
    </xsd:annotation>

    <!-- Import the common NSI framework types. -->
    <xsd:import namespace="http://schemas.ogf.org/nsi/2013/04/framework/types"
        schemaLocation="ogf_nsi_framework_types_v2_0.xsd"/>

    <xsd:element name="stp" type="tns:StpType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
            </xsd:documentation>
        </xsd:annotation>
    </xsd:element>

    <xsd:element name="stpList" type="tns:StpListType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
            </xsd:documentation>
        </xsd:annotation>
    </xsd:element>

    <xsd:complexType name="StpType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                The Service Termination Point (STP) type used for path selection.

                Elements:

                networkId - A globally unique identifier (URN) that identifies the
                Network.  Rather than forcing parsing of an STP to determine the
                Network, a separate Network object is defined to allow an
                intermediate NSA to forward the message to the target Network
                without needing to know about the STPs within that domain.

                localId - A locally unique identifier for the STP within the
                associated network.

                labels - Technology specific attributes associated with
                the Service Termination Point.
```

```
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="networkId"       type="xsd:string" />
            <xsd:element name="localId"         type="xsd:string" />
            <xsd:element name="labels"          type="ftypes:TypeValuePairListType"
minOccurs="0" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="StpListType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                A simple ordered list type of Service Termination Point (STP). List
                order is determined by the integer order attribute in the orderedSTP
                element.

                Elements:

                orderedSTP - A list of STP ordered 0..n by their integer order attribute.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="orderedSTP" type="tns:OrderedStpType" minOccurs="0"
maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="OrderedStpType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                A Service Termination Point (STP) that can be ordered in a list for
                use in PathObject definition.

                Attributes:

                order - Order attribute is provided only when the STP is part of an
                orderedStpList.

                Elements:

                stp - The Service Termination Point (STP).
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="stp" type="tns:StpType" />
        </xsd:sequence>
        <xsd:attribute   name="order" type="xsd:int" use="required" />
    </xsd:complexType>
</xsd:schema>
```

12 Appendix – NSI-CS Framework Types Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
The OGF takes no position regarding the validity or scope of any intellectual property or
other rights that might be claimed to pertain to the implementation or use of the
technology described in this document or the extent to which any license under such rights
might or might not be available; neither does it represent that it has made any effort to
identify any such rights.  Copies of claims of rights made available for publication and
any assurances of licenses to be made available, or the result of an attempt made to
obtain a general license or permission for the use of such proprietary rights by
implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or
patent applications, or other proprietary rights which may cover technology that may be
required to practice this recommendation.  Please address the information to the OGF
Executive Director.
```

Open Grid Forum NSI Framework v2.0

Description: This is the NSI framework common types schema for the reference web services implementation of the OGF NSI Framework v2.0. Comments and questions can be directed to the mailing list group mailing list (nsi-wg@ogf.org).
-->

```
<xsd:schema targetNamespace="http://schemas.ogf.org/nsi/2013/04/framework/types"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:tns="http://schemas.ogf.org/nsi/2013/04/framework/types">

    <xsd:annotation>
        <xsd:documentation xml:lang="en">
            This is an XML schema document describing the OGF NSI Framework
            common types v2.0.
        </xsd:documentation>
    </xsd:annotation>

    <!-- ****************************************************************
         *             Complex XML types used in message elements       *
         **************************************************************** -->

    <xsd:element name="serviceException" type="tns:ServiceExceptionType" />

    <xsd:complexType name="ServiceExceptionType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                Common service exception used for SOAP faults and Failed
                message.

                Elements:

                nsaId - NSA that generated the service exception.

                connectionId - The connectionId associated with the reservation
                impacted by this error.

                errorId - Error identifier uniquely identifying each known
                fault within the protocol.  Acts as a parent functionality
                classification for service specific errors.

                text - User friendly message text describing the error.

                serviceError - An optional service specific error element
                defining the serviceType and service specific error Id.  Must
                be included if this error is specific to the service.

                variables - An  optional collection of type/value pairs providing
                additional information relating to the error.

                childException - Hierarchical list of service exceptions
                capturing failures within the request tree.
```

```
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="nsaId"          type="tns:NsaIdType" />
            <xsd:element name="connectionId"   type="tns:ConnectionIdType" minOccurs="0"
/>
            <xsd:element name="errorId"        type="xsd:string" />
            <xsd:element name="text"           type="xsd:string" />
            <xsd:element name="serviceError"   type="tns:ServiceSpecificErrorType"
minOccurs="0" />
            <xsd:element name="variables"      type="tns:VariablesType" minOccurs="0" />
            <xsd:element name="childException" type="tns:ServiceExceptionType"
minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="ServiceSpecificErrorType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                A type definition containing service specific error information.

                Elements:

                serviceType - The service type identifying the applicable
                service description in the context of the NSA generating the
                error.

                serviceErrorId - The service specific error Id.

                serviceErrorText - User friendly message text describing the
                error.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="serviceType"      type="xsd:string" />
            <xsd:element name="serviceErrorId"   type="xsd:string" />
            <xsd:element name="serviceErrorText" type="xsd:string" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="VariablesType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                A type definition providing a set of zero or more type/value
                variables used for modeling generic attributes.

                Elements:

                variable - The variable containing the type/values.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="variable" type="tns:TypeValuePairType" minOccurs="0"
maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="TypeValuePairListType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                A holder element providing an attribute list definition for the
                type/value pair.

                Elements:

                attribute - An instance of a type/value pair.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="attribute" type="tns:TypeValuePairType" minOccurs="0"
maxOccurs="unbounded" />
```

```
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="TypeValuePairType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                Definition for a simple type and multi-value tuple.  Includes
                simple string type and value, as well as more advanced
                extensions if needed.  A targetNamespace attribute is included
                to provide context where needed.

                Elements:
                value - A string value corresponding to type.

                any - Provides a flexible mechanism allowing additional elements
                to be provided as an alternative, or in combination with value.
                Use of this element field is beyond the current scope of this
                NSI specification, but may be used in the future to extend the
                existing protocol without requiring a schema change.

                Attributes:
                type - A string representing the name of the type.

                targetNamespace - An optional URL to qualify the name space of
                the capability.

                anyAttribute - Provides a flexible mechanism allowing additional
                attributes non-specified to be provided as needed for peer-to-peer
                NSA communications.  Use of this attribute field is beyond the
                current scope of this NSI specification, but may be used in the
                future to extend the existing protocol without requiring a schema
                change.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="value" type="xsd:string" nillable="true" minOccurs="0"
maxOccurs="unbounded" />
            <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
        </xsd:sequence>
        <xsd:attribute name="type" type="xsd:string" use="optional" />
        <xsd:attribute name="targetNamespace" type="xsd:anyURI" use="optional" />
        <xsd:anyAttribute namespace="##other" processContents="lax" />
    </xsd:complexType>

   <!-- ************************************************************
        *                        XML base types                      *
        ************************************************************ -->

    <xsd:simpleType name="ConnectionIdType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                A connectionId is a simple string value that uniquely identifies
                a reservation segment within the context of a Provider NSA.  This
                value is not globally unique.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:string" />
    </xsd:simpleType>

    <xsd:simpleType name="DateTimeType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                The time zone support of W3C XML Schema is quite controversial
                and needs some additional constraints to avoid comparison
                problems. These patterns can be kept relatively simple since
                the syntax of the datetime is already checked by the schema
                validator and only simple additional checks need to be added.
                This type definition checks that the time part ends with a "Z"
                or contains a sign.
            </xsd:documentation>
```

```
        </xsd:annotation>
        <xsd:restriction base="xsd:dateTime">
            <xsd:pattern value=".+T.+(Z|[+-].+)"/>
        </xsd:restriction>
    </xsd:simpleType>

    <xsd:simpleType name="NsaIdType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                A specific type for a Network Services Agent (NSA) identifier
                just in case we need to change it in the future.  This type
                will be populated with a OGF URN (reference artifact 6478
                "Procedure for Registration of Subnamespace Identifiers in
                the URN:OGF Hierarchy") to be used for compatibility with
                other external systems.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:anyURI" />
    </xsd:simpleType>

    <xsd:simpleType name="UuidType">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                Universally Unique Identifier (UUID) URN as per ITU-T Rec.
                X.667 | ISO/IEC 9834-8:2005 and IETF RFC 4122.
            </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:anyURI">
            <xsd:pattern value="urn:uuid:[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-
[a-f0-9]{12}"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:schema>
```

13 References

[BRADNER] Scott Bradner. Key Words for Use in RFCs to Indicate Requirement Levels, RFC 2119. The Internet Society. March 1997. http://tools.ietf.org/html/rfc2026

[BUSH] Randy Bush, Thomas Narten. Clarifying when Standards Track Documents may Refer Normatively to Documents at a Lower Level. RFC 3967. The Internet Society. December 2004. http://tools.ietf.org/html/rfc3967

[CATLETT] Charlie Catlett, Cees de Laat, David Martin, Gregory B. Newby, Dane Skow. GFD-C.152: Open Grid Forum Document Process and Requirements. Open Grid Forum. June 2009. http://www.ogf.org/documents/GFD.152.pdf

[RESCORLA] Eric Rescorla, Brian Korver, Internet Architectures Board, Guidelines for Writing RFC Text on Security Considerations. RFC 3552. The Internet Society. July 2003. http://tools.ietf.org/html/rfc3552