OIF E-NNI Signaling Specification

IA # OIF-E-NNI-Sig-02.0

April 16, 2009

Implementation Agreement created and approved
by the Optical Internetworking Forum
www.oiforum.com

The OIF is an international non profit organization with over 90 member companies, including the world's leading carriers and vendors. Being an industry group uniting representatives of the data and optical worlds, OIF's purpose is to accelerate the deployment of interoperable, cost-effective and robust optical internetworks and their associated technologies. Optical internetworks are data networks composed of routers and data switches interconnected by optical networking elements.

With the goal of promoting worldwide compatibility of optical internetworking products, the OIF actively supports and extends the work of national and international standards bodies. Working relationships or formal liaisons have been established with Ethernet Alliance, IEEE 802.1, IEEE 802.3ba, IETF, IPv6 Forum, ITU-T SG13, ITU-T SG15, MEF, ATIS-OPTXS,ATIS-TMOC, TMF and the XFP MSA Group.

For additional information contact:
The Optical Internetworking Forum, 48377 Fremont Blvd.,
Suite 117, Fremont, CA 94538
510-492-4040 ✦ info@oiforum.com

www.oiforum.com

Working Group: **Architecture and Signaling**

TITLE: **OIF E-NNI Signaling Specification**

SOURCE:

TECHNICAL EDITOR

George Newsome
Ciena Corporation
1201 Winterson Road
Linthicum, MD 21090
Phone: 732 308 1518
Email: gnewsome@ieee.org

WORKING GROUP CHAIR

Jonathan Sadler
Tellabs
1415 West Diehl Road
Naperville, IL 60563
Phone: +1.630.798.6182
Email: jonathan.sadler@tellabs.com

List of Contributors

The E-NNI 2.0 was based on the E-NNI 1.0 [OIF-E-NNI-Sig-01.0]

We acknowledge the work of the contributors to the E-NNI 1.0:

Osama Aboul-Magd	Richard Graveman	Satoru Okamoto	Vishnu Shukla
Nazik Andonian	Jim Jones	Lyndon Ong	John Strand
Olga Aparicio	Nic Larkin	Tomohiro Otani	Eve Varma
Greg Bernstein	Monica Lazer	Dimitrios Pendarakis	Amy Wang
Kalyani Bogineni	Dmitry Levandovsky	Rajender Razdan	Dongmei Wang
Carmine Daloia	Guangzhi Li	Siva	Jennifer Yates
Sudheer Dharanikota	Zhi-Wei Lin	Sankaranarayanan	Erning Ye
Hans-Martin Foisel	Yoshiharu Maeno	Stephen Shew	Lucy Yong

We acknowledge the work of the contributors to the E-NNI 2.0 extensions:

Alessandro D'Alessandro	Lyndon Ong
Fred Gruman	Vijay Pandian
Jim Jones	Evelyne Roch
Hans-Martin Foisel	Walter Rothkegel
Richard Graveman	Jonathan Sadler
Tom Hambleton	Stephen Shew
Monica Lazer	Remi Theillaud
Thierry Marcot	Vishnu Shukla
Scott McNown	Mike Soulakis
David McWalter	Eve Varma
George Newsome (Editor)	

1 Table of Contents

2 List of Figures

3 List of Tables

4 Introduction

This document specifies the content and operation of the [OIF-E-NNI-02.0] signaling protocol. The E-NNI signaling protocol communicates between control domains, discussed below, to create connections. E-NNI signaling functions, along with the [OIF-UNI-02.0] and I-NNI signaling protocols (the latter not specified by OIF), are used to establish end-to-end transport services.

The deployment of Automatically Switched Optical Networks (ASONs) into new and existing networks occurs within the context of commercial operator business practices and heterogeneous transport networks (e.g., with respect to transport technologies, vendors, and approach to management and control). This is true even within a single carrier's network. These business and operational considerations require an optical control plane architecture and supporting protocols to protect such commercial business operating practices that, for example, generally segment transport networks into domains according to managerial or policy considerations. Per [G.8080], the term domain is used to express differing administrative or managerial responsibilities, trust relationships, addressing schemes, infrastructure capabilities, survivability techniques, distributions of control functionality, etc. The control plane supports establishing on-demand services through the automatic provisioning of end-to-end connections across one or more domains. This involves both call and connection aspects:

- The call (service) aspect involves the provisioning of end-to-end services, while respecting commercial business operating practices, as shown in Figure 1.

- The connection aspect involves the automatic provisioning of resources in support of end-to-end services that may span one or more domains.

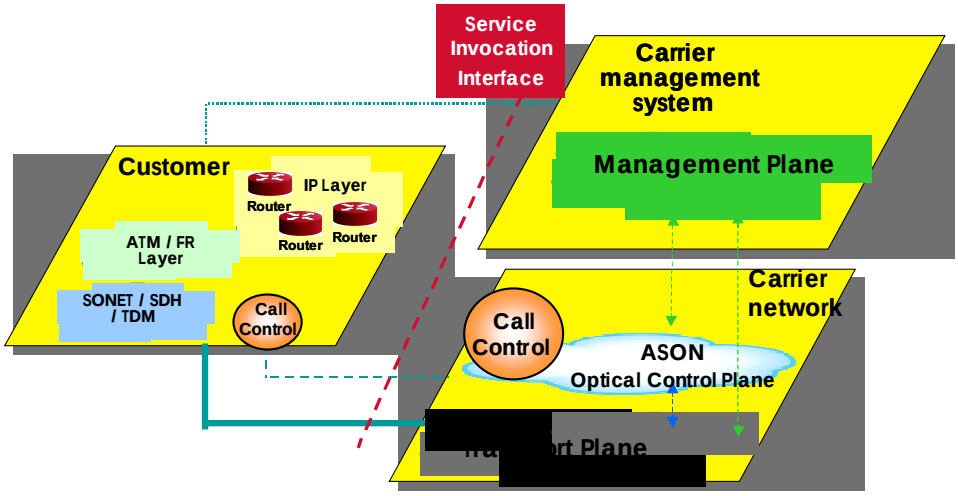


Figure 1: Call (Service) Aspect of Optical Control Plane

As mentioned above, domains are established by operators' policies and have a range of membership criteria; i.e., a domain represents a collection of entities grouped for a particular purpose. A control domain is a type of transport domain, where the criterion for membership is the scope of a control plane component that is responsible for the transport resources within the transport domain. The link and subnetwork connections between and within domains are described in terms of reference points. Because domains are established via operator policies, inter-domain reference points are service demarcation points (i.e., points where call control is provided).

- The reference point between a user and a provider domain is the UNI, which represents a user-provider service demarcation point.

- The reference point between domains is the E-NNI, which represents a service demarcation point supporting multi-domain connection establishment. The nature of the information exchanged between control domains across the E-NNI reference point captures the common semantics of the information exchanged amongst its constituent components, while allowing for different representations inside each control domain.

- The reference point within a domain is an I-NNI, which represents a connection point supporting subnetwork connection establishment.

Figure 2 illustrates a simple example of control plane domain configuration for a multi-carrier (Carriers A and B) and multi-vendor (Vendors X and Y within Carrier B) transport network. In this example, Vendor X and Vendor Y have used different control plane signaling protocols within their domains. By policy, the control plane has been subdivided into multiple control domains, interconnected by E-NNI signaling interfaces, which have been delimited according to Carrier A and Carrier B's administrative boundaries, and, in this example, further subdivided

within Carrier B according to vendor domain. This subdivision enables business boundaries and signaling protocol heterogeneity to be handled.

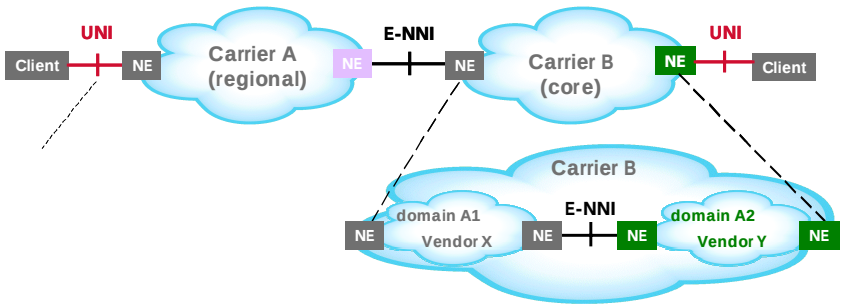


Figure 2: Example of Control Plane Subdivided into Multiple Control Domains

It should be noted that from an E-NNI signaling perspective, there is no distinction between the inter- and intra-carrier scenarios. However, inter- versus intra-carrier considerations have implications for such matters as policy, routing, addressing, and security. Treatment of these subjects is beyond the scope of this Implementation Agreement.

4.1 Problem Statement

The advent of the automatic switched optical network has necessitated the development of interoperable procedures for requesting and establishing dynamic connection services across heterogeneous networks. Developing such procedures requires defining:

- Control domains and associated reference points (E-NNI, I-NNI, UNI)
- Services offered by the optical transport network across control domains
- Signaling protocols used to invoke the services across E-NNI interfaces
- Mechanisms used to transport signaling messages

The first phases of specifying the user-provider call control signaling interface and External Network Node Interface (E-NNI) signaling protocols has been completed in [OIF-UNI1-R2] and [OIF-ENNI-SIG-01]. This revised E-NNI Signaling specification includes E-NNI 1.0 principal ballot comments, resolutions and updates from the 2004 World Interop Demo and the 2005 Supercomm Interop Demo, and revisions to support UNI 2.0 features [OIF-UNI-02.0].

4.2 Scope

The scope of this Implementation Agreement is the specification of E-NNI signaling abstract messages, attributes, and flows to establish end-to-end dynamic connections across multiple control domains. The Implementation Agreement also provides a concrete realization of the abstract messages and attributes based on the RSVP-TE protocols.

The E-NNI Signaling 2.0 specification applies to SDH / SONET, [G.709] OTN (ODUk), and Ethernet connection services in support of UNI 2.0 features. This Implementation Agreement also describes the support of Soft Permanent Connection (SPC) and Switched Connection (SC) services over the E-NNI, associated architecture and reference configurations, and identifiers relevant to signaling (including signaling communications).

The ASON architecture supports both single- and multi-layer scenarios. Multi-layer scenarios include network elements that support more than a single layer, a layer network that supports virtual concatenation and its server layers, transport services that exist where the client layer has no resources in the subnetwork except at its edges, etc. For example, at the edge of a multi-layer transport network, a network element may support client layer networks that are not directly supported in the core of a multilayer transport network. The client CI (Characteristic Information, [G.805]) could be adapted, possibly multiple times, onto server layer connections. Multilayer support refers to the ability to handle the adaptation of one CI into another CI. This Implementation Agreement is limited to operations on a single layer network (see [G.800]) at a time; however, this is not intended to restrict the interface from signaling about different layers at different times. Neither does it restrict the addition of further signaling capabilities in future versions of the E-NNI.

There is no support for Ethernet Switching at the E-NNI 2.0 interfaces. Ethernet services carried in adaptations to a different server layer are supported, but the interactions between the layers are beyond the scope of this document. UNI 2.0 Ethernet services are supported over provisioned server layer trails as shown in Figure 3 or over switched connections or soft permanent server layer connections as shown in Figure 4. In E-NNI 2.0, Ethernet signaling is limited to providing a forwarding function for messages and objects in client layers (Ethernet and VCAT) to support UNI 2.0. Future versions of E-NNI or other IAs will further define signaling messages, objects, and behaviors required to support Ethernet services over the E-NNI.

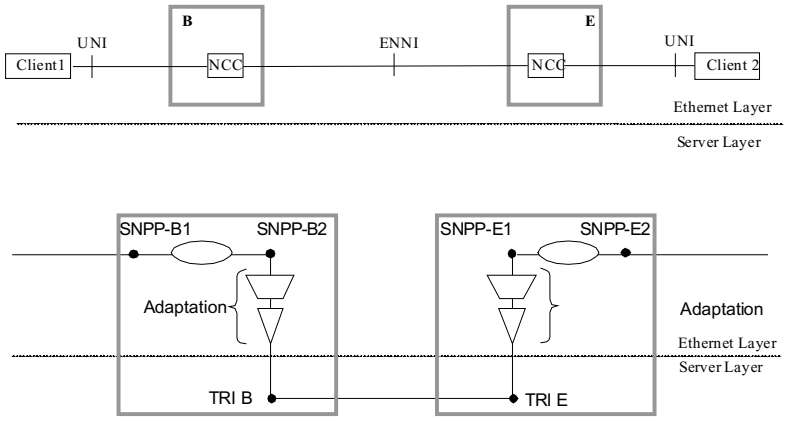


Figure 3: Ethernet Traffic Adapted onto a Provisioned Server Layer Trail

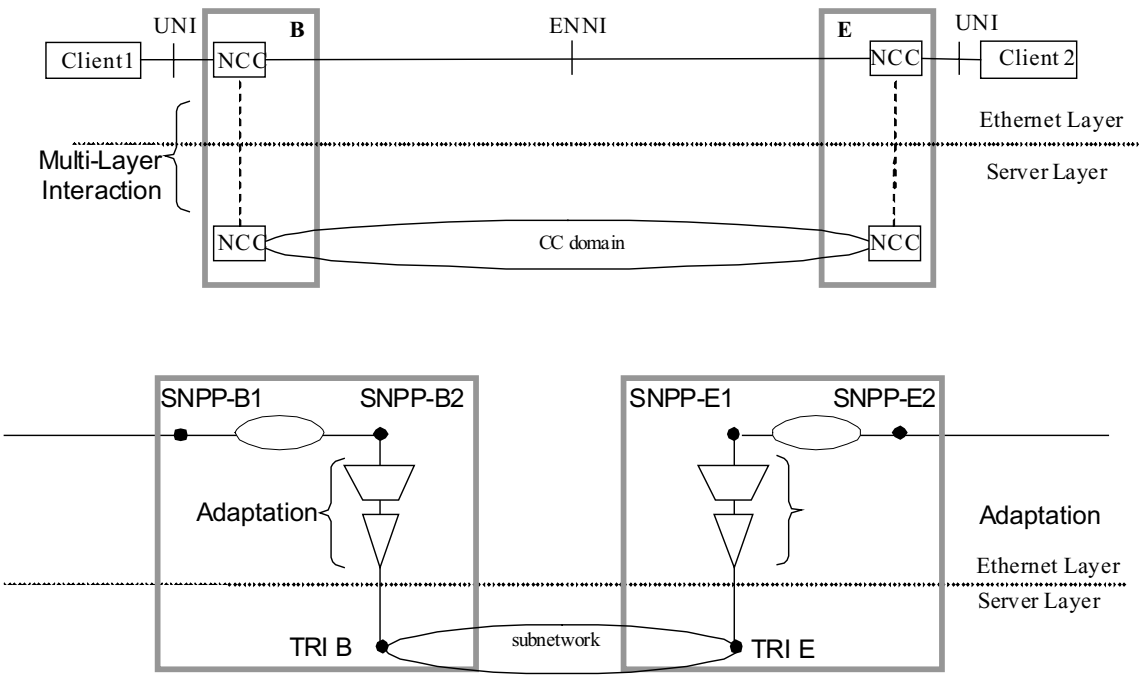


Figure 4: Ethernet Traffic Adapted onto a Switched or Soft Permanent Server Layer Connection

Routing, reachability, and address resolution protocols are outside the scope of this Implementation Agreement. Specifications related to generic areas such as Signaling Communications Network (SCN) design, auto-discovery, and policy are also outside the scope of this Implementation Agreement.

4.3 Relationship to Other Standards Bodies

This Implementation Agreement, to the greatest extent possible, uses available global standards documents. Specifically, the SDH / SONET and OTN structure and format definitions are based on [G.707] and [G.709], respectively. Control Plane Architecture is based on [G.8080]. Signaling is based upon the requirements defined in ITU-T Recommendation [G.7713], the protocols defined in [G.7713.2], and foundation IETF protocols. In case of differences related to ASON feature support, the ITU-T text is assumed to take precedence. For features already supported in existing standards, no new protocol messages or information elements are defined beyond those already specified; however, in some cases specific choices are given as to options exercised or detailed usage within the scope of the OIF E-NNI. In particular, Section 13 defines those objects expected in specific messages within RSVP-TE, including the detailed methods applied as needed. The security and logging in this IA are based on the OIF profiles of the work in the IETF's IPSEC and SYSLOG Working Groups.

As the intent of the OIF is to develop E-NNI protocols in close alignment with ITU-T Recommendations and foundation IETF RFCs, any changes made in those respective standards will be considered in future updates of this Implementation Agreement. In an analogous manner, should new signaling capabilities be considered in future OIF work, it is expected that the OIF would liaise this information to ITU-T and IETF.

4.4 Merits to OIF

The E-NNI Signaling 2.0 specification is a key stride towards the implementation of an open inter-domain signaling protocol that enables dynamic setup and release of various services. This activity supports the overall mission of the OIF.

4.5 Working Groups

Architecture & Signaling Working Group

Carrier Working Group

Interoperability Working Group

OAM&P Working Group

4.6 Document Organization

This document is organized as follows:

- Section 4: Introduction
- Section 5: Terminology and Abbreviations
- Section 6: Services Supported Over the E-NNI
- Section 7: Identifiers for Signaling
- Section 8: Signaling Communications Network
- Section 9: E-NNI Signaling Reference Configurations
- Section 10: E-NNI Signaling Abstract Messages and Attributes
- Section 11: E-NNI Security and Logging
- Section 12: E-NNI Signal Flow
- Section 13: RSVP-TE Extensions for E-NNI Signaling
- Section 14: Compatibility with UNI and E-NNI
- Section 15: References
- Section 16: Appendix A: Detailed Example of Message Object Contents
- Section 17: Appendix B: List of companies belonging to OIF when document is approved

4.7 Keywords

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].

5 Terminology and Abbreviations

AGC	Access Group Container (see [G.8080])
ASON	Automatically Switched Optical Network (see [G.8080])
CC	Connection Controller (see [G.8080] and [G.7713])
CCC	Calling/Called Party Call Controller (see [G.8080])
CI	Characteristic Information (see [G.805])
CP	Connection Point (see [G.805])
CTP	Connection Termination Point (see [G.805])
DCN	Data Communications Network (see [G.7712])
ECC	Embedded Control Channels (see [G.7712])
E-NNI	External NNI (see [G.8080])
eNNI-D	The logical control plane entity that terminates E-NNI signaling in the downstream direction with respect to control plane initiation
eNNI-U	The logical control plane entity that terminates E-NNI signaling in the upstream direction with respect to control plane initiation
ERO	Explicit Route Object
GMPLS	Generalized MPLS
I-NNI	Internal NNI (see [G.8080])
IP	Internet Protocol version 4 (IPv4) or Internet Protocol version 6 (IPv6)
IPsec	Internet Protocol Security (see [OIF-SEC] and [SecAdd])
LC	Link Connection (see [G.805])
LIH	Logical Interface Handle
MBB	Make Before Break
MCN	Management Communications Network (see [G.7712])
MPLS	Multiprotocol Label Switching
NCC	Network Call Controller

NE	Network Element
NNI	Network Node Interface
Node ID	Node Identifier (see [G.7715.1])[1]
OAM	Operations And Maintenance
OTN	Optical Transport Network (see [G.709])
PC	Protocol Controller (see [G.8080])
RA	Routing Area (see [G.8080])
RRO	Record Route Object
RSVP	Resource Reservation Protocol (see [RFC2205])
RSVP-TE	RSVP Traffic Engineering (see [RFC3209])
SC	Switched Connection service (see [G.8080])
SC PC ID	Signaling Controller Protocol Controller Identifier
SCN	Signaling Communications Network (see [G.7712])
SDH	Synchronous Digital Hierarchy (see [G.707])
SNC	Subnetwork Connection (see [G.805])
SNP	Subnetwork Point (see [G.8080])
SNPP	Subnetwork Point Pool (see [G.8080])
SNPP Link	Subnetwork Point Pool Link (see [G.8080])
SONET	Synchronous Optical Network (see [T1.105])
SPC	Soft Permanent Connection service (see [G.8080])
TCP	Termination Connection Point (see [G.805])
TNA	Transport Network Assigned (see [G.8080])
TNE	Transport Network Element (see [OIF-UNI-02.0])
TRI	Transport Resource Identifier (see [G.8080])

[1] The Node ID identifies a node in the transport topology graph. This definition differs from that given in the OIF UNI 2.0 specification.

TTP	Trail Termination Point (see [G.805])
UML	Unified Modeling Language
UNI	User Network Interface (see [OIF-UNI-020], [G.8080])
VCAT	Virtual Concatenation

6 Services Supported Over the E-NNI

ITU-T standard [G.8080] defines three basic connection service types according to the distribution of call and connection management functionality between the control and the management planes:

- Permanent connection service: End-user to end-user connection that is provisioned by a management system or manual methods, involving configuration of every network element along the path with the required information.
- Switched Connection **(SC**) service: End-user to end-user connection that is fully established via the optical control plane and involving automated routing, with or without a distributed route computation approach, and signaling.
- Soft Permanent Connection **(SPC**) service: End-user to end-user connection in which the user-to-network portions of the end-to-end connection are established by a network management system, and the network portion of the end-to-end connection is established using the optical control plane. In the network portion of the connection, requests for establishing the connection are initiated by the management plane and setup by the control plane. It should be noted that hybrid SPC/SC service scenarios are also possible in which only one of the user-to-network portions of the end-to-end service is established by a network management system.

When either SPC or SC services (including hybrid SPC/SC) are supported across multiple control domains, E-NNI signaling is required (see Figure 5).

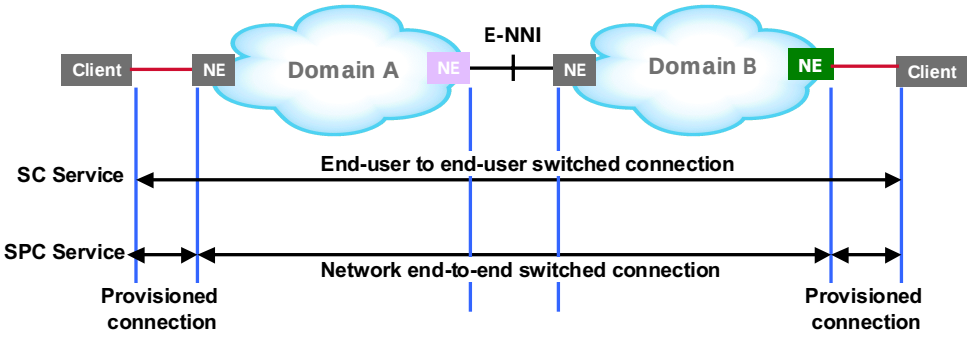


Figure 5: SC and SPC Connection Services Across Multiple Control Domains

The following sections provide ASON architectural context and describe how these UNI services impact E-NNI interface specifications. Additional information related to the definition of the UNI services may be found in [OIF-UNI-02.0], and [G.8080]. At the E-NNI, SC, SPC, and hybrid services are indistinguishable and receive the same treatment.

6.1 Support for Calls and Connections

Call and connection separation is a key aspect of the ASON architecture that distinguishes the service request itself (a.k.a. "call over a service interface" between the user and network or between networks) from the means by which the service request is realized within the network. As calls are end-to-end service associations, call state is not only maintained at the end-points, but is also required to be available at points where policy is applied (i.e., E-NNI). It should be noted that from a protocol perspective, two approaches are possible for supporting separation of calls and connections: (1) complete dissociation of the call operation (independent call and connection messages) from the connection operation, and (2) logical separation in which the existing connection protocol is used to enable call control. The second approach is followed within [G.7713.2], where call setup and teardown are signaled simultaneously along with connection setup and teardown. In other words, the call control information elements are "piggybacked" onto connection control messages.

Figure 6 illustrates an example of call and connection separation across multiple control domains. When a call spans multiple signaling control domains, it is composed of call segments that run between pairs of call-state aware points. In general, the scope of connection control is limited to a single call segment (e.g., due to independence of survivability schemes for each domain, a service realized differently in each domain, etc.), and one or more connections may be established in support of individual call segments. Note that the number of connections associated with individual call segments may not be the same even within a single end-to-end call. The example in Figure 6 illustrates UNI call segments with single associated LCs, whereas the subnetwork call segment for control domain 1 has two associated SNCs.

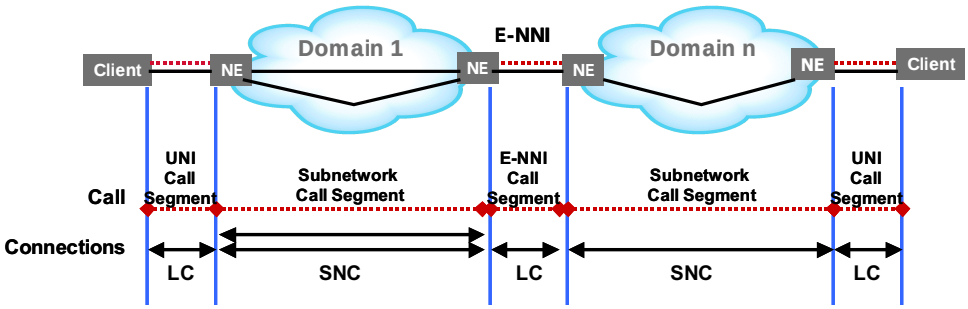


Figure 6: Call and Connection Separation Across Multiple Control Domains

The architecture also supports flexible choices of signaling across different domains. In this example, the UNI and E-NNI call segments use GMPLS RSVP-TE signaling, whereas the I-NNI's may use different signaling protocols.

6.2 Control Plane Component Representation

ITU-T Rec. G.8080 [G.8080] describes the ASON control plane architecture in terms of components and interfaces that represent logical functions (abstract entities) rather than physical implementations. To facilitate the construction of different scenarios, the ASON component

approach uses the Unified Modeling Language (UML). Key components referenced in the remainder of this document are enumerated below:

- Connection Controller (CC): Connection Controller components cooperate to set up connections.

- Calling / Called Party Call Controller (CCC) and Network Call Controller (NCC): Call Controller components cooperate to control the setup, release, and modification of calls. They are relevant to service demarcation points (i.e., CCC is relevant to the client facing side of the UNI; NCCs are relevant to the network facing side of the UNI and the E-NNI).

- Protocol Controller (PC): The Protocol Controller maps relevant control component (e.g., CC, CCC, NCC) parameters into messages, which are carried by an implemented protocol to support interconnection over a physical interface. (This includes support for implementations that handle, for example, multiple layers.)

Focusing upon the interactions of Call Controllers, as described in [G.7713] (Figure 6-1/G.7713/Y.1704) and illustrated in Figure 7, the Calling Party Call Controller (CCC-a) interacts with a Called Party Call Controller (CCC-z) by means of one or more intermediate Network Call Controllers (NCC). The NCC function is provided at the network edge (i.e., network facing UNI reference point) and may also be provided at gateways between control domains (i.e., separated by the E-NNI reference point). The functions performed by NCCs are defined according to policies associated with interactions across the UNI and E-NNI reference points.

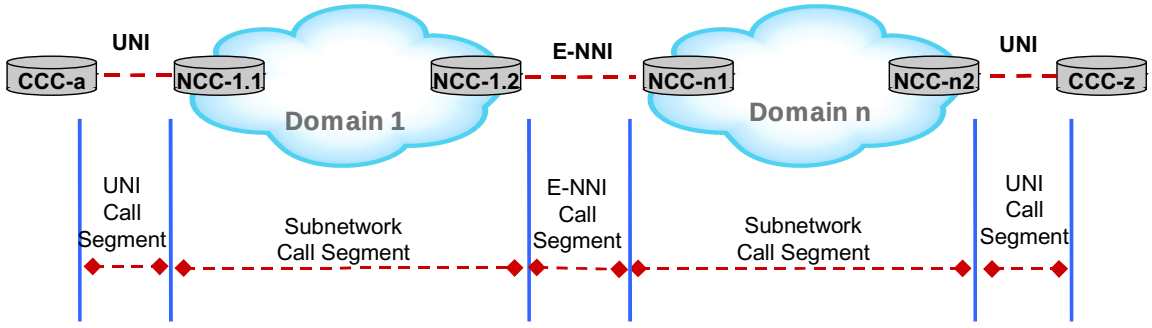


Figure 7: Interactions of Call Controllers Across Multiple Signaling Control Domains

The roles of the NCCs are as follows:

- correlate SNCs to the call within each control domain (NCC-1 ingress and egress pair for domain 1 and NCC-n ingress and egress pair for domain n)

- interact with CCC-a and CCC-z to correlate associated LCs to the call (ingress NCC-1 with CCC-a and egress NCC-n with CCC-z, respectively)

- interact with their peer NCCs at E-NNI control domain boundaries to correlate LCs to the call (egress NCC-1 to ingress NCC-n)

The Connection Controllers (CCs) establish the connections associated with each call segment.

6.3 Support for Switched Connection Services

A Switched Connection Service is characterized by a connection between two endpoints where the endpoints have signaling active on them. Note that this section refers to "endpoints" while previous sections referred to "clients." The difference is that the endpoint refers to an identifier, whereas client refers to a device originating or sinking the signal. A client has an endpoint identifier, though the endpoint of a connection need not terminate on a client.

When supporting an end-to-end Switched Connection Service between source and destination UNIs that traverses one or more E-NNIs, the call parameters originating from the source user must be preserved and passed across the E-NNI(s). For example, it must be possible to pass service-related attributes as call attributes (which impact, for example, diversity requirements) within the E-NNI signaling message. As discussed earlier, how the service is realized may be different within each domain. Additionally, it must be possible to pass explicit route information associated with E-NNIs within the signaling message across the E-NNI.

6.4 Support for Soft Permanent Connection Services

A Soft Permanent Connection Service is characterized by a connection between two endpoints where the endpoints do not have signaling active on them.

When supporting a Soft Permanent Connection service across one or more E-NNIs, the call parameters originating from the source network must be preserved and passed across the E-NNI. For example, it must be possible to pass service-related attributes as call attributes (which impacts, for example, diversity requirements) within the E-NNI signaling message. Here too, how the service is realized may be different within each domain.[2] Additionally, it must be possible to pass explicit route information associated with E-NNI gateways within the signaling message across the E-NNI.

6.5 Support for Hybrid Switched Connection / Soft Permanent Connection Services

A connection between two endpoints where one endpoint has signaling active and the other does not is called a Hybrid Switched Connection / Soft Permanent Connection. The originating signaling controller cannot identify whether the far endpoint of a connection is configured for Soft Permanent Connection operation.

Only the terminating network signaling controller can identify the egress interface type, based on the configuration of the egress interface.

This scenario is addressed in Section 6.1.2 of [OIF-UNI-02.0].

When supporting a Hybrid SC / SPC service across one or more E-NNIs, the call parameters originating from the source network must be preserved and passed across the E-NNI. For example, it must be possible to pass service-related attributes as call attributes (which impacts, for example, diversity requirements) within the E-NNI signaling message. Here too, how the service is realized may be different within each domain.[2] Additionally, it must be possible to pass explicit route information associated with E-NNI gateways within the signaling message across the E-NNI.

[2] Additional details on signaling requirements related to diversity and restoration (related to network Class of Service) may be found in [G.7713].

7 Identifiers for Signaling

An identifier provides a set of characteristics for an entity that makes it uniquely recognizable. There are two types of identifiers:

- A *name* identifies an entity uniquely within the context, or namespace, in which it is defined. It should be noted that the same entity might have different names in different namespaces.
- An *address* identifies a position in a specific topology. Addresses are unique for the topology and are typically hierarchically composed to allow for summarization of addresses for locations that are close together.

The major distinction between a name and an address is that an address is defined in terms of location in a topology. An entity can be located at a particular point in a topology for a period of time, and then may move to a different point in the topology. While the entity's address will change as a result of moving, the entity's name does not change. Names and addresses have an associated scope; the larger the scope, the more elements are needed in the identifier.

Three categories of identifiers are used for signaling entities in later sections: identifiers for transport resources used by the control plane, identifiers for signaling PCs, and identifiers for locating signaling PCs in the SCN. No assumption should be made regarding dependencies among these identifiers because they come from separate identifier spaces.

Table 1 through Table 3 provides a summary of signaling identifier categories used in the remainder of this document, and mapping among ITU-T, [OIF-UNI-02.0], and E-NNI 2.0 terms with a focus on the E-NNI context. In Section 13.3, the relevant identifiers are mapped to RSVP-TE objects.

It is important to note that signaling identifiers must be understood in the broader context of other identifier spaces relevant to transport resources. This is illustrated in Figure 8 (based upon Figure 9 of [G.7718 / Y.1709]).

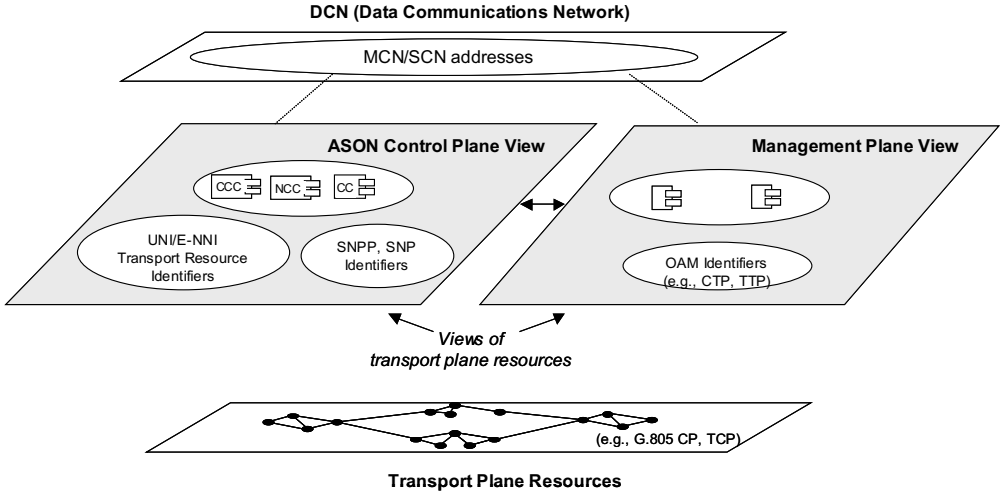


Figure 8: Identifier Space Relationships

7.1 Transport Resource Identifiers

Transport Resource Identifiers (TRI) are used by ASON control components to refer to [G.805] transport plane resources: Transport Network Assigned (TNA) and Subnetwork Point Pool (SNPP) identifiers.

Transport Network Assigned (TNA), ([G.8080] UNI Transport Resource) identifiers are used to identify transport resources at a UNI reference point. At an E-NNI reference point, Transport Resource Identifiers represent the resources between network domains (as illustrated in Figure 9).

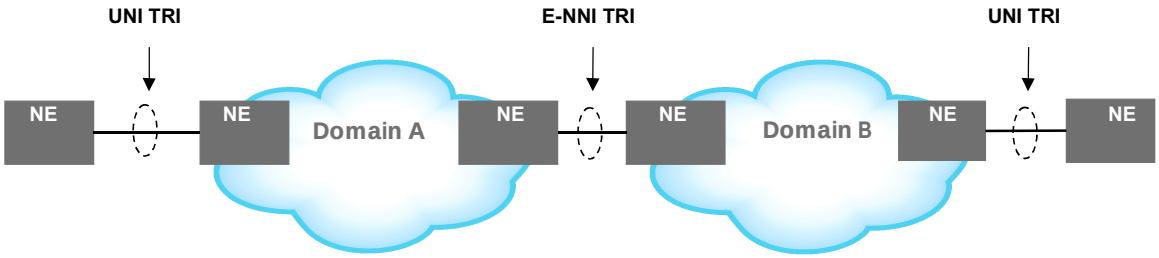


Figure 9: ASON Transport Resource Identifiers

The CCCs and NCCs use the UNI and E-NNI TRIs, respectively, to provide source and destination information for call control messages.

Referring to Figure 8, the SNP is an ASON control plane abstraction that represents an actual or potential underlying [G.805] transport resource CP (or CTP) or an actual or potential TCP (or TTP). The term SNPP refers to a set of subnetwork points (SNPs) grouped together for routing. An SNPP Link refers to an association between SNPPs on different subnetworks or Routing Areas, and SNPP Link identifiers allow these links to be distinguished. The two ends of the SNPP Link are assumed correlated with each other (either automatically via discovery or manually via provisioning). An SNPP ID identifies the end of an SNPP Link. This is illustrated in Figure 10 below.

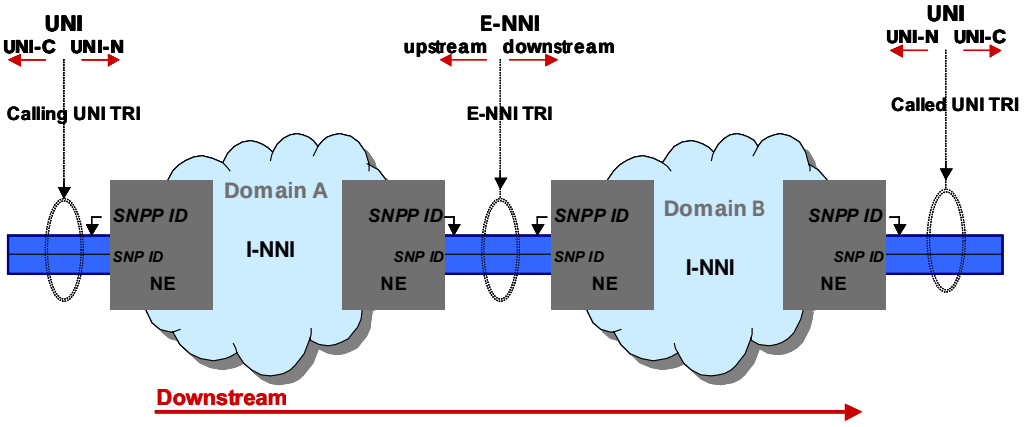


Figure 10: Example of SNP and SNPP IDs

UNI/E-NNI Transport Resource Identifiers (TRIs) may have a 1:N or N:1 relationship with SNPP Link identifiers. The N:1 relationship allows for aliases (multiple names to access the same

resource) and allows for a group of links that give access to the same services to be represented using a single TRI. The example in Figure 11, below, illustrates a scenario in which there are single and multiple SNPP Links per E-NNI Transport Resource.

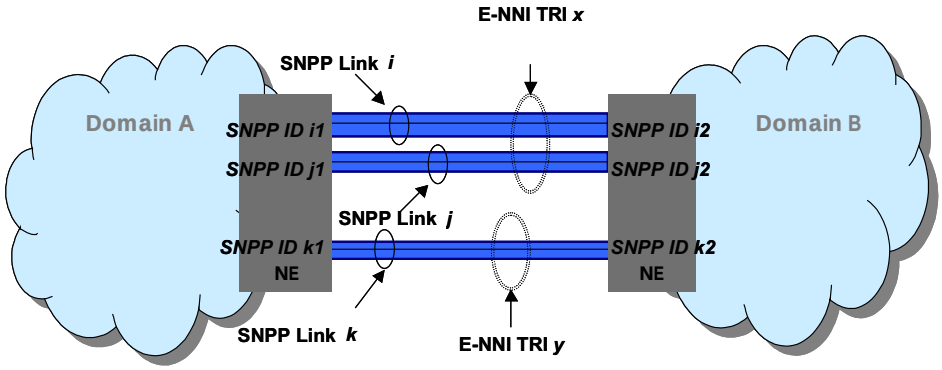


Figure 11: SNPP ID and E-NNI TRI Relationships

Table 1, below, provides a synopsis of TRI relationships and mapping among ITU-T, [OIF-UNI-02.0], and E-NNI 2.0 terms with a focus on the E-NNI context.

Table 1: Transport Resource Identifier Relationships (See Figure 12)

ITU-T		OIF		Comments
G.8080 / G.7713	Scope	OIF UNI 2.0	E-NNI Signaling 2.0	
SNPP ID	Local	Node ID / Logical Port ID	Node ID / Logical Port ID	Table 7-3 / G.7713 [Note: multiple SNPP IDs may be supported on the downstream node for a particular link]
[E-NNI upstream context] SNPP ID	Local	N / A	Node ID / Initiating Logical Port ID	Not in G.7713; included for completeness
[E-NNI downstream context] SNPP ID	Local	N / A	Node ID / Terminating Logical Port ID	Not in G.7713, Table 7-3; included for completeness
SNP ID	Local	Generalized Label	Generalized Label	Generalized label MUST be taken from a logical port ID
[E-NNI upstream context] SNP ID	Local	N / A	Initiating Generalized Label	
[E-NNI downstream context] SNP ID	Local	N / A	Terminating Generalized Label	
[UNI-N] DEST SNP ID	Only in scope for Remote end	SPC Destination Generalized Label	N / A	Scoped by either SNPP (which might not be in the scope of a UNI TRI) or UNI TRI (inherently scoped re SNPP)
UNI Transport Resource ID (TRI)	End-to-end	TNA name	N / A	E-NNI behavior: end-to-end or carry transparently
Calling UNI TRI		Source TNA name		Table 7-1 / G.7713
Called UNI TRI		Destination TNA name		Table 7-1 / G.7713
E-NNI Transport Resource ID	End-to-end	N / A	TNA name	In the case where carriers on either side of the E-NNI link may not wish to expose their internal routable address to each other, an alias for the E-NNI TRI may be used.

Representation of TRIs, SNPP and SNP IDs in [OIF-UNI-02.0] and E-NNI 2.0 terminology is illustrated in Figure 12 below.

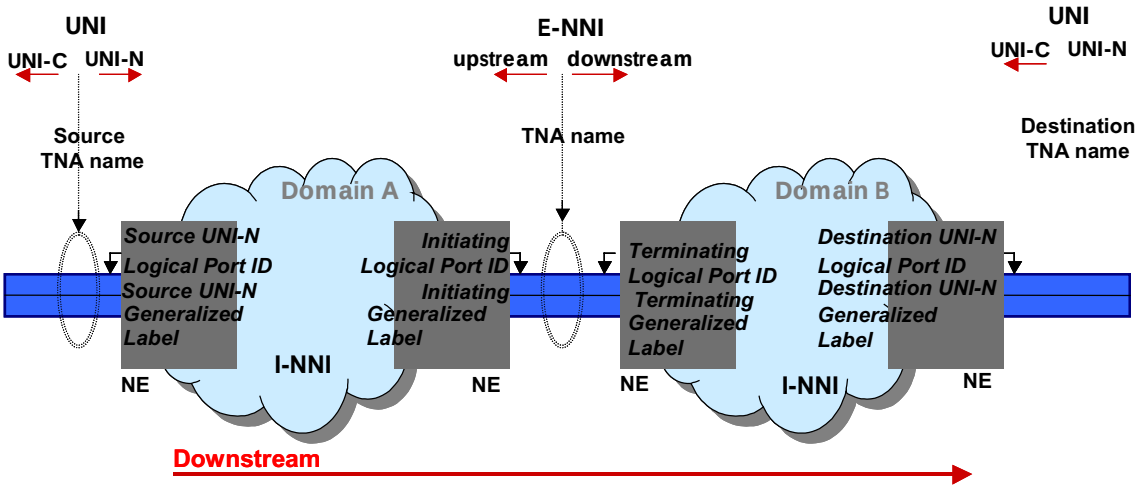


Figure 12: Example - Use of UNI 2.0 and E-NNI 2.0 Terminology

7.2 Signaling Protocol Controller Identifiers

As discussed earlier, PCs are used for protocol specific communications. The NCC and CCC PCs must have identifiers that allow them to maintain call signaling relationships with peers. In the local context of an E-NNI, the upstream node NCC PC identifier is referred to as an *Initiating NCC PC ID*; the identifier in the downstream direction is referred to as a *Terminating NCC PC ID* (illustrated in Figure 13). These identifiers must be unique within the scope specified in Table 2 to specify a particular E-NNI signaling channel unambiguously.

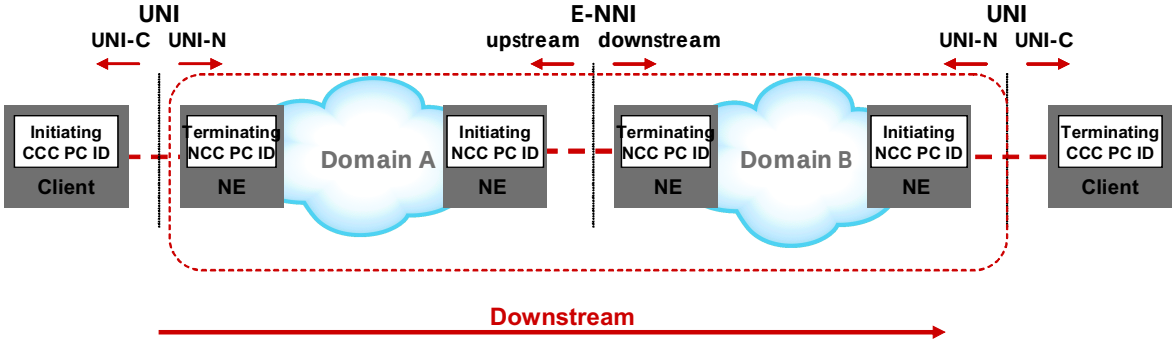


Figure 13: Upstream and Downstream NCC PC IDs

Table 2, below, provides a synopsis of Signaling PC identifier relationships with a focus on the UNI-N and E-NNI. Note that upstream or downstream context is from the perspective of a node providing the signaling, and the table concentrates on the downstream relationship.

Table 2: Signaling Protocol Controller Identifier Relationships (Downstream Context)

ITU-T		OIF	
G.8080 / G.7713	Scope	OIF UNI 2.0	E-NNI 2.0
NCC PC ID	Global	UNI-N SC PC ID	ENNI SC PC ID
Initiating NCC PC ID	Global	Destination UNI-N SC PC ID at Destination UNI	eNNI-U SC PC ID
Terminating NCC PC ID	Global	Source UNI-N SC PC ID at Source UNI	eNNI-D SC PC ID
CC PC ID	Global	UNI SC PC ID	eNNI SC PC ID
Initiating CC PC ID	Global	Destination UNI-N SC PC ID at Destination UNI	Shared with eNNI-U SC PC ID
Terminating CC PC ID	Global	Source UNI-N SC PC ID at Source UNI-N	Shared with eNNI-D SC PC ID

Note that the global scope of the SC PC ID applies to use of the SC PC ID to provide a Call ID within a carrier domain. For calls across carriers, SC PC IDs used as Call IDs must contain the Carrier ID component.
Also note that the NCC PC ID and CC PC ID share the same identifier in E-NNI Signaling 2.0. Distinct identifiers are required to disassociate call and connection control fully. This is for further study.
A representation in terms of [OIF-UNI-02.0] and E-NNI 2.0 terminology is illustrated in Figure 14.

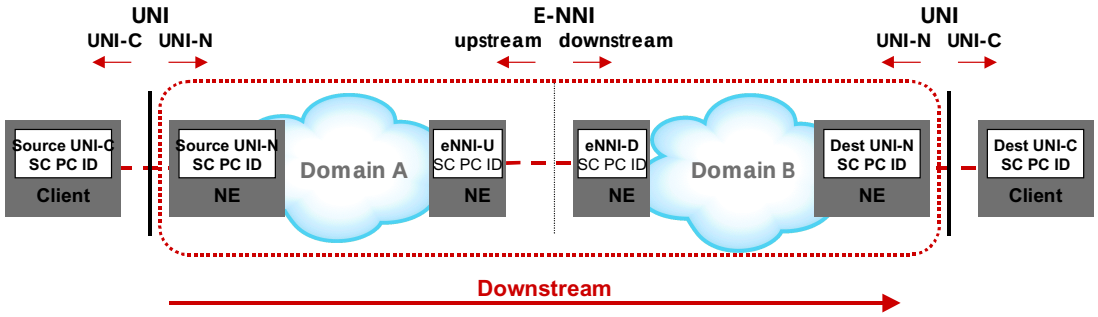


Figure 14: Representation of Signaling PC IDs in OIF UNI 2.0 and E-NNI 2.0 Terminology

7.3 Signaling Protocol Controller SCN Addresses

The Signaling Communications Network (SCN) topology is defined by the set of links and routers used to build a data network for carrying control-plane (i.e., signaling and routing) messages and is further discussed in Section 8. It has addresses for the points where systems participating in the control plane attach to the SCN, as well as for the routers that perform switching in the SCN. The points of attachment are identified by SCN addresses, which the signaling PCs use to communicate with each other via the SCN, as illustrated in Figure 15. Consequently, a signaling PC SCN address is based on the topology of the SCN carrying signaling messages and not the topology of the transport plane. All PCs require an SCN address for exchanging call/connection signaling messages.

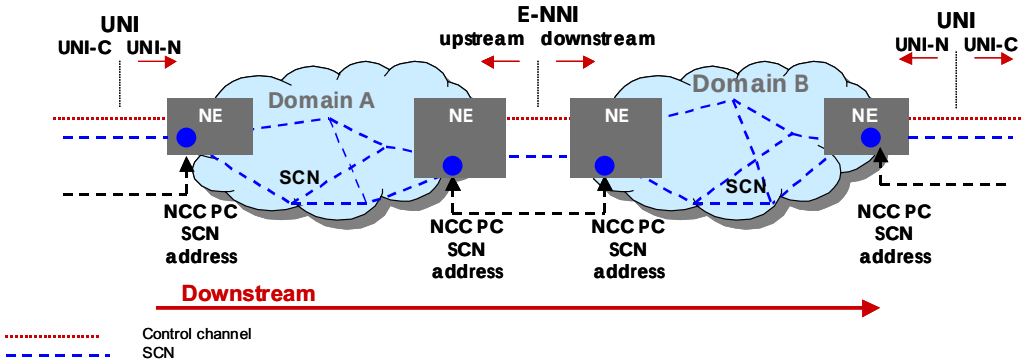


Figure 15: NCC PC SCN Addresses

Note that while the Protocol Controller ID is distinct from the Protocol Controller SCN address, these may be assigned the same value for administrative simplicity.

Table 3, below, provides a synopsis of Signaling PC SCN address relationships among ITU-T, [OIF-UNI-02.0], and OIF E-NNI Signaling 2.0. The table below solely focuses upon the UNI-N and E-NNI contexts.

Table 3: Signaling Protocol Controller SCN Address Relationships

ITU-T G.8080 / G.7713	Scope	OIF UNI 2.0	OIF E-NNI 2.0	Comments
NCC PC SCN address	Local	UNI-N SC PC SCN address	eNNI SC PC SCN address	
Initiating NCC PC SCN address	Local	Destination UNI-N SC PC SCN address (at Destination UNI)	eNNI-U SC PC SCN address	
Terminating NCC PC SCN address	Local	Source UNI-N SC PC SCN address (at Source UNI)	eNNI-D SC PC SCN address	
CC PC SCN address	Local	Shared with UNI SC PC SCN address	Shared with NCC PC SCN address	A distinct address may be needed for fully disassociated call and connection control
Initiating CC PC SCN address	Local	Shared - see Note 1 below	Shared with Initiating NCC PC SCN address	A distinct address may be needed for fully disassociated call and connection control
Terminating CC PC SCN address	Local	Shared – see Note 2 below	Shared with Terminating NCC PC SCN address	A distinct address may be needed for fully disassociated call and connection control
Note 1: Source UNI: Shared with Source UNI-C SC PC SCN address. Destination UNI: Shared with Destination UNI-N SC PC SCN address Note 2: Source UNI: Shared with Source UNI-N SC PC SCN address. Destination UNI: Shared with Destination UNI-C SC PC SCN address				

8 Signaling Communications Network

One consequence of distributed call and connection management is the need for a resilient SCN architecture that enables interactions of the protocols running on communicating control plane nodes. The requirements, architecture, and protocol specifications for a Data Communications Network (DCN), including the optical control plane infrastructure for the IP-based SCN, are provided in [G.7712]. Optical control-plane-capable network elements must support this functionality to provide for transport of optical control plane signaling messages.

Transport network elements currently support data communication functionality for transport of management messages between NEs and management elements, termed the Management Communications Network (MCN). However, optical-control-plane capable NEs need data communication functionality for both management and signaling applications. In this case, applications may be considered as running over separate logical networks – the MCN and SCN. The DCN represents the physical communications network supporting the logical MCN and SCN. The logical MCN and SCN may be supported via physically separate DCN networks, or a single DCN may support both the logical MCN and SCN as two applications sharing the same communications network. It is important to note that the communications network supporting inter-PC signaling communications cannot be assumed to be congruent with the corresponding transport plane connectivity. Consequently, a number of SCN topologies are possible.
The resiliency of the DCN supporting the logical SCN is critical to the viability of switched connection services, as failures affecting the SCN would impact:

- New call/connection requests, because the signaling network may not be available to carry the messages related to the new call or connection request.
- The ability to tear down existing calls or connections, because the signaling network may not be available to carry the messages related to tearing down an existing call/connection.
- The ability to restore existing connections, because the signaling network may not be available to carry messages related to restoration. Note that this impact only applies when a failure exists on the signaling network as well as the physical transport network.

The different options for considering the SCN in the context of the E-NNI may be impacted by whether an intra- or inter-carrier E-NNI is being supported. Examples include:

- Keeping the different SCN RAs distinct and performing some manual provisioning of reachable SCN addresses between them.
- Having the SCNs in a common parent SCN RA, with Level 0 routers interconnecting them.

These examples are illustrated in Figure 16 below.

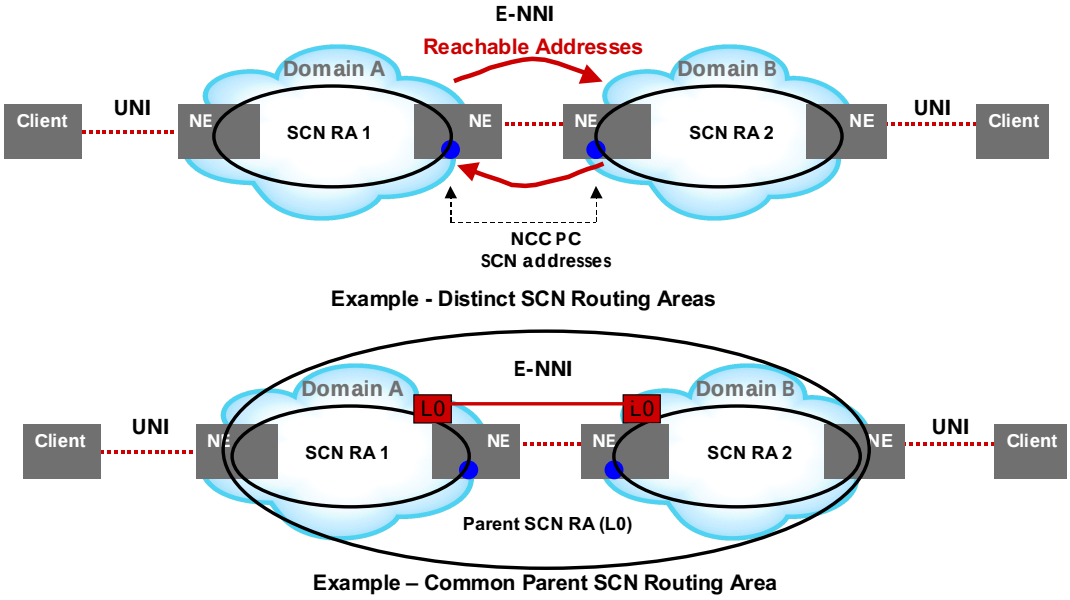


Figure 16: Example SCN Architectures for E-NNI

9 E-NNI Signaling Reference Configurations

As with UNI service invocation, E-NNI signaling may be supported via two invocation models, one called direct invocation and the other called indirect invocation. For the direct invocation model, the NCC PC is integrated within the NE. In the indirect invocation model, the NCC PC is not integrated into a particular NE and may support E-NNI signaling functions for one or more NEs. Examples of four invocation scenarios, with respect to Domain A, are illustrated in Figure 17.

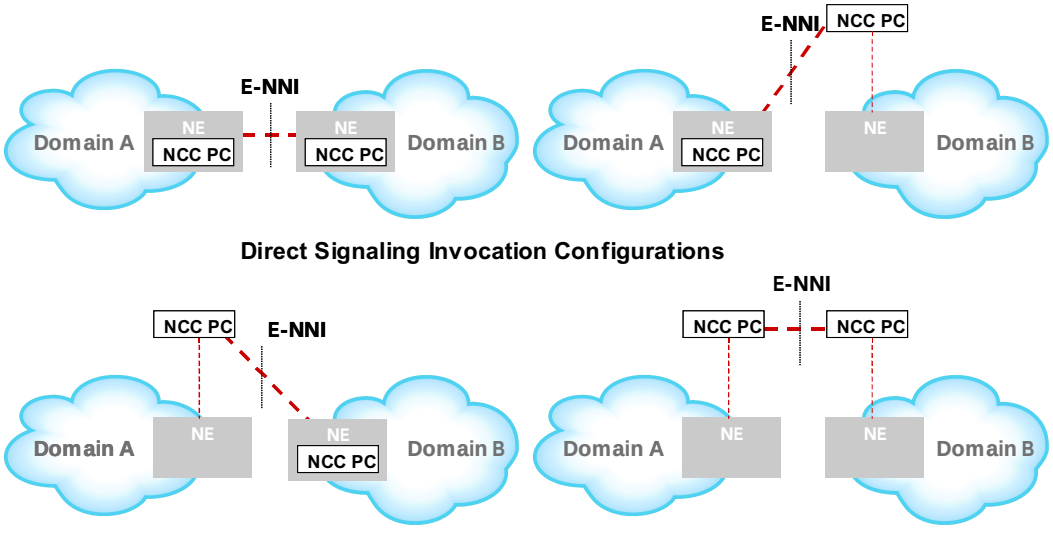


Figure 17: E-NNI Signaling Invocation Reference Scenarios

10 E-NNI Signaling Abstract Messages and Attributes

E-NNI signaling supports the exchange of messages for providing connection management operations across an inter-domain interface. This section discusses the abstract messages necessary to support the operations required for E-NNI signaling. These messages are described as "abstract" because the actual realization depends on the signaling protocol used. This section uses the reference configurations in Section 9 to define interfaces where E-NNI signaling is applicable. The E-NNI signaling messages support the ability to establish end-to-end connections dynamically across multiple control domains, which provides for both switched connection and soft permanent connection services.

10.1 Abstract Messages and Error Codes

In E-NNI 2.0, the E-NNI abstract call message attributes are implicitly included in the connection messages. The following abstract call messages (Table 4) are currently defined:

Table 4: E-NNI Abstract Call Messages

Abstract Message	Implicit Signaling
Call Setup Request	Implicitly signaled in the Connection Setup Request for the first connection of the call.
Call Setup Indication	Implicitly signaled in the Connection Setup Indication for the first connection of the call.
Call Setup Confirm	Implicitly signaled in the Connection Setup Confirm for the first connection of the call.
Call Release Request	Implicitly signaled in the Connection Release Request of the last connection of the call.
Call Release Indication	Implicitly signaled in the Connection Release Indication of the last connection of the call.
Call Query Request	Implicitly signaled in the Connection Query Request for each connection within the call.
Call Query Indication	The call information is piggybacked onto the Connection Query Indication for each connection within the call.
Call Modify Request	- If the call is modified by changing the bandwidth of an existing connection, the Call Modify Request is implied in the Connection Modify Request. - If the call is modified by adding a connection, the Call Modify Request is implied in the Connection Setup Request. - If the call is modified by removing a connection, the Call Modify Request is implied in the Connection Release Request.
Call Modify Indication	- If the call is modified by changing the bandwidth of an existing connection, the Call Modify Indication is implied in the Connection Modify Indication. - If the call is modified by adding a connection, the Call Modify Indication is implied in the Connection Setup Indication. - If the call is modified by removing a connection, the Call Modify Indication is implied in the Connection Release Indication.

Abstract Message	Implicit Signaling
Call Modify Confirm	- If the call is modified by changing the bandwidth of an existing connection, the Call Modify Confirm is implied in the Connection Modify Confirm. - If the call is modified by adding a connection, the Call Modify Request is implied in the Connection Setup Confirm. - If the call is modified by removing a connection, there is no Call Modify Confirm.

Table 5 provides the current list of abstract messages to support E-NNI signaling, based on the message sets defined in [G.7713]. The message sets are separated into five types. These include connection setup messages, connection release messages, connection query messages, connection notification messages, and connection modification messages.

Table 5: E-NNI Signaling Abstract Messages

Abstract Messages		Reference
Connection Setup Messages	ConnectionSetupRequest	10.1.1.1
	ConnectionSetupIndication	10.1.1.2
	ConnectionSetupConfirm (optional)	10.1.1.3
Connection Release Messages	ConnectionReleaseRequest	10.1.2.1
	ConnectionReleaseIndication	10.1.2.2
Connection Query Messages	ConnectionQueryRequest	10.1.3.1
	ConnectionQueryIndication	10.1.3.2
Connection Notification Messages	ConnectionNotification	10.1.4.1
Connection Modification Messages	ConnectionModifyRequest	10.1.5.1
	ConnectionModifyIndication	10.1.5.2
	ConnectionModifyConfirm	10.1.5.3

A single abstract message for Signaling Adjacency Maintenance is also defined.

10.1.1 Connection Setup Messages

10.1.1.1 Connection Setup Request

The connection setup request message is used by the eNNI-U to transmit a service request towards the eNNI-D. This message provides information needed to set up the E-NNI portion of the end-to-end connection as well as additional information that may be used by downstream CCs to complete the end-to-end request. This message is specified as having a global scope, i.e., the message indicating the service request is transmitted end-to-end. Table 6 shows the abstract message and its content.

Table 6: ConnectionSetupRequest Abstract Message

Message: ConnectionSetupRequest Scope: Global Direction: eNNI-U eNNI-D			
Attribute	**Type**	**Scope**	**Call/Conn**
Source TNA	Mandatory	Global	Call
Destination TNA	Mandatory	Global	Call
Initiating NCC PC ID	Mandatory	Local	Connection
Terminating NCC PC ID	Mandatory	Local	Connection
Connection Name	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call
SNP ID[1]	Optional	Local	Connection
SNPP ID	Mandatory	Local	Connection
Directionality	Optional	Global	Connection
DEST SNP ID[2]	Optional	Local[3]	Call
Explicit Route	Optional	Global[4]	
Service Level	Mandatory[5]	See Note 6	Call
Contract ID	Optional	Global	Call
Encoding Type	Mandatory	Global	Call/ Connection
Switching Type	Mandatory	Global	Call/ Connection
SONET/SDH, OTN or Ethernet traffic parameters	Mandatory	Global	Call/ Connection
Generalized Payload Identifier	Optional	Global	Call

NOTE 1: The SNP (Section 10.2.3.2) and SNPP ID (Section 10.2.3.3) are identifiers that refer to the resource to be allocated on the local link. They are generally an ordered list of IDs. For a bidirectional connection request, the SNP/SNPP ID list contains both downstream and upstream ID information. Also note that both SNP ID and DEST SNP ID are shown as optional. This is because it is possible to direct a connection to an SNPP and have the network choose the specific SNP to use.

NOTE 2: The DEST SNP ID refers to the SNP ID of the final destination of the connection (or call), and which may be a UNI-C or UNI-N SNP.

NOTE 3: This attribute is meaningful only at the destination of the SPC connection, i.e., the SPC identifier has meaning at the destination TNA. As such, all intermediate nodes forward this ID without processing.

NOTE 4: This attribute provides explicit route information across multiple control domains (link and "abstract" node level) delimited by E-NNIs. It may provide more detailed nodal routing information if policy allows.

NOTE 5: It is mandatory to have a service level attribute. If the attribute is not present, a default service level should be applied.

NOTE 6: This attribute is unique within the scope of a single carrier. It allows translation of "equivalent" service level information between control domains.

10.1.1.2 Connection Setup Indication

The connection setup indication message is used by the eNNI-D to respond to a service request towards the eNNI-U. This message provides information needed to set up the E-NNI portion of

the end-to-end connection and additional information that may be used by eNNI-U to complete the end-to-end request, e.g., which SNP is used to service the request. In the case of setup rejection, this message includes error codes regarding the rejection. This message is specified as having a global scope, i.e., the message is transmitted end-to-end. Table 7 shows the abstract message and its content.

Table 7: ConnectionSetupIndication Abstract Message

Message: ConnectionSetupIndication Scope: Global Direction: eNNI-D eNNI-U			
Attribute	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Mandatory	Local	Connection
Terminating NCC PC ID	Mandatory	Local	Connection
Connection Name	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call
SNP ID	Mandatory	Local	Connection
Connection Status	Mandatory	Local	Connection
Directionality	Optional	Global	Connection
Error Code	Optional	Global/Local	Connection
Bandwidth Modification Support	Mandatory	Global	Connection

The following error codes are defined for carriage by this message:

- Called Party Busy
- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid / unknown connection ID
- Invalid / unknown Call ID
- Invalid SNP
- Unavailable SNP
- Invalid SNPP
- Unavailable SNPP
- Unavailable directionality
- Invalid SPC SNP
- Invalid route
- Unavailable service level

10.1.1.3 Connection Setup Confirmation

The connection setup confirmation message is used by the eNNI-U to respond to a setup indication towards the eNNI-D. This message is mandatory if confirmation was requested in the connection setup indication. This message provides final confirmation of setup of the E-NNI portion of the end-to-end connection. In the case of setup rejection, this message includes error codes regarding the rejection. This message is specified as having a global scope, i.e., the message is transmitted end-to-end. Table 8 shows the abstract message and its content.

Table 8: ConnectionSetupConfirm Abstract Message

Message: ConnectionSetupConfirm Scope: Global Direction: eNNI-U eNNI-D			
Attribute	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Mandatory	Local	Connection
Terminating NCC PC ID	Mandatory	Local	Connection
Connection Name	Mandatory	Local	Connection
Connection Status	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call
Error Code	Optional	Local	Connection

The following error codes are defined for carriage by this message:

- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid / unknown connection ID
- Invalid / unknown Call ID
- Invalid SNP
- Unavailable SNP

10.1.2 Connection Release Messages

10.1.2.1 Connection Release Request

The connection release request message is used by either the eNNI-U or eNNI-D to forward a release request towards the corresponding protocol controller (eNNI-D or eNNI-U respectively). This message provides information needed to release the E-NNI portion of the end-to-end connection and additional information that may be used by either eNNI-U or eNNI-D to complete the end-to-end request. No error codes are carried in this message. This message is specified as having a global scope, i.e., the message is transmitted end-to-end. Table 9 shows the abstract message and its content.

Table 9: ConnectionReleaseRequest Abstract Message Attributes

Message: ConnectionReleaseRequest Scope: Global Direction: eNNI-D eNNI-U or eNNI-U eNNI-D			
Attributes	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Mandatory	Local	Connection
Terminating NCC PC ID	Mandatory	Local	Connection
Connection Name	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call

Note: In the case where a release request is generated at an intermediate point of the connection (i.e., not at an endpoint) the connection notification message is used to notify the source or destination of the connection to initiate the release request. As such, in this scenario the connection notification generated at the intermediate CC triggers either the source or destination CC to initiate a release request, and it may include additional information to mark this request as intermediate-generated. No error codes are carried in this message.

10.1.2.2 Connection Release Indication

The connection release indication message is used by either the eNNI-D or eNNI-U to respond to a release request towards the corresponding protocol controller (eNNI-U or eNNI-D, respectively). This message provides final confirmation of the release of the E-NNI portion of the end-to-end connection. It has global scope, i.e., the message is transmitted end-to-end. Table 10 shows the abstract message and its content.

Table 10: ConnectionReleaseIndication Abstract Message Attributes

Message: ConnectionReleaseIndication Scope: Global Direction: eNNI-D eNNI-U or eNNI-U eNNI-D			
Attributes	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Mandatory	Local	Connection
Terminating NCC PC ID	Mandatory	Local	Connection
Connection Name	Mandatory	Local	Connection
Connection Status	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call
Error Code	Optional	Local	Connection

The following error codes are defined for this message:

- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid / unknown connection ID
- Invalid / unknown Call ID

10.1.3 Connection Query Messages

10.1.3.1 Connection Query Request

The connection query request message is used by either the eNNI-U or eNNI-D to initiate a query request towards the corresponding protocol controller (eNNI-D or eNNI-U, respectively). This message requests connection specific information for one (or more) established connections. If no Call or Connection Names are specified, all connections managed by the eNNI-D or eNNI-U MUST be returned. If only the Call Name is specified, then information for all connections within a call is returned. This message is specified as having a local scope. Table 11 shows the abstract message and its contents.

Table 11: ConnectionQueryRequest Abstract Message Attributes

Message: ConnectionQueryRequest Scope: Local Direction: eNNI-D eNNI-U or eNNI-U eNNI-D			
Attributes	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Optional	Local	Connection
Terminating NCC PC ID	Optional	Local	Connection
(List of) Connection Names	Optional	Local	Connection
Call Name	Optional	Global	Call

10.1.3.2 Connection Query Indication

The connection query indication message is used by either the eNNI-U or eNNI-D to respond to a query request towards the corresponding protocol controller (eNNI-D or eNNI-U, respectively). This message provides connection specific information for one or more established connections. If the status of multiple connections has been requested, the following contents are repeated for each connection.

- Connection Name
- Connection specific details

This message is specified as having a local scope. Table 12 shows the abstract message and its content.

Table 12: ConnectionQueryIndication Abstract Message Attributes

Message: ConnectionQueryIndication Scope: Local Direction: eNNI-D eNNI-U or eNNI-U eNNI-D			
Attribute	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Mandatory	Local	Connection
Terminating NCC PC ID	Mandatory	Local	Connection
Connection Name	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call
Error Code	Optional	Local	Connection
Connection specific details below			
Connection Status	Mandatory	Local	Connection
Source TNA Name	Optional	Local	Connection
Destination TNA Name	Optional	Local	Connection
Local Logical Port Identifier	Optional	Local	Connection
Local Generalized Label	Optional	Local	Connection
Contract ID	Optional	Local	Connection
Encoding Type	Optional	Local	Connection
Switching Type	Optional	Local	Connection
SONET/SDH, OTN or Ethernet traffic parameters	Optional	Local	Connection
Directionality	Optional	Local	Connection
Explicit Route	Optional	Local	Connection
Generalized Payload Identifier	Optional	Local	Connection
Service Level	Optional	Local	Connection
Diversity	Optional	Local	Connection

The *connection specific details* attribute may contain any attributes associated with the connection. The types of information that may be carried as part of the *connection specific details* are those defined in the Section 10.2 and Table 17. This attribute may contain the status of one or more connections. For example, these may include the SNP used, the particular recovery mechanism used, etc.

The following error codes are defined for this message, and apply to the message as a whole, and not to the individual connections:

- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid / unknown connection ID
- Invalid / unknown Call ID

10.1.4 Connection Notification Messages

10.1.4.1 Connection Notification

The connection notification message is used by either the eNNI-U or eNNI-D to initiate a notification towards the corresponding protocol controller (eNNI-D or eNNI-U, respectively).

This message notifies the source or destination of the need to delete the connection and should result in the source or destination triggering a connection release request. It has global scope. Table 13 shows the abstract message and its content.

Table 13: ConnectionNotification Abstract Message Attributes

Message: ConnectionNotification Scope: Local Direction: eNNI-D eNNI-U or eNNI-U eNNI-D			
Attribute	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Mandatory	Local	Connection
Terminating NCC PC ID	Mandatory	Local	Connection
(List of) Connection Names	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call
Error Code	Optional	Local	Connection
Connection Status	Optional	Local	Connection

Note that an Error Code can be generated anywhere along the path, and that Error Code is passed hop by hop back along the path. Connection Status may be used to convey management intent, and, just as the Error Code, can be originated anywhere along the path. It is passed back hop by hop.

The following error codes are defined for this message:

- Service-affecting defect (resulting in failed connection)
- Non-service-affecting defect (no failed connection)

10.1.5 Connection Modify Messages

10.1.5.1 Connection Modify Request

The connection modify request message is sent by the eNNI-U towards the eNNI-D to request a modification to an existing connection. For E-NNI 2.0, the connection bandwidth can be modified. (In [OIF-UNI-02.0], the bandwidth can be modified. For Ethernet EVPL services, the generalized label representing the CE-VLAN identifiers can be modified. However, only the connection bandwidth modification is acted upon at the E-NNI.) No error codes are carried in this message. Table 14 shows the abstract message and its content.

Table 14: ConnectionModifyRequest Abstract Message Attributes

Message: ConnectionModifyRequest Scope: Global Direction: eNNI-U eNNI-D				
Attribute	**Type**	**Scope**	**Call/Conn**	**Modifiable**
Source TNA	Mandatory	Global	Call	No
Destination TNA	Mandatory	Global	Call	No
Initiating NCC PC ID	Mandatory	Local	Call	No
Terminating NCC PC ID	Mandatory	Local	Call	No
Connection Name	Mandatory	Local	Connection	No
Call Name	Mandatory	Global	Call	No
SNP ID	Optional	Local	Connection	Yes
SNPP ID	Mandatory	Local	Connection	No
DEST SNP ID	Optional	Local	Call	Yes
Explicit Route	Optional	Global	Connection	Yes
SONET/SDH, OTN or Ethernet traffic parameters	Mandatory	Global	Call/ Connection	Yes

10.1.5.2 Connection Modify Indication

The connection modify indication message acknowledges the modification of the connection to the eNNI-U that initiated the connection modification request. The connection modify indication message is sent from the eNNI-D towards the eNNI-U. Table 15 shows the abstract message and its content.

Table 15: ConnectionModifyIndication Abstract Message Attributes

Message: ConnectionModifyIndication Scope: Global Direction: eNNI-D eNNI-U			
Attribute	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Mandatory	Local	Call
Terminating NCC PC ID	Mandatory	Local	Call
Connection Name	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call
SNP ID	Optional	Local	Connection
SNPP ID	Mandatory	Local	Connection
Connection Status	Mandatory	Local	Connection
Error Code	Optional	Local	Call/ Connection

No additional Error Codes are added for this message.

10.1.5.3 Connection Modify Confirm

The connection modify confirm message is sent by the eNNI-U to respond to a setup indication towards the eNNI-D. This message provides final confirmation of modification of the E-NNI portion of the end-to-end connection. This message is specified as having a global scope, i.e., the message is transmitted end-to-end. Table 16 shows the abstract message and its content.

Table 16: ConnectionModifyConfirm Abstract Message Attributes

Message: ConnectionModifyConfirm Scope: Global Direction: eNNI-U eNNI-D			
Attribute	**Type**	**Scope**	**Call/Conn**
Initiating NCC PC ID	Mandatory	Local	Call
Terminating NCC PC ID	Mandatory	Local	Call
Connection Name	Mandatory	Local	Connection
Connection Status	Mandatory	Local	Connection
Call Name	Mandatory	Global	Call
Error Code	Optional	Local	Call / Connection

10.2 Abstract Attributes

Table 17 lists the abstract attributes to support the E-NNI signaling, based on the attributes defined in [G.7713] and [OIF-UNI-02.0].

Table 17: E-NNI Signaling Abstract Attributes

Abstract Attributes		**Reference**	**Cross Ref**
Identification Attributes	Source TNA name	7.1, 10.2.1.1	From [OIF-UNI-02.0]; see Table 1
	Destination TNA name	7.1, 10.2.1.1	From [OIF-UNI-02.0]; see Table 1
	Initiating NCC PC ID	7.2	See Table 2
	Terminating NCC PC ID	7.2	See Table 2
	Connection Name	10.2.3	"Local Connection ID" in [OIF-UNI-02.0] and the "connName" in [G.7713]
	Call Name	10.2.3.1	Known as "callName" in [G.7713]
	SNP ID	7.1, 10.2.3.2	From [G.7713]; see Table 1
	SNPP ID	7.1, 10.2.3.3	From [G.7713]; see Table 1
	DEST SNP ID	10.2.3.4	See Table 1
Service Attributes	Directionality	10.2.4.1	Same in [OIF-UNI-02.0] and [G.7713]
	Encoding Type	10.2.4.2	[OIF-UNI-02.0], Section 10.13.2.1
	Switching Type	10.2.4.3	[OIF-UNI-02.0], Section 10.13.2.2
	SONET/SDH, OTN, or Ethernet traffic parameters	10.2.4.4	See [OIF-UNI-02.0] Section 10.13.2.3
Routing Attributes	Explicit Route	10.2.5.1	Not in [OIF-UN1]. Known as "Explicit Resource List" in [G.7713]
Policy Attributes	Service Level	10.2.6.1	This attribute is the same in [OIF-UNI-02.0] and is a combination of "Class of Service" and "Grade of Service" in [G.7713]
Miscellaneous Attributes	Connection Status	10.2.7.1	From [OIF-UNI-02.0]
	Error Code	10.2.7.2	From [OIF-UNI-02.0]

10.2.1 Identification Related Attributes

10.2.1.1 Source / Destination TNA

These attributes are described in Section 7.1. The Source TNA attribute is validated by the source control domain and carried in the message across the E-NNI. The Destination TNA attribute may be used within the egress domain to determine a route within that domain to reach the user.

10.2.2 Initiating / Terminating NCC PC

These attributes are described in Section 7.2.

10.2.3 Connection Name

This attribute represents a locally unique name assigned to a connection. The protocol controller initiating a setup request assigns the connection name.

10.2.3.1 Call Name

This attribute represents a globally unique name assigned to a call at the UNI call boundary (and, as such, the E-NNI is not involved with call name assignment other than verification of a valid or known call name, except as described in Section 14.2.1). The call name is assumed to remain constant across call modification operations, i.e., across the “life” of the call.

10.2.3.2 SNP ID

This attribute represents the transport plane link connection resource used to support the requested service over the E-NNI. The eNNI-U may choose an SNP to use; however, the eNNI-D may override this SNP with a different SNP.

10.2.3.3 SNPP ID

This attribute represents transport link resources used to support the requested service over the E-NNI. An SNPP ID identifies the transmitting (requesting) end of an SNPP Link; however, as the two ends of the SNPP Link are assumed to be correlated with each other (either automatically via discovery or manually via provisioning), specifying a single endpoint identifies the SNPP Link to be used. Different addressing formats may be implemented for this attribute. For example, based on [RFC4201], it could be a numbered or unnumbered link, or a numbered or unnumbered bundle link.

10.2.3.4 DEST SNP ID

This attribute represents the SNP used at the destination network-to-user connection. The DEST SNP ID is locally unique to the destination end and is provided to the control plane by an external agent (e.g., the management system that initiated the SPC connection request). The attribute can represent either a UNI-C or a UNI-N SNP or Access Group, and it can be used in either SC or SPC scenarios. Note that in the case of an SPC connection, both source and destination user-to-network connections are provisioned. As such, when setting up a network switched connection to support the SPC service, the destination SNP associated with the provisioned connection needs to

be specified to allow the control plane to connect the switched connection to the destination portion of the provisioned connection.

10.2.4 Service Related Attributes

10.2.4.1 Directionality

This attribute indicates whether the requested service is for a unidirectional or bidirectional connection.

10.2.4.2 Encoding Type

The Encoding Type specifies the encoding format of the signal to be transported across the UNI. The encoding options specified are:

- SDH ITU-T [G.707]
- SONET ANSI [T1.105]
- ODUk layer [G.709]
- OCh layer [G.709]
- Ethernet IEEE PHY 802.3 [IEEE802.3]

10.2.4.3 Switching Type

This indicates the type of switching that should be performed on a particular link. The switching type options specified are Layer 2 (Ethernet), TDM (SONET, SDH, OTN ODUk), Lambda (OTN OCh), or Data Channel Switching Capable (DCSC).

10.2.4.4 Traffic Parameters

Traffic parameters are specified for SONET/SDH, OTN, and Ethernet. For full details see [OIF-UNI-02.0] Section 10.13.2.3.

10.2.5 Routing Related Attributes

10.2.5.1 Explicit Route

This attribute represents a list of links and subnetworks that the requested service traverses. Depending on the amount of routing information exchanged, this attribute may contain detailed route information or an abstracted view.

10.2.6 Policy Related Attributes

10.2.6.1 Service Level

This attribute contains information about the service level associated with the requested service across the E-NNI interface. It is necessary to enable a consistent translation of “equivalent” service level information between control domains.

10.2.7 Miscellaneous Attributes

10.2.7.1 Connection Status

This indicates the status of a connection.

10.2.7.2 Error Code

The error code is used to describe the errors resulting from connection actions.

- Calling Party Busy
- Called Party Busy
- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Invalid / unknown connection ID
- Invalid / unknown Call ID
- Invalid SNP
- Unavailable SNP
- Invalid SNPP
- Unavailable SNPP
- Unavailable directionality
- Invalid SPC SNP
- Invalid route
- Unavailable service level
- Service-affecting defect (resulting in failed connection)
- Non-service-affecting defect (no failed connection)

11 E-NNI Security and Logging

11.1 Security

Because optical calls and connections carry high volumes of data and consume significant network resources, security is required to safeguard transport networks against attacks that may compromise their control plane, seek disruption or unauthorized use of their resources, or attempt to gain unauthorized information about their configuration or use.

The requirements for and goals of security functionality for the E-NNI are specified in Section 6.5.4 of UNI 2.0 Common [OIF-UNI-02.0].

The details for implementing security mechanisms satisfying these requirements are provided in the *Security Extension for the UNI and NNI* [OIF-SEC] and the *Addendum to the Security Extension for the UNI and NNI* [SecAdd]. These Implementation Agreements describe optional mechanisms (1) to authenticate entities exchanging information across a Control Plane interface; (2) to guarantee the integrity of this information; and (3) to protect the confidentiality of this information where required.

The security mechanisms for the E-NNI found in [OIF-SEC] and [SecAdd] consist of a single, *optional-to-implement* security capability for all E-NNI Control Plane protocols based on a profile of IPsec. These mechanisms, when present, offer a complete set of protocol security capabilities and contain a mandatory-to-implement option to promote *interoperability* of security. These mechanisms also provide a large number of additional, potentially desirable features including replay detection, key exchange and key update, protection against certain denial of service or traffic analysis attacks, optimizations to improve efficiency, and anonymity of the parties with respect to eavesdroppers.

11.2 Logging

Implementations supporting the Security Extension SHOULD also support logging as specified in *OIF Control Plane Logging and Auditing with Syslog* [SysLog] and the guidelines listed below. This requirement includes all of the capabilities in [SysLog] to configure, control, and secure these records. Implementations not supporting the Security Extension MAY support logging. The following guidelines specify how [SysLog] should be used to support E-NNI 2.0:

- The CONFIG@26041 Structured Data item SHOULD additionally contain the E-NNI 2.0 identifiers defined in this Implementation Agreement.
- The PROT@26041 Structured Data item SHOULD contain the entire packet including network layer headers. Further formatting of this Structured Data item is NOT RECOMMENDED.
- Additional capabilities to control the generation and disposition of records containing the PROT@26041 Structured Data item according to abstract message type or E-NNI identifiers MAY be provided.
- A new CONNECT@26041 Structured Data item corresponding to the CALL@26041 record defined in [SysLog] SHOULD be provided.
- The SUM@26041 Structured Data item SHOULD include counts of calls, connections, messages by types, and errors by types.

Messages containing the NE_ID@26041, CONFIG@26041, PROT@26041, CALL@26041, CONNECT@26041, and SUM@26041 Structured Data items SHOULD be logged with SEVERITY Informational (6). E-NNI error messages, Signaling Channel failures, and Control Plane failures MUST be logged with a higher (i.e., lower numbered) SEVERITY than Informational.

12 E-NNI Signal Flow

This section provides a general protocol neutral description of the signal flow for the E-NNI interface. In the following discussion, there is no implication that the same protocol is used at every E-NNI interface consistent with support for protocol heterogeneity as described earlier within the Introduction and Section 6. No assumptions are made regarding the protocols used at the UNI or for the I-NNI. In this section, the UNI and I-NNI signal flows illustrated in all figures must be interpreted to represent the information exchange necessary for operation of the E-NNI; they are not meant to show the signaling flows for UNI or I-NNI.

12.1 Normal (Successful) Operation

This section provides the E-NNI signal flow for normal setup and release requests. This includes setup and release for both switched connections (i.e., UNI initiated) and soft permanent connections (i.e., management system initiated).

12.1.1 Normal Setup Request

To support an SPC, it is assumed that the transport plane connections between the source user and the network, and the destination user and the network, have been provisioned. Control plane operation only provides the connectivity within the network needed to create the end-to-end view of the connection. To initiate setup for the network portion of the connection, a management system requests the control plane to set a network connection up. Information associated with the request is initially sent to the ingress control domain and then propagated across the E-NNI signaling interface. This process continues until the request is received by the destination control domain. The request is completed after multiple message exchanges to set the connection up. The E-NNI signaling interface assumes that there will be three-message handshaking involved in setting up the E-NNI portion of the network connection. Note that the third message (Setup Confirm) is optional, but MUST be included if requested by the destination. Figure 18 illustrates the signal flow across the E-NNI signaling interface, where I-NNI flows are provided for the sake of completeness.

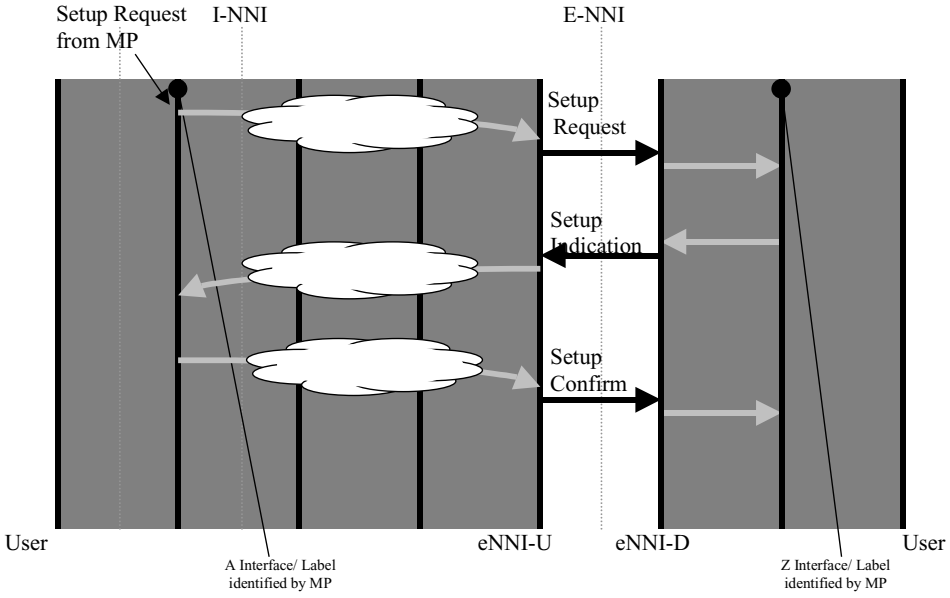


Figure 18: Basic Soft Permanent Connection Setup

In the case of a SC, a control plane operation is used to support the setup of the end-to-end connection. Note that this case assumes the use of a UNI signaling protocol for initiating the request. Figure 19 illustrates the signal flow across the E-NNI signaling interface, where UNI and I-NNI flows are provided for the sake of completeness.

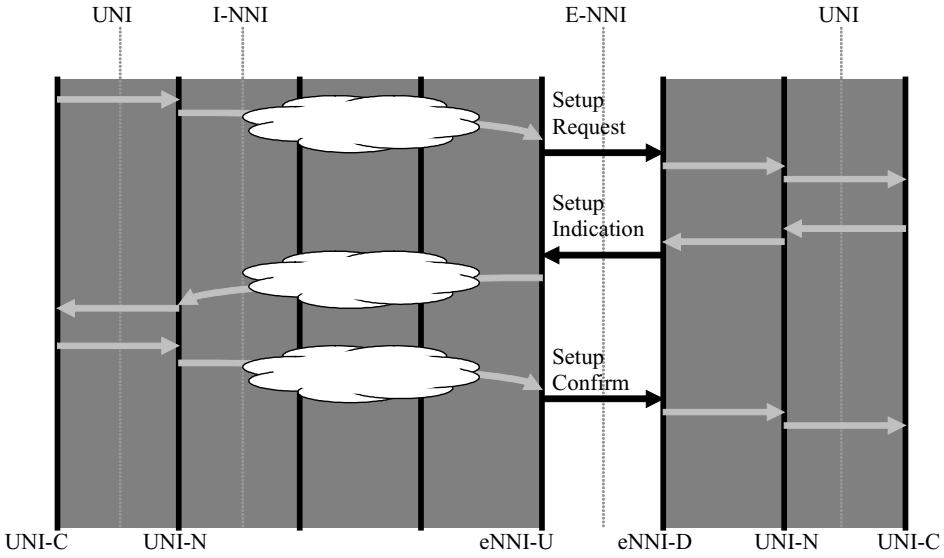


Figure 19: Basic Switched Connection Setup

12.1.2 Normal Release Request

In the case of an SPC, to initiate release for the network portion of the connection, a management system requests the control plane to release a network connection. This is independent of any actions that may be taking place for the provisioned portion of the connection (e.g., whether the provisioned connection is released before or after the switched network connection). The E-NNI signaling interface assumes a two-way handshake for releasing the E-NNI portion of the network connection. By indicating the intention to release a connection, alarms can be suppressed along the path of the connection before the cross-connections are removed. This is indicated by the Start to Release event in Figure 20 through Figure 22. Figure 20 illustrates the signal flow (from left to right) across the E-NNI signaling interface.

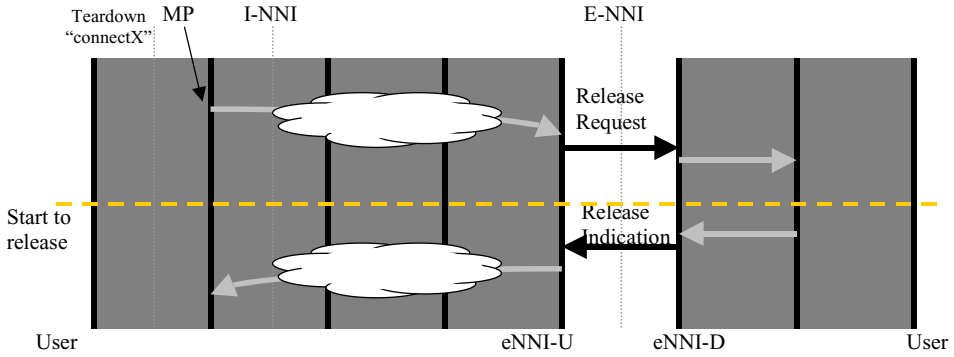


Figure 20: Basic SPC Release

To release a SC, several signal flows may be followed. This depends on who initiated the release request. Control plane operation is used in this case to support the release of the end-to-end connection. The E-NNI signaling interface assumes that there will be two-message handshaking for releasing the E-NNI portion of the network connection. Figure 21 and Figure 22 illustrate the signal flow across the E-NNI signaling interface for the cases of source UNI-C and destination UNI-C initiated releases, respectively.

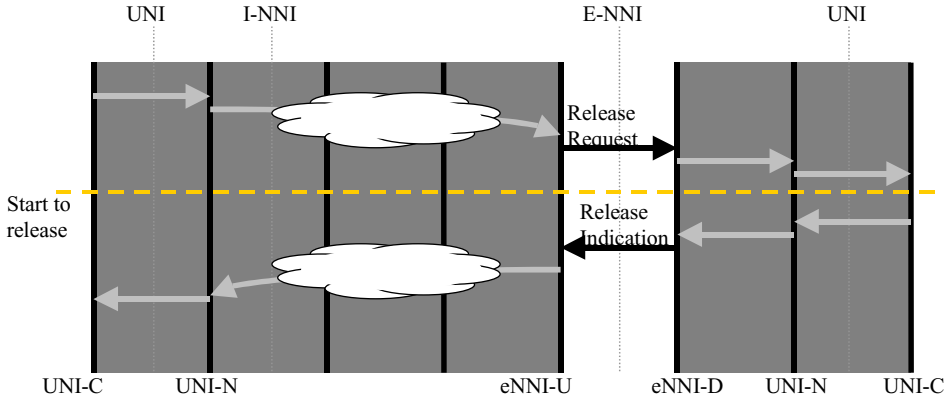


Figure 21: Basic SC Release: Source UNI-C Initiated

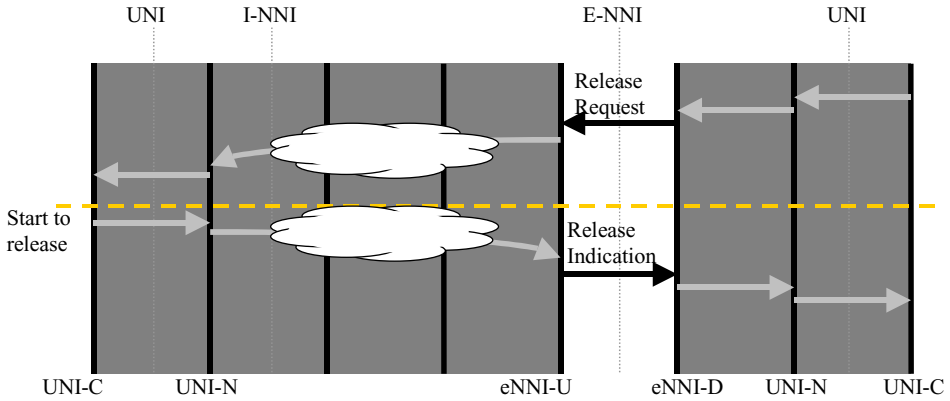


Figure 22: Basic SC Release: Destination UNI-C Initiated

Additional scenarios may exist for the release of an SPC or SC connection whereby intermediate control plane nodes (connection controllers) may initiate the connection release. This may occur because of a failure of the connection, or because of policies associated with connection release across domains. In this instance, an intermediate control plane node (CC) initiates a connection release by sending a release notification. To avoid a condition where releasing a connection may result in propagation of alarms, the intermediate node (CC) first notifies the source or destination nodes of the need to initiate a release. Subsequent actions follow those of Figure 20, Figure 21, or Figure 22 above, as appropriate. For E-NNI Signaling 2.0, a release notification initiated from the eNNI-U or eNNI-D node is always sent in the upstream direction. These cases are shown in Figure 23 and Figure 24. Figure 25 and Figure 26 illustrate the signal flows during a network node initiated release request.

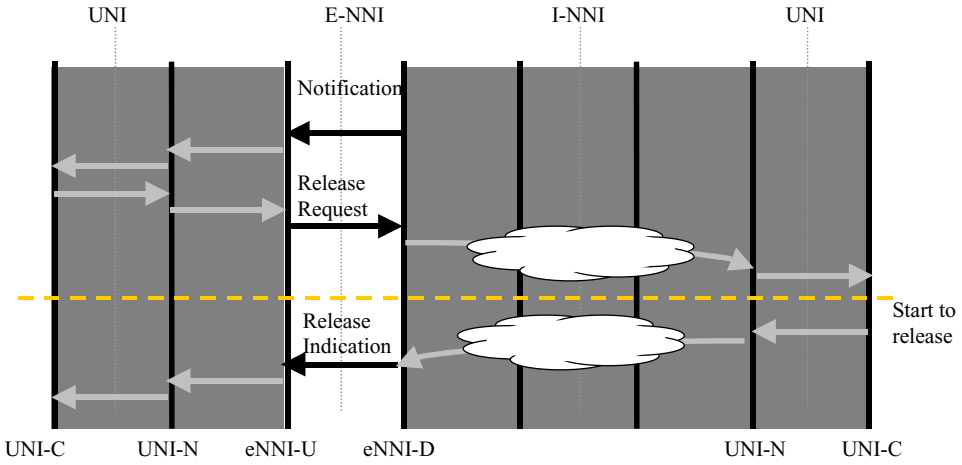


Figure 23: Teardown/Release Initiated by eNNI-D

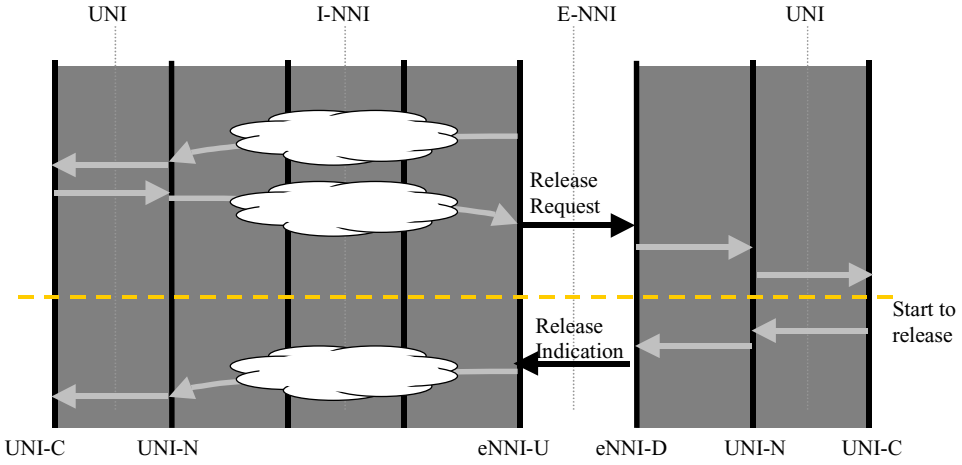


Figure 24: Teardown/Release Initiated by eNNI-U

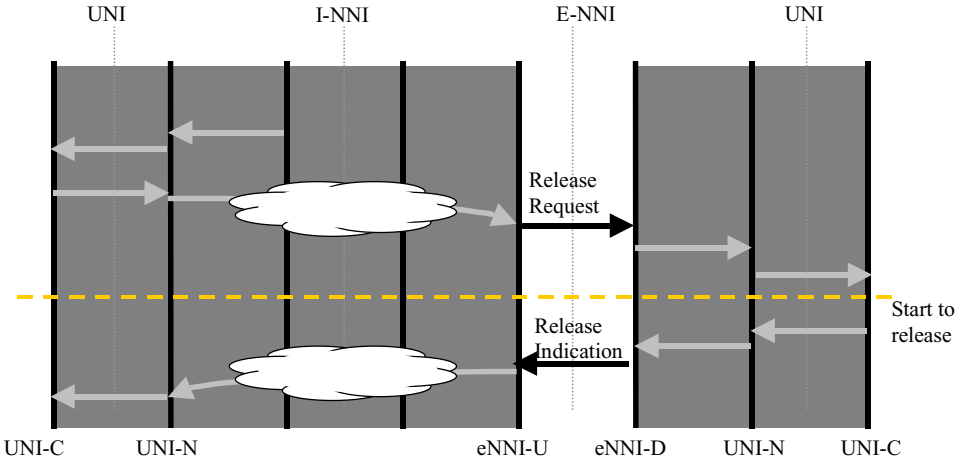


Figure 25: Teardown/Release Initiated by an Upstream Network Node

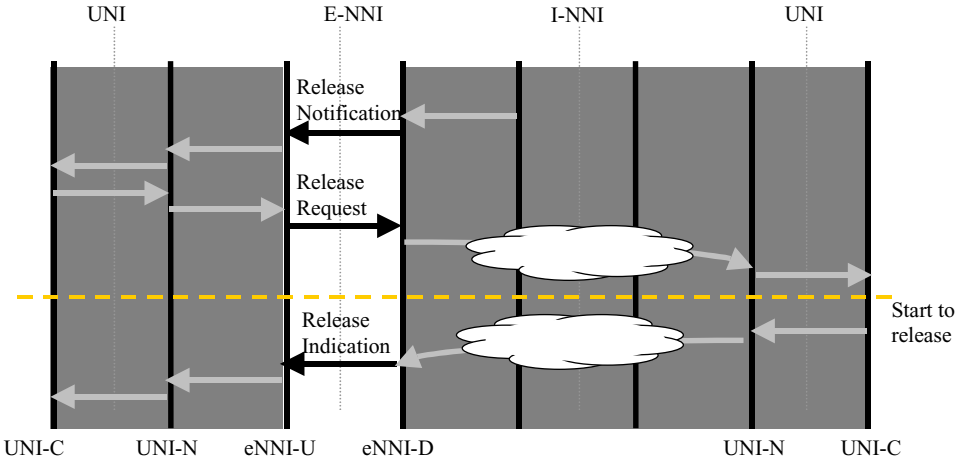


Figure 26: Teardown/Release Initiated by a Downstream Network Node

It is possible that the network could initiate a release notification in the downstream direction. This case is also possible when the release is initiated by an eNNI-U or eNNI-D node that supports the E-NNI Signaling 1.0 specification. In this case, the E-NNI 2.0 node continues to propagate the release notification towards the destination node. Once the release notification reaches the destination node, then the signaling behavior follows that shown in Figure 22. The signal flow for this case is shown in Figure 27.

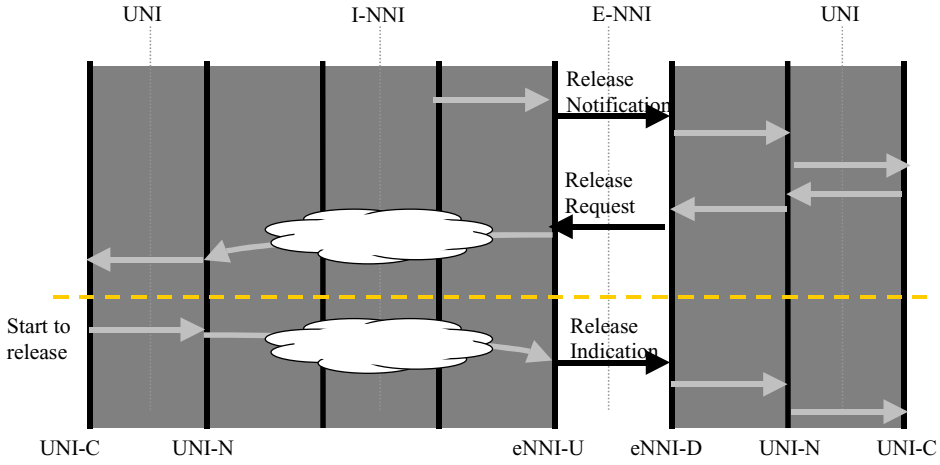


Figure 27: Teardown/Release Initiated in Downstream Direction

12.1.3 Connection Modify Request

E-NNI 2.0 supports modification of existing calls and connections. Call modification is supported by adding or removing connections to or from an existing call. Connection modification is supported by modifying the bandwidth of an existing connection. Figure 28 illustrates the signal flow across the E-NNI signaling interface, where UNI and I-NNI flows are provided for the sake of completeness.

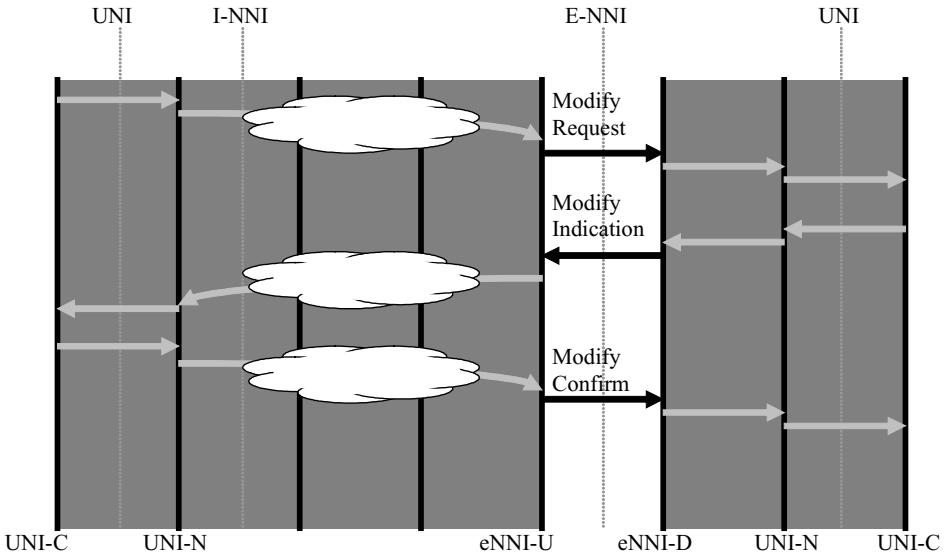


Figure 28: Connection Modification Request

12.2 Exception and Defect Handling Operations

This section describes the E-NNI signal flows for exception and defect handling. Note that this section does not provide information on what causes the control plane to reject a request. There may be many reasons ranging from resource conflicts to policy decisions to request time-outs. This section only provides the signal flow (and thus the behaviors) as a result of the decision to reject a request.

12.2.1 Setup Request Rejections

A request may be rejected while in different phases of the signal flow. The cause of the rejection is beyond the scope of this document; however, as a result of a decision to reject a request certain signal flows must be followed. If logging is used, the rejection SHOULD be logged with a severity higher than Informational. A rejection may occur during processing of the request or indication messages:

- During the initial request message, any nodes (CCs) along the message path (up to and including the destination node, NCC) may have cause to reject the request. As a result of this rejection, the control plane node (CC) that decides to reject the request sends a Setup Indication Message with an error (to release any pending resources) towards the source node (NCC).

- During the indication response message, any nodes (CCs) along the message path (up to and including the source node, NCC) may have cause to reject the request. As a result of this rejection, the control plane node (CC) that decides to reject the request sends a Setup Indication Message with an error (to release any pending resources) towards the source node (NCC). In addition, a downstream confirmation message with an error code is sent to release any reserved resources.

Figure 29 illustrates the signal flows that occur when a setup indication is rejected.

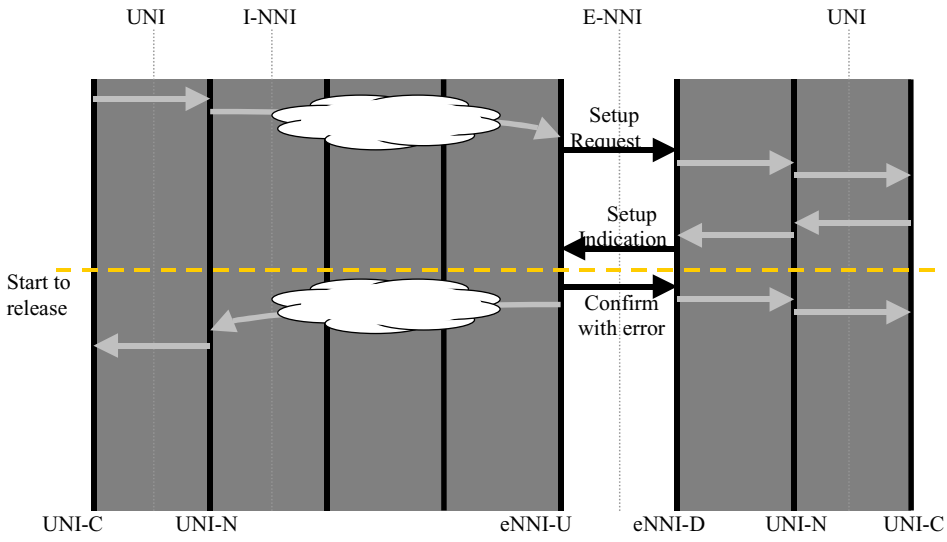


Figure 29: Connection Setup Indication Rejection

12.2.2 Release Request Rejections

The only scenario in which a release request may be rejected is if the request is not valid. For example, a message may have improper, missing, or contradictory parameters (e.g., a request to release something that does not exist) that prevent the request from being carried out. Note that when using the Security Extension, improperly authenticated messages will be discarded at the network layer.

For all other scenarios, a release request must be carried out. If there were errors introduced during the release operation that resulted in connections not been released, then error messages may be sent to an external controller (e.g., management system); however, the response to a release request should be successful. This is critical because a validated release request implies that billing for a user-requested service must be stopped and resources associated with the requested connection must be released (to prevent partial connections).

Consequently, for release requests that are validated, the normal release request signal flows in Section 12.1.2 are applicable; for release requests that are not validated, a message specifying unauthorized request may be sent. For release actions that cannot be completed as a result of errors, an alarm may also be raised to indicate the error. Action resulting from the alarm relies on decisions made by the external controller and is thus outside the scope of this document.

If logging is used, the rejection or error SHOULD be logged with a severity higher than Informational.

12.2.3 Modification Request Rejections

Call Modification either adds a new connection or modifies an existing connection. For modification of an existing connection, the modification procedure uses a make-before-break procedure to establish a new connection and then deletes the original connection once the new connection is established successfully. In both call modification cases, a failure during the modification procedure should result in reverting to the original, pre-modification state.

A modification request can fail due to a number of reasons such as an invalid modification request, failure to allocate additional bandwidth, or a signaling failure. An E-NNI node rejecting a modification request SHOULD respond by sending a ConnectionSetupIndication (for call modification adding a new connection) or a ConnectionModificationIndication (for call modification by modifying an existing connection) with the Error Code indicating the reason for the failure. The original connections that existed prior to the modification request should not be affected by the failure.

Alarms for call modification failures are generated at the source UNI-N or UNI-C. Also, the source UNI-N is responsible for the retrying a failed bandwidth decrease attempt as described in [OIF-UNI-02.0-RSVP] Section 8.10.2.

If logging is used, the rejection SHOULD be logged with a severity higher than Informational

12.2.4 Signaling Channel Failure

A signaling channel failure MUST NOT result in the release of established connections. Setup requests in the process of being completed may be removed (either during the failure or after recovery from failure). Established connections associated with a pending release request MUST be released (either during the failure or after recovery from failure). If logging is used, signaling channel failure SHOULD be logged with a severity higher than Informational.

12.2.5 Control Plane Failure

A control plane failure (including a failure to receive a refresh message whether the signaling adjacency is operational or not) MUST NOT result in the release of established connections. Setup requests in the process of being completed may be removed (either during the failure or after recovery from failure). Established connections associated with a pending release request MUST

be released (either during the failure or after recovery from failure). If logging is used, control plane failure SHOULD be logged with a severity higher than Informational.

12.2.6 Transport Resource Failure

A transport resource failure (e.g., link or NE) may result in the release of established connections. This depends on the type of connection and the service level associated with each connection. Service levels are specific to each carrier, and their definition is outside the scope of this document. Service levels dictate behavior in case of failure of the transport resource, and failure of a transport resource may result in various protection and restoration actions depending on the selected service level. Note that even in the case of a protected connection, the original connection may be released while a new connection is set up (which also depends on the type of protection used for the particular connection). If logging is used, transport resource failure SHOULD be logged with a severity higher than Informational.

13 RSVP-TE Extensions for E-NNI Signaling

The RSVP-TE extensions to support the E-NNI signaling mechanism are based on the protocol capabilities as specified in [RFC2205], [RFC2961], [RFC3209], [RFC3473], [RFC3474], [RFC3476], [RFC4328], and [RFC4606]. This section provides a summary of the messages, objects, and error codes relevant to this Implementation Agreement.

13.1 Overview of RSVP-TE Operation

An overview of the basic RSVP-TE operation may be found in [RFC3209] and [RFC3473]. When a eNNI-U (eNNI-D) sends an RSVP message, it MUST address the message directly to its eNNI-D (eNNI-U) peer. The peer's SCN address is used for this purpose. A node (signaling protocol controller) should use IP encapsulation of RSVP messages. Furthermore, the IPv4 Router Alert option MUST NOT be set in any RSVP messages. The IPv4 header fields are shown in Table 18.

Table 18: IPv4 Header for E-NNI RSVP Messages

IPv4 Header Values for E-NNI RSVP Messages	
Version	4
Header Length	5
TOS	As defined in [RFC791]
Total Length	Message length
Identification	As defined in [RFC791]
Flags	As defined in [RFC791]
Fragment Offset	As defined in [RFC791]
TTL	>= 1
Protocol	46
Header Checksum	As defined in [RFC791]
Source Address	eNNI-U/eNNI-D SC PC SCN IP address
Destination Address	eNNI-D/eNNI-U SC PC SCN IP address

The format of the RSVP <Common Header> object is defined in [RFC2205], Section 3.1.1.

The flag field of the RSVP common header MUST be set to 1, to indicate support of the Bundle and Srefresh messages. As a result, an implementation MUST be able to process Bundle and

Srefresh messages received from a neighbor, and MAY choose to use bundling and Srefresh when sending messages.

13.2 Messages and Error Codes

Table 19 provides a mapping of the abstract connection messages to specific RSVP-TE messages used to support signaling across the E-NNI interface.

Table 19: Mapping of Abstract Messages to RSVP-TE Messages

Abstract Messages	RSVP-TE Messages
ConnectionSetupRequest	Path
ConnectionSetupIndication	Resv In case of error - PathErr
ConnectionSetupConfirm	ResvConf In case of error - PathTear
ConnectionReleaseRequest	Path or Resv (with Delete and Reflect bits (D&R bits)) Path or Resv (with Admin Down and Reflect (A&R bits)) – only for compatibility with UNI/E-NNI 1.0
ConnectionReleaseIndication	PathTear or PathErr (w/ Path_State_Removed flag)
ConnectionQueryRequest	Implicit
ConnectionQueryIndication	Implicit
ConnectionNotification	Notify (w/ D bit set) or PathErr
ConnectionModifyRequest	Path
ConnectionModifyIndication	Resv In case of error - PathErr
ConnectionModifyConfirm	ResvConf In case of error - PathTear
Signaling Adjacency Maintenance	Hello

Table 20 provides information on which abstract messages are associated with each RSVP message.

Table 20: RSVP Messages by Abstract Message

RSVP-TE Messages	Abstract Messages
Path	ConnectionSetupRequest ConnectionReleaseRequest ConnectionModifyRequest
Resv	ConnectionSetupIndication ConnectionReleaseRequest ConnectionModifyIndication
PathErr	ConnectionSetupIndication ConnectionReleaseIndication ConnectionNotification ConnectionModifyIndication
ResvConf	ConnectionSetupConfirm ConnectionModifyConfirm

RSVP-TE Messages	Abstract Messages
PathTear	ConnectionSetupConfirm ConnectionReleaseIndication
Notify (w/ D bit set)	ConnectionNotification
Hello	Signaling Adjacency Maintenance

Table 21 provides a mapping of the abstract error codes to specific RSVP-TE error codes and values used to support signaling across the E-NNI interface. These error codes are specified in [RFC2205], [RFC3209], and [RFC4974].

Table 21: Mapping of Abstract Error Codes to RSVP-TE Error Codes and Values

Abstract Errors	RSVP-TE Error Codes/Error Values
Calling Party Busy	ERROR_SPEC 24/5
Called Party Busy	ERROR_SPEC 24/103
Unauthorized sender (policy error)	ERROR_SPEC 2/100
Unauthorized receiver (policy error)	ERROR_SPEC 2/101
Invalid / unknown connection ID	ERROR_SPEC 24/102
Invalid / unknown call ID	ERROR_SPEC 24/105
Invalid SNP	ERROR_SPEC 24/6 or 24/11 or 24/12 or 24/14
Unavailable SNP	ERROR_SPEC 24/6 or 24/11 or 24/12 or 24/14
Invalid SNPP	ERROR_SPEC 24/104
Unavailable SNPP	ERROR_SPEC 24/104
Unavailable directionality	ERROR_SPEC 24/6 or 24/11
Invalid SPC SNP	ERROR_SPEC 24/106
Invalid route	ERROR_SPEC 24/1, 24/2, 24/3, or 24/7
Invalid recovery	ERROR_SPEC 24/15 or 24/100
Unavailable recovery	ERROR_SPEC 24/15 or 24/100
Unavailable service level	ERROR_SPEC 24/101 or 2/{any}
Service-affecting defect	ERROR_SPEC 24/5, 24/9, 24/100, 24/101 or 24/103
Non-service-affecting defect	General protocol error: general RSVP-TE error codes/values

13.2.1 Hello Message (Msg Type = 20 [RFC3209])

This message is specified in [RFC3209]. Refer also to Section 14.2.9.

The Hello message is used to establish a signaling adjacency and for communications failure detection. It has the following format:

<Hello message> ::= <Common Header>
<HELLO>
<RESTART_CAP>

Hello messages are retransmitted periodically to an adjacent E-NNI signaling peer. The retransmission interval SHALL be administratively configurable. The default value is 5 seconds.

13.2.2 Path Message (Msg Type = 1 [RFC2205])

This message is specified in [RFC2205], with further extensions made by [RFC2961], [RFC3209], and [RFC3473].

<Path Message> ::=
<Common Header>
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
< SESSION> < RSVP_HOP>
<TIME_VALUES>
<GENERALIZED_LABEL_REQUEST>
<CALL_ID>
[<LABEL_SET> ...]
[<SESSION_ATTRIBUTE>]
[<EXPLICIT_ROUTE>]
<NOTIFY_REQUEST>
[<ADMIN_STATUS>]
<Generalized UNI>
[<POLICY_DATA> ...]
<sender descriptor>

Note: The GENERALIZED_UNI object is always present because it will either be provided by a UNI client initiating, adding, or deleting, a connection, or it will be provided by the ingress domain when initiating an SPC or hybrid connection to carry the source and destination TNAs, as well as other optional GENERALIZED_UNI sub-objects. This does not imply that intermediate NNI nodes should be able to process the GENERALIZED_UNI object, only that they are able to forward it between the source and destination of the connection. The format of the < Generalized UNI > and <sender descriptor> objects are described in [OIF-UNI-02.0-RSVP], Section 9.1.3 “Path Message”.

For SONET/SDH calls, for example,

<sender descriptor> ::=
<SENDER_TEMPLATE> <SENDER_TSPEC>
[<RECORD_ROUTE>]
[<UPSTREAM_LABEL>]

A Path message establishing a unidirectional connection over an NNI interface does not include an UPSTREAM_LABEL object.

13.2.3 Resv Message (Msg Type = 2 [RFC2205])

This message is specified in [RFC2205], with further extensions made by [RFC2961], [RFC3209], and [RFC3473]. The Resv message is used for connection creation and call/connection modification. The RESV_CONFIRM object may be included in the Resv message to request a ResvConf message to confirm the connection setup. Once the RESV_CONFIRM object has been included in the Resv message, it would normally be included in full refreshes of the Resv without generating additional ResvConf messages in response. A subsequent trigger change in the Resv message MAY result in a new ResvConf response. To force a new ResvConf message, the

RESV_CONFIRM object SHOULD be removed from the Resv message and then inserted into a subsequent Resv message so that the change acts as a trigger for a new ResvConf.

The format of the E-NNI Resv message is as shown below:

<Resv Message> ::= <Common Header>
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<SESSION> < RSVP_HOP >
<TIME_VALUES>
<CALL_ID>
<NOTIFY_REQUEST>
[<ADMIN_STATUS>]
[<POLICY_DATA> ...]
[<RESV_CONFIRM>]
<STYLE>
<FF flow descriptor> | <SE flow descriptor>

<FF flow descriptor> ::=
<SONET/SDH_FLOWSPEC> | <G.709 FLOWSPEC> | <ETH FLOWSPEC>
<FILTER_SPEC>
<GENERALIZED_LABEL>
[<RECORD_ROUTE>]

<SE flow descriptor> ::=
<SONET/SDH_FLOWSPEC> | <G.709 FLOWSPEC> | <ETH FLOWSPEC>
<FILTER_SPEC>
<GENERALIZED_LABEL>
[<RECORD_ROUTE>]

13.2.4 ResvConf Message (Msg Type = 7 [RFC2205])

This message is specified in [RFC2205], with further extensions made by [RFC2961], [RFC3209], and [RFC3473]. Note that the Call Name (CALL_ID) is not included in the ResvConf message, in accordance with [RFC3474]. As a result, the attributes are not aligned with the abstract Connection Setup Confirm message. It is still possible to correlate the ResvConf message with the proper Resv state based on the SESSION and flow descriptor.

The ResvConf message is originated at the source UNI-C to acknowledge the receipt of a trigger[3] Resv message that includes a RESV_CONFIRM Object. ResvConf messages are sent from the source UNI-C to the corresponding UNI-N, and from the destination UNI-N to the destination UNI-C. While the E-NNI processes ResvConf on a hop by hop basis, the message scope is end-to-end, and the network MUST relay the ResvConf message from source UNI-N to destination UNI-N.

The format of the ResvConf message is shown below:

[3] A trigger Resv message is a message that modifies the reservation state. Examples include the original Resv message sent during connection establishment in response to the first Path message and the Resv message that includes a change in the FILTER_SPEC object sent during connection modification.

<ResvConf message> ::= <Common Header>
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<SESSION> <ERROR_SPEC>
<RESV_CONFIRM>
<STYLE>
<FLOW_SPEC > | <FILTER_SPEC>

13.2.5 PathTear Message (Msg Type = 5 [RFC2205])

This message is specified in [RFC2205], with further extensions made by [RFC2961], [RFC3209], and [RFC3473].

<PathTear Message> ::= <Common Header>
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<SESSION>
<CALL_ID>
<RSVP_HOP>
<sender descriptor> /* (see Section 13.2.2) */

13.2.6 PathErr Message (Msg Type = 3 [RFC2205])

This message is specified in [RFC2205], with further extensions made by [RFC2961], [RFC3209], and [RFC3473]. The PathErr message is used to report errors and for connection deletion.

The format of the E-NNI PathErr message is as follows:

<PathErr message> ::= <Common Header>
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<SESSION>
<CALL_ID>
<ERROR_SPEC>
[<ACCEPTABLE_LABEL_SET>]
[<POLICY_DATA> …]
<sender descriptor> /* see Section 13.2.2*/

13.2.7 Notify Message (Msg Type = 21 [RFC3473])

The Notify message is used to support intermediate node initiated deletion, and the ADMIN_STATUS object is mandatory. This message is specified by [RFC3473].

<Notify message> ::= <Common Header>
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<ERROR_SPEC>
<notify session lists> /* 1 or more notify sessions */

<notify session lists> ::= [<notify session lists>]
<upstream notify session> | <downstream notify session>

<upstream notify session> ::= <SESSION> <CALL_ID> <ADMIN_STATUS[4]>
[<POLICY_DATA>...] <sender descriptor>
<downstream notify session> ::= <SESSION> <CALL_ID> <ADMIN_STATUS>
[<POLICY_DATA>...] <flow descriptor list>
<flow descriptor list> ::=
<FF flow descriptor> | <SE flow descriptor> /* 1 or more flow descriptor */

Note that the downstream notify session adds the ADMIN_STATUS object as per [RFC4974]. It enables the following situations to be handled:

a) a UNI/ENNI 1.0 node allows sending network deletion both upstream and downstream. If there was a UNI/ENNI 1.0 node upstream of an ENNI 2.0 interface, the ENNI 2.0 interface may receive a downstream network deletion request from the UNI/ENI 1.0 node.

b) a network I-NNI domain may initiate a downstream graceful deletion.

Note that the IETF model allows a NOTIFY Message to be sent to any recipient. The E-NNI 2.0 IA uses the NOTIFY Message only to perform Graceful Deletions for intermediate nodes, so the NOTIFY Message can only be directed to nodes with an upstream or downstream signaling adjacency with this node. The NOTIFY message has end-to-end significance, though it is processed on a hop by hop basis, and MUST be forwarded to continue the intermediate deletion message flow.

13.2.8 Srefresh Message

This message is specified by [RFC2961].

13.2.9 Ack Message

This message is specified by [RFC2961].

13.3 Attributes

Table 22 provides a mapping of the abstract attributes to specific RSVP-TE objects used to support signaling across the E-NNI interface.

Table 22: Mapping of Abstract Attributes to RSVP-TE Objects

Abstract Attributes	RSVP-TE Objects
Source TNA name	GENERALIZED_UNI/Source_TNA
Destination TNA name	GENERALIZED_UNI/Destination_TNA
DEST SNP ID	GENERALIZED_UNI/SPC_LABEL
Initiating NCC PC ID	SENDER_TEMPLATE
Terminating NCC PC ID	SESSION
Connection name	SESSION + SENDER_TEMPLATE
Call name	CALL_ID

[4]In E-NNI Signaling 2.0, the ADMIN_STATUS is mandatory in the Notify message because it is required for support of intermediate node initiated deletion. This does not imply that it is mandatory in future applications, and the presence of ADMIN_STATUS should not be used as a test for validity of the Notify message.

Abstract Attributes	RSVP-TE Objects
SNP ID	SENDER_TSPEC, RSVP_HOP, LABEL, GENERALIZED_UNI/EGRESS_LABEL
SNPP ID	RSVP_HOP[5], LABEL_SET
Directionality	Implied by UPSTREAM_LABEL
Explicit route	EXPLICIT_ROUTE, RECORD_ROUTE
Recovery	PROTECTION
Service level	GENERALIZED_UNI/DIVERSITY, GENERALIZED_UNI/SERVICE_LEVEL, POLICY_DATA, SESSION_ATTRIBUTE
Contract ID	POLICY_DATA
Encoding Type	GENERALIZED_LABEL_REQUEST/ LSP_ENC_TYPE
Switching Type	GENERALIZED_LABEL_REQUEST/ SWITCHING_TYPE
SONET/SDH, OTN or Ethernet traffic parameters	SONET/SDH_SENDER_TSPEC, SONET/SDH_FLOWSPEC G.709 TSPEC, G.709 FLOWSPEC ETHERNET SENDER_TSPEC, ETHERNET FLOWSPEC
Generalized Payload Identifier	GENERALIZED_LABEL_REQUEST/G-PID
Connection Status	ADMIN_STATUS

Table 23 provides a summary of the various objects used to support E-NNI signaling, along with codepoints assigned to the objects that are relevant to support the applications across the E-NNI interface. Note that this table only specifies the codepoints that are relevant to the OIF E-NNI specification and does not list all available codepoints (e.g., SENDER_TEMPLATE only lists C-type 7). Also, unless otherwise specified in this section, formats of these objects are as defined in the associated reference.

Table 23: Summary of RSVP-TE E-NNI Objects

RSVP-TE Object	Class-Num/ C-type[/ Type/ [Sub-type]]	Reference
ACCEPTABLE_LABEL_SET	130/{same as label_set}	[RFC3473], [RFC4328]
ADMIN_STATUS[1]	196/1	[RFC3473]
CALL_ID	230/<1,2>	[RFC3474]
ERROR_SPEC	6/3/{same as RSVP_HOP}[2]	[RFC2205], [RFC3209], [RFC3471], [RFC3473] Refer to Section 13.3.1
EXPLICIT_ROUTE	20/1/<3,4>[3]	[RFC3209], [RFC3473], [RFC3477] Refer to Section 13.4
FILTER_SPEC	10/{same as SENDER_TEMPLATE}	[RFC2205], [RFC3209], [RFC3473]

[5] The "IPv4 Next/Previous Hop Address" field of the RSVP_HOP object is not part of the SNPP ID as it is a control address.

RSVP-TE Object	Class-Num/ C-type[/ Type/ [Sub-type]]	Reference
SONET/SDH_FLOWSPEC	9/4	[RFC4606]
G.709 FLOWSPEC	9/5	[RFC4328]
ETHERNET FLOWSPEC	9/6	[ETH_PARAM]
GENERALIZED_UNI /DESTINATION_TNA	229/1/2/<1,2,3>	[OIF-UNI-02.0] Refer to Section 13.4.2
GENERALIZED_UNI /DIVERSITY	229/1/3/1	[OIF-UNI-02.0] Refer to Section 13.4.2
GENERALIZED_UNI /EGRESS_LABEL	229/1/4/1	[OIF-UNI-02.0] Refer to Section 13.4.2
GENERALIZED_UNI /SERVICE_LEVEL	229/1/5/1	[OIF-UNI-02.0] Refer to Section 13.4.2
GENERALIZED_UNI /SOURCE_TNA	229/1/1/<1,2,3>	[OIF-UNI-02.0] Refer to Section 13.4.2
GENERALIZED_UNI /SPC_LABEL[4]	229/4/2	[RFC3474] Refer to Section 13.4.2
HELLO_REQUEST/ HELLO_ACK	22/<1,2>	[RFC3209], [RFC3473]
RSVP_LABEL (GENERALIZED_LABEL)[5]	16/2	[RFC3473], [RFC4328]
GENERALIZED _LABEL_REQUEST	19/4	[RFC3473], [RFC4328]
LABEL_SET[6]	36/1	[RFC3473], [RFC4328]
MESSAGE_ID	23/1	[RFC2961]
MESSAGE_ID_ACK/ MESSAGE_ID_NACK	24/<1,2>[7]	[RFC2961]
MESSAGE_ID_LIST	25/1	[RFC2961]
NOTIFY_REQUEST	195/1	[RFC3473]
POLICY_DATA	14/1	[RFC2205]
PROTECTION	37/1	[RFC3473]
RECORD_ROUTE	21/{same as ERO}	[RFC3209], [RFC3473], [RFC3477]
RECOVERY_LABEL	34/{same as RSVP_LABEL}	[RFC3473], [RFC4328]
RESTART_CAP	131/1	[RFC3473]
RESV_CONFIRM	15/1	[RFC2205] Refer to Section 13.4.3
RSVP_HOP	3/3/<3,4,5>[8]	[RFC2205], [RFC3471], [RFC3473] Refer to Section 13.4.4
SENDER_TEMPLATE	11/7	[RFC2205], [RFC3209], [RFC3473] Refer to Section 13.4.5
SONET/SDH_TSPEC	12/4	[RFC4606]
G.709 TSPEC	12/5	[RFC4328]

RSVP-TE Object	Class-Num/ C-type[/ Type/ [Sub-type]]	Reference
ETHERNET TSPEC	12 / 6	[ETH_PARAM]
SESSION	1 / 15	[RFC2205], [RFC3209], Refer to Section 13.4.6
SESSION_ATTRIBUTE	207 / <1,7>	[RFC3209]
STYLE[9]	8 / 1	[RFC2205]
SUGGESTED_LABEL	129 / {same as RSVP_LABEL}	[RFC3473]
TIME_VALUES[10]	5 / 1	[RFC2205]
UPSTREAM_LABEL	35 / {same as RSVP_LABEL}	[RFC3473], [RFC4328]

Note 1: The absence of this object is equivalent to receiving an object containing values all set to zero (0).

Note 2: {text} where text is a comment.

Note 3: <…> indicates the different C-types or sub-types defined for the particular object.

Note 4: The port identifier contained in the SPC_LABEL sub-object is the logical port identifier assigned at the destination UNI-N; the port identifier contained in the EGRESS_LABEL sub-object is a logical port identifier assigned at the destination UNI-C.

Note 5: The format of LABEL is dependent on the signal types defined by LABEL_REQUEST object.

Note 6: A LABEL_SET object contains a list of “sub-channels” whose type is inferred from the label type field. Each of the sub-channels represents a label (wavelength, fiber, timeslot, etc.). A given LABEL_SET object MUST include a single label type. The interpretation of the label depends on the type of the link over which the label is to be used, so each sub-channel does NOT need its own header within the LABEL_SET object.

Note 7: MESSAGE_ID_NACK is a sub-type of MESSAGE_ID_ACK.

Note 8: RSVP_HOP Types 4 and 5 SHOULD NOT be generated but MUST be supported if received for E-NNI 1.0 backward compatibility.

Note 9: Both “fixed filter” and “shared explicit” styles are used.

Note 10: The value MUST be coordinated with Srefresh intervals to ensure proper refresh of the state information.

13.3.1 ERROR_SPEC

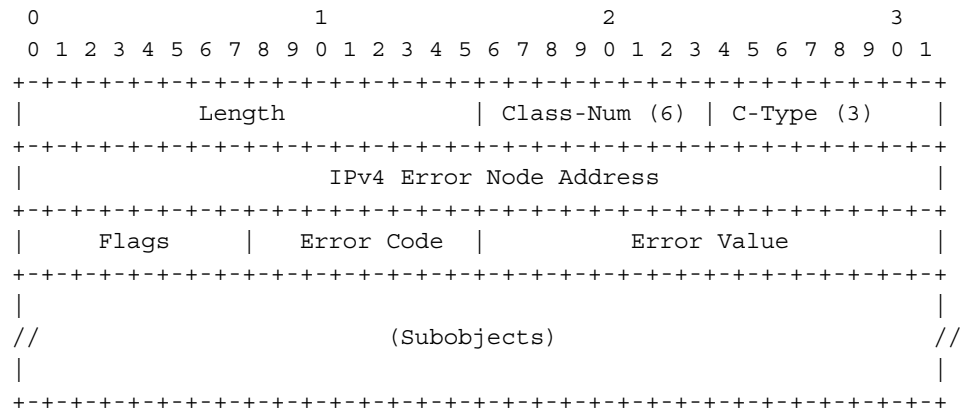
The IPv4 IF_ID_ERROR_SPEC (class = 6, C-Type =3) is defined in [RFC3473]. The IPv4 IF_ID_ERROR_SPEC MUST be supported. In E-NNI signaling, the error node address MUST be set to the SC PC ID of E-NNI (Identifier of eNNI-U or eNNI-D) that reported the error.

During graceful deletion, the path_State_Removed flag SHOULD be set, and Error Code 0 (confirmation) and Error Value 0 SHOULD be used.

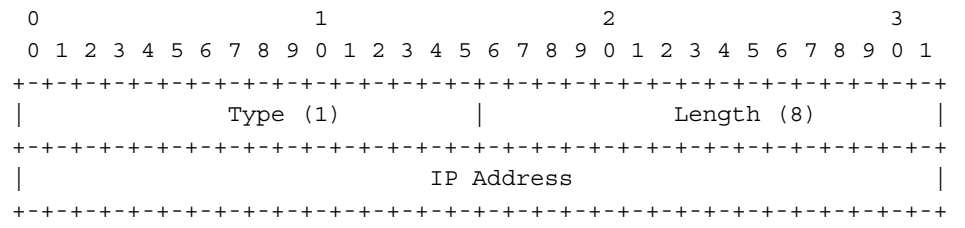
When the Error Code is non-zero, the IPv4 IF_ID_ERROR_SPEC SHOULD contain an IPv4 (Type 1) or an IPv4 IF_INDEX (Type 3) sub-object, as defined in [RFC3471]. The content of the IPv4 sub-object or IF_INDEX sub-object is significant for routing, and the identifiers used are those advertised by routing and signaled in the ERO. The IF_INDEX sub-object corresponds to a hop in the ERO. In addition to either a Type 1 or Type 3 sub-object, further objects may be included using definitions in [RFC4920].

The Type 1 object reports a whole transport node failure, and the IPv4 Address is the Node ID of the failed node. The Type 3 sub-object also carries an interface ID, specifying a link failure at that transport node.

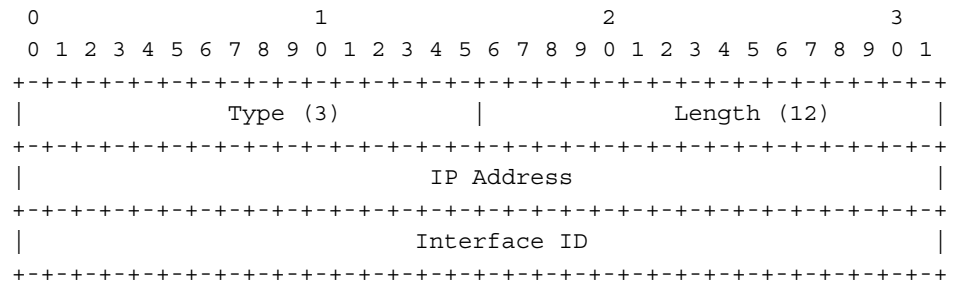
The format of the [RFC3473] IF_ID ERROR_SPEC, Type 1 sub-object is depicted below:



The format of the [RFC3473] IF_ID ERROR_SPEC, Type 1 sub-object is depicted below:

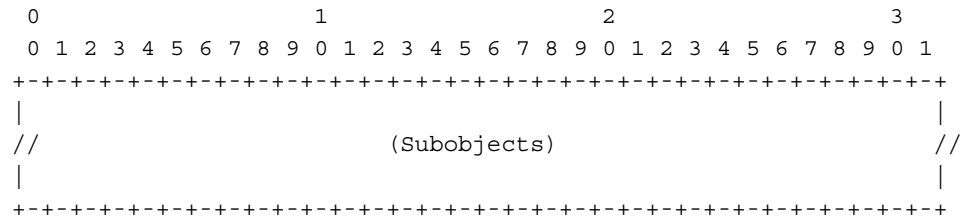


The format of the [RFC3473] IF_ID ERROR_SPEC, Type 3 sub-object is depicted below:

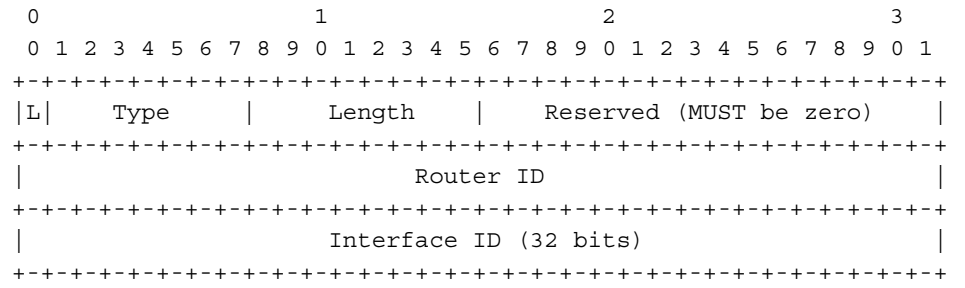


13.4 EXPLICIT_ROUTE

The Explicit Route Object Class is 20 and C_Type 1. The EXPLICIT_ROUTE object has the following format [RFC3209]:



The sub-objects in the ERO MUST be used to select transport links that the connection will travel through. The sub-objects of Type 3 (Label) defined in [RFC3473] and Type 4 (Unnumbered interface ID) defined in [RFC3477] MUST be supported in the E-NNI. As mentioned in Section 7.1, Table 1, E-NNI 2.0 maps the ASON SNPP ID definition into the tuple <RA ID, NodeID, IfIndex>. Note that the abstract RA ID is implied by NodeId, and does not appear explicitly in any signaling object. As a consequence, the ERO HOP Types 1, 2, and 32 are not required and not supported.



In the Type 4 (Unnumbered interface ID) sub-object, depicted above, the L bit SHALL NOT be set in any of the ERO sub-objects.

The Router ID MUST be set to “Node” ID and is a transport plane name. The Interface ID is the identifier assigned to the “link” by the transport “node”. It could indicate a single link or bundled link.

Both node ID and Interface ID refer to the upstream node. This choice simplifies processing of the ERO by providing a single encoding optimized for the most common situation of the processing node being at the ingress end of a link.

An example showing the specification of Explicit Route Objects in Path messages is shown in Figure 30.

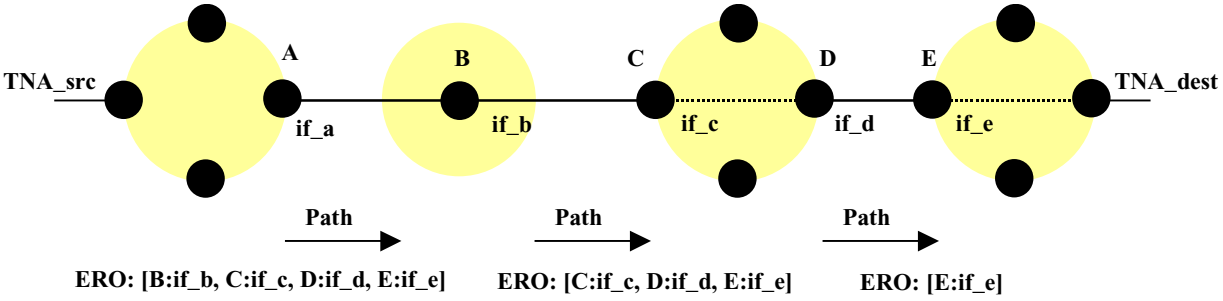


Figure 30: Example ERO Specification

13.4.1 Record Route Object

The RECORD_ROUTE object is defined in [RFC3209]. The object contains a series of variable-length data items called sub-objects. Two of the sub-objects defined by the IETF for the RRO object MAY be used in E-NNI 2.0 signaling (Table 24).

Table 24: RRO Sub-objects

Sub-object	Type	Contents
Label	0x03	Use is as defined in [RFC3209] and extended in [RFC3473].
Unnumbered IPv4	0x04	Use is as defined in [RFC 3477]. NodeID is used for RouterID.

All other RRO sub-objects MUST NOT be used in E-NNI 2.0 signaling.

The RRO object syntax is designed such that, with minor changes, the whole object can be used as input to the EXPLICIT_ROUTE object. Consequently, the RRO objects MAY identify a link by specifying only the *upstream* link end, only the *downstream* link end, or by specifying *both* link ends.

13.4.2 Generalized UNI Object

This object is used to specify the calling and called party identifiers and other call attributes. They are requested by the user through the control plane (UNI signaling) or by the management plane (configured at the network side). The attributes are used by the called party call controller and may be used by the destination network call controller for call validation. While the majority of sub-objects must be transmitted by E-NNI signaling without any alteration, some sub-objects may be translated when the E-NNI is used between carriers. As an example, the Service Level value may be translated when going from one domain to another.

The contents of a GENERALIZED_UNI object are a series of variable-length data items. For future compatibility, the Type and Sub-Type values are assigned according to the rules pertaining to RSVP objects ([RFC2205], Section 3.10), but from their own number space. The treatment of future Type and Sub-Type values is the same as specified for RSVP Class-Num and C-Type, respectively. If an error message is to be sent due to an unrecognized Type or SubType value, a node SHOULD use the error code “unknown Class-Number” or “unknown C-Type with known Class-Number” and the error value set to the Class-Number and C-Type of the GENERALIZED_UNI object.

Note that in E-NNI 2.0, the EGRESS_LABEL (Type 4 sub-object) or SPC_LABEL value is set to zero when a label value is not required. The particular egress label type provided does not specify whether signaling is active on the egress interface or not.

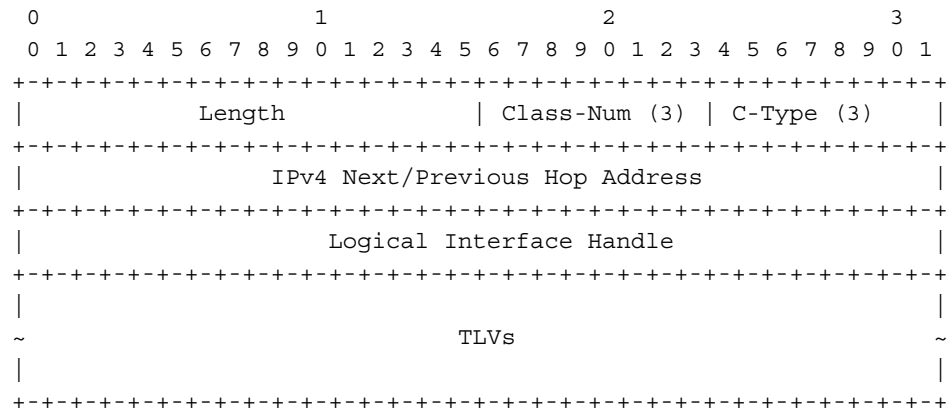
13.4.3 RESV_CONFIRM

The IPv4 RESV_CONFIRM (class = 15, C-Type =1) object is defined in [RFC2205]. IPv4 RESV_CONFIRM MUST be supported.

In E-NNI signaling, the receiver address MUST be set to the SC PC ID of the downstream E-NNI (Identifier of eNNI-D).

13.4.4 RSVP_HOP

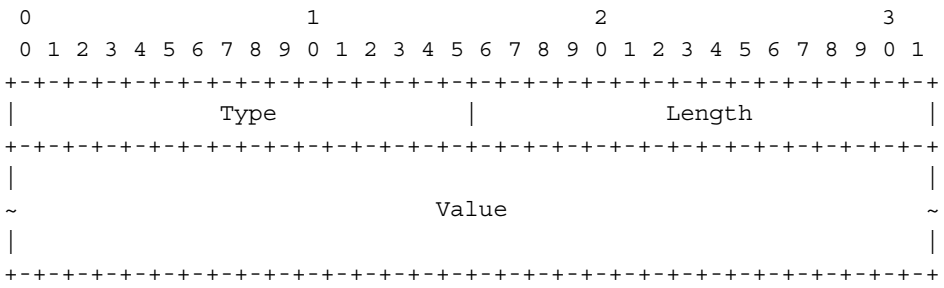
The IF_ID RSVP_HOP object is used for E-NNI interfaces. The format of this object is provided in [RFC3473]:



The IPv4 Next / Previous Hop Address field SHOULD be set to the SC PC ID (Identifier of the eNNI-U or eNNI-D) corresponding to this link. In Path messages, it contains the corresponding SC PC ID of the upstream domain and in Resv messages the corresponding SC PC ID of the downstream domain.

As described in [RFC2205], a node receiving an LIH (Logical Interface Handle) in a Path message saves its value and returns it in the HOP objects of subsequent Resv messages sent to the node that originated the LIH. Other than satisfying this requirement, *a node receiving an LIH should not expect to receive any specific value in the LIH*. Nodes MAY include a value equal to that of the interface ID (see below) or 0.

The format of TLVs is defined in [RFC3471]:

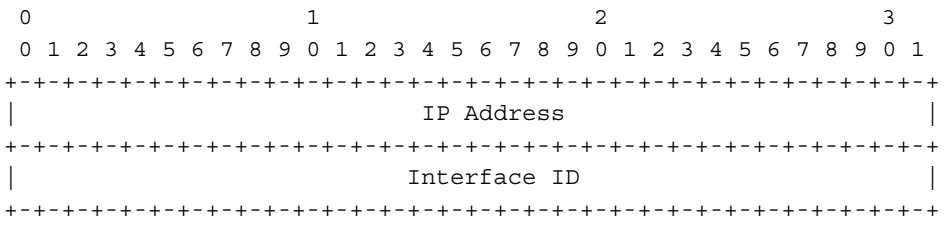


The Type 3 MUST be supported:

Type	Length	Format	Description	
3	12	See below	IF_INDEX	(Interface Index)

Use of the Type 1 and Type 2 TLVs in transport networks is not supported and is for further study.

For Type 3 the Value field has the format:



The IF_ID_RSVP_HOP object MUST be used to select the transport link where a connection's resources should be allocated. The IP Address field should be set to the node ID corresponding to this link.

- For a unidirectional LSP, a downstream data link MUST be indicated.
- For bidirectional LSPs, a common downstream and upstream data link is normally indicated. In the special case where a bidirectional LSP traverses a bundled link, it is possible to specify a downstream data link that differs from the upstream data link. When two RSVP_HOP sub-objects are required, the Type 3 sub-object MUST be used as follows:
- The first sub-object MUST represent the downstream data link
- The second sub-object MUST represent the upstream data link

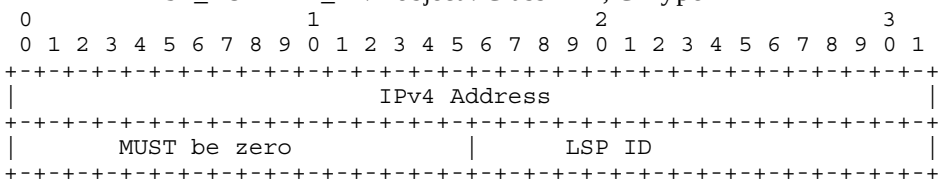
The interface ID carries the interface identifier for a single link or a bundled component link. The mapping of Interface IDs should be maintained at both the eNNI-U and eNNI-D i.e., the local and remote interface ID might not be identical.

Note: Use of node identifiers beyond IP addresses may be desirable (for example specifying nodes by a name). Support for this capability is for further study.

13.4.5 SENDER_TEMPLATE

The LSP_TUNNEL_IPv4 object (C-Type = 7) is defined in [RFC3209]. The LSP_TUNNEL_IPv4 MUST be supported, and MUST be used in SENDER_TEMPLATE and FILTER_SPEC across E-NNI interfaces. The LSP_TUNNEL_IPv4 object has the following format [RFC3209]:

- LSP_TUNNEL_IPv4 object : Class = 11, C-Type = 7



IPv4 Address: This MUST be set to the SC PC ID of the upstream E-NNI (Identifier of eNNI-U).

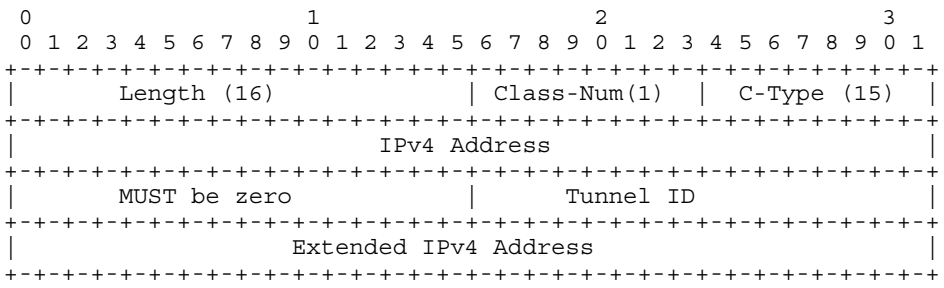
LSP ID: A 16-bit identifier used in the SENDER_TEMPLATE and the FILTER_SPEC.

The combination of the LSP_TUNNEL_IPv4_SENDER_TEMPLATE object and E-NNI_IPv4_SESSION object MUST uniquely identify a connection at a local E-NNI. In the case of connection bandwidth modification using the make-before-break procedure, the LSP_TUNNEL_IPv4_SENDER_TEMPLATE LSPID will change during the duration of the connection; all other parameters specified by these two objects remain unmodified. Otherwise, these two objects remain unmodified for the duration of the connection. An unrecognized connection ID SHOULD result in an error message with error code “Routing Problem: Invalid / Unknown Connection ID”.

13.4.6 SESSION

The E-NNI_IPv4 SESSION object (C-Type = 15) is defined in [RFC 3474]. The E-NNI_IPv4 SESSION MUST be supported across E-NNI interfaces. The SESSION object with C-Type = 15 has the following format:

- E-NNI_ IPv4_SESSION object: Class = 1, C-Type = 15



IPv4 Address: This MUST be set to the SC PC ID of the downstream E-NNI (eNNI-D).

Tunnel ID: A 16-bit identifier, assigned by the sender of the Path message. This ID remains constant during the life of a connection.

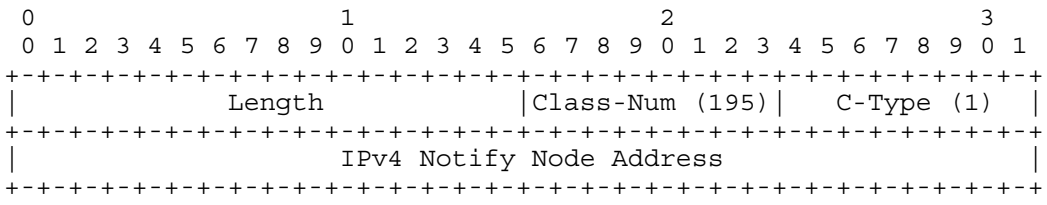
Extended IPv4 address: This MUST be set to the SC PC ID of the upstream E-NNI (Identifier of eNNI-U).

The combination of the LSP_TUNNEL_IPv4_SENDER_TEMPLATE object and E-NNI_IPv4_SESSION object MUST uniquely identify a connection at a local E-NNI. In the case of connection bandwidth modification using the make-before-break procedure, the LSP_TUNNEL_IPv4_SENDER_TEMPLATE LSPID will change during the duration of the connection; all other parameters specified by these two objects remain unmodified. Otherwise, these two objects remain unmodified for the duration of the connection.

An unrecognized connection ID SHOULD result in an error message with error code “Routing Problem: Invalid / Unknown Connection ID”.

13.5 NOTIFY_REQUEST

The Notify Node Address field contains the SC PC ID of the E-NNI node that generates the object:



13.5.1 CALL ID

The CALL_ID format is defined in [RFC3474] with further clarifications provided in [OIF-UNI-02.0-RSVP]. To guarantee uniqueness of the CALL_ID across multiple domains, the Source LSR address MUST be set to the SC PC ID of the source node generating the Call ID.

13.6 RSVP-TE Signal Flows

RSVP-TE for E-NNI follows the signal flows specified in Section 12. This section describes the RSVP-TE signal flows for connection setup, connection modification, and connection deletion.

13.6.1 Connection Setup

Figure 31 shows the setup of a connection across the E-NNI interface. Upon receiving a connection request from the network, the eNNI-U sends a Path message to the eNNI-D. The eNNI-D continues the setup request downstream. When the eNNI-D receives the setup response indication from the network, it generates a Resv message to the eNNI-U. Within the setup indication, the egress node can request an optional confirmation message. If the eNNI-U receives the confirmation from the network, it sends a ResvConf message to the eNNI-D. The eNNI-D continues to forward the confirmation to the egress.

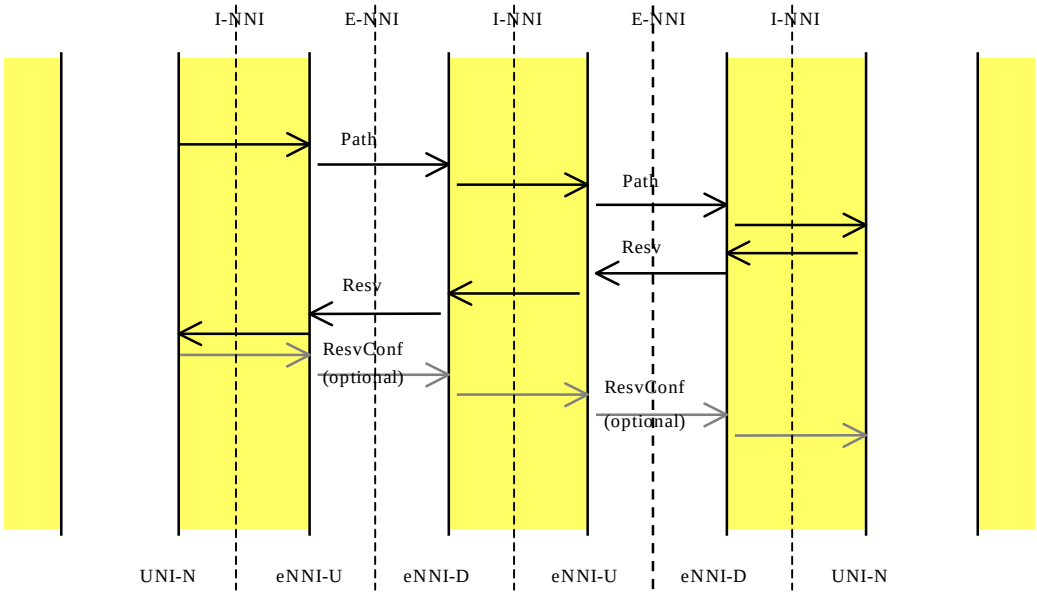


Figure 31: Basic Connection Setup Across the E-NNI

A connection setup can fail for a number of reasons including policy failure, inability to allocate resources, or destination not reachable. In the case that the eNNI-D fails the connection setup, or if the eNNI-D receives a connection setup failure indication from the network, it MUST delete its own path state and send a PathErr with the Path_State_Removed to the eNNI-U. The eNNI-U then forwards the connection failure indication towards the ingress node. This signal flow is shown in Figure 32.

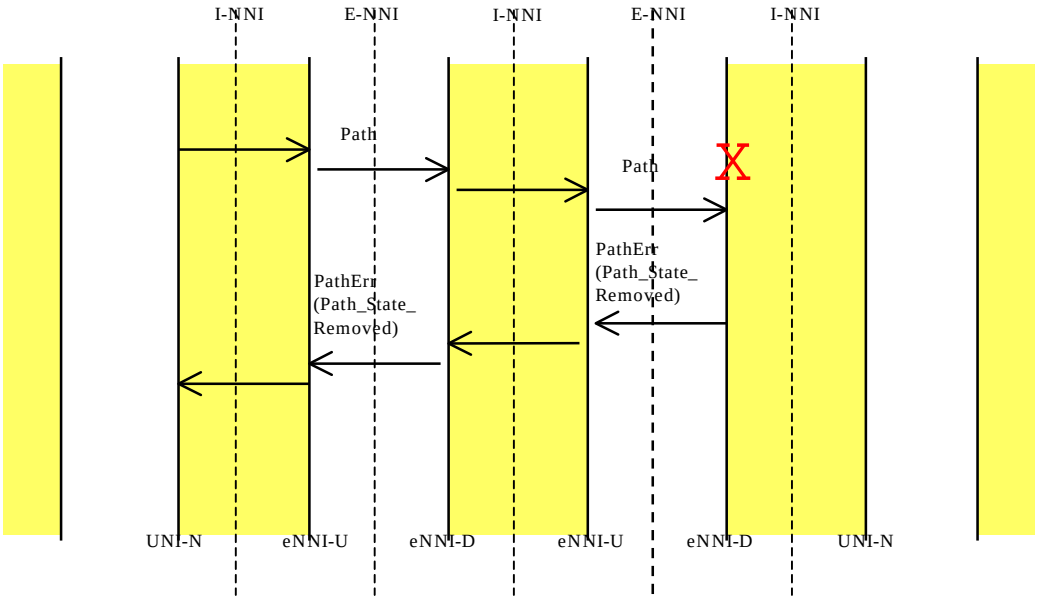


Figure 32: Connection Setup Failure

A connection setup can also fail during the indication (Resv) phase as shown in Figure 33. For instance, the label allocation can fail perhaps due to contention with another connection setup. In this case, the eNNI deletes its path state and generates PathErr with Path_State_Removed in the upstream direction and PathTear in the downstream direction.

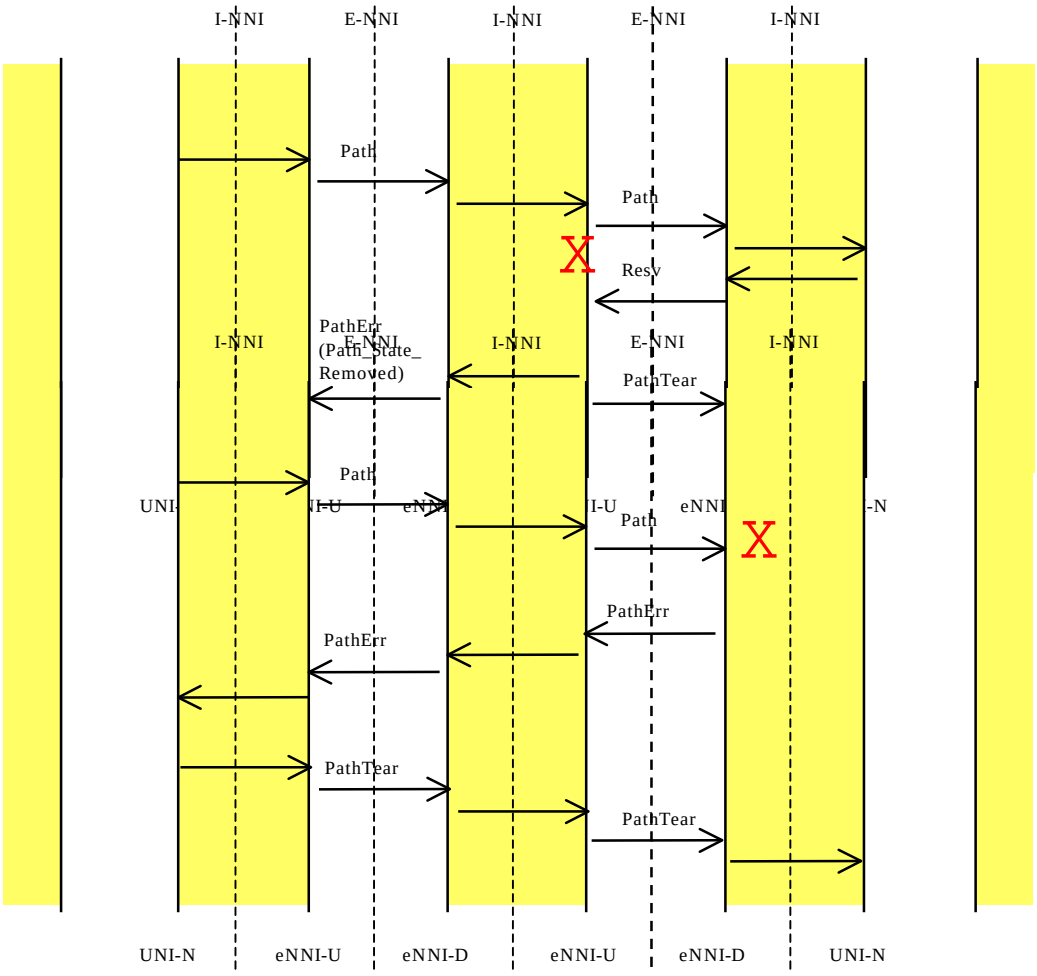


Figure 33: Connection Setup Failure during Indication

If the Path_State_Removed flag is not set in the PathErr message, then the source UNI-C or UNI-N deletes the connection explicitly. This is shown in Figure 34. When the explicit teardown request reaches the eNNI, the eNNI-U sends PathTear to the eNNI-D. A node receiving a PathTear that does not match any path state MUST acknowledge the message if the PathTear carries a MESSAGE_ID with the Ack_Desired flag set and then discard the PathTear message.

Figure 34: Connection Setup Failure without Setting the Path_State_Removed flag

13.6.2 Call Modification

A call can be modified in two ways. First, a call can be modified by adding or removing a connection to an existing call. In this case, the connection modification procedures are not used. Instead, a new connection setup request or connection release request is issued. Connections being added may follow the same route as existing connections or may be diversely routed (follow different routes). Diverse routing of connections can only be accomplished by adding connections to an existing call. The second method is to modify an existing connection within a call.

13.6.2.1 Call Modification by Adding and Removing Connections

Adding and removing connections to an existing call can be used to modify the bandwidth of a call. A failure to add or remove a connection does not impact other connections in the call. That is, the connections remain independent of each other within the call.

Figure 35 shows a successful addition of a connection to an existing call. The call is established when the first connection is established. If the source subsequently wants to modify the call by adding another connection, it will generate a new Path message with the same CALL_ID as in the existing connection. The presence of the CALL_ID in a Path message for a new connection is used to infer that a connection is being added to the specified call. CALL_ID is used to correlate the various connections at the E-NNI nodes. Upon receiving the call modification request for this scenario, the eNNI-U will generate a new Path message with the same CALL_ID. This new Path message will have a different connection identifier (TUNNEL_ID) and a new MESSAGE_ID. If the E-NNI node does not have an existing call state for the received CALL_ID, then the E-NNI node handles this as a new call setup request instead of a call modification.

Failure to add a connection to an existing call does not impact other connections because each connection has its own RSVP state.

The connection deletion message sequence is described in Section 13.6.3. Individual connections within a call can be deleted from the source, the destination, or from the network including from the E-NNI nodes. Each connection deletion is performed independently. A call without connections is not supported. Removal of the last connection results in the removal of the call state.

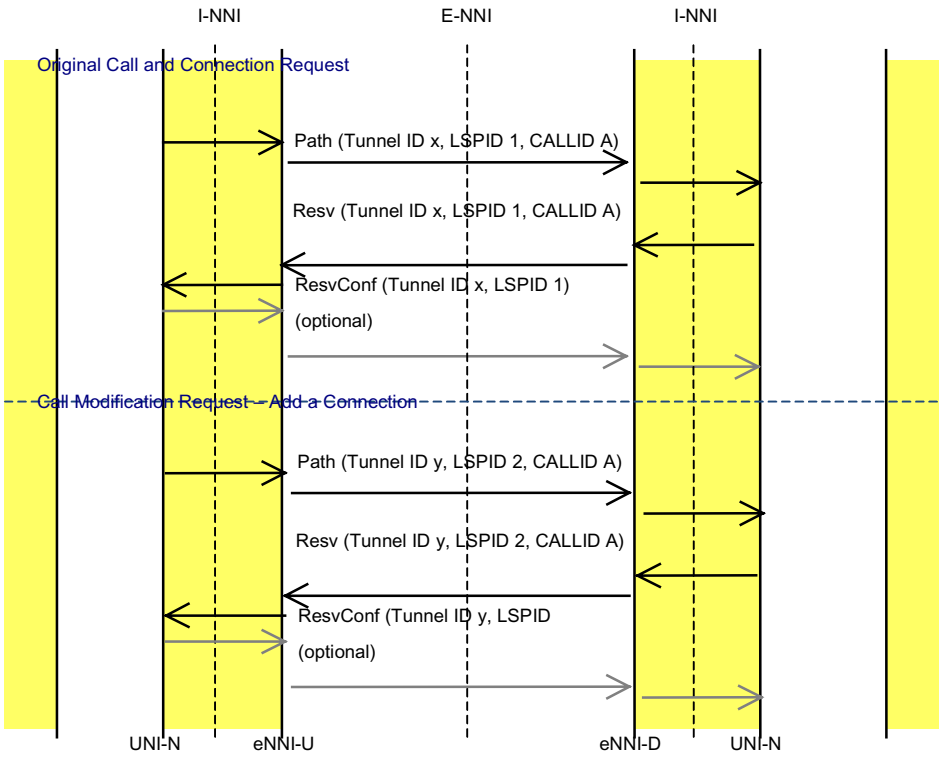


Figure 35: Successful Call Modification – Adding a Connection

13.6.2.2 Call Modification by Modification of an Existing Connection

In the second method, call modification can be supported by modifying an existing connection. E-NNI Signaling 2.0 supports the non-disruptive connection modification of the following service parameters:

- Bandwidth fields of the Ethernet SENDER_TSPEC / FLOWSPEC (CIR, CBS, EIR, or EBS) and / or the CE-VLAN ID mapping information
- Multiplier field of the SONET or OTN SENDER_TSPEC / FLOWSPEC

To support connection modification, the original connection must have been established with the Shared Explicit (SE) reservation style. This allows two or more RSVP Path states to share the same resources (such as bandwidth or CE-VLAN ID information). An E-NNI 2.0 implementation that supports non-disruptive service parameter modification SHOULD request the SE reservation style in the SESSION_ATTRIBUTE of the original Resv message. If the SE reservation style was not used in the Resv message, an E-NNI 2.0 node MUST NOT forward a non-disruptive connection modification. In this case, the E-NNI 2.0 node generates an error indication with the error code set to “Traffic Control Error: Service Unsupported”.

A non-disruptive connection modification of a service parameter is illustrated in Figure 36. This same message flow is used to support modification of the bandwidth of an existing connection or modification of the CE-VLAN ID mapping to an EVC.

The call is established when the first connection is established. The eNNI-U sends a Path message to the eNNI-D to request call and connection creation. The Path request contains a SESSION_ATTRIBUTE object with the SE Style request flag set. The CALL_ID is set to the value received in the connection setup request received from the network. If a CALL_ID was not present in the connection setup request, then the eNNI-U assigns a unique CALL_ID object and adds it to the Path message (see Section 14.2.1). The eNNI-U also assigns a locally unique SESSION object for this connection and a unique LSP ID within the SENDER_TEMPLATE scoped to the SESSION object.

If the eNNI-U receives a connection modification request for this connection, it must correlate the request to an existing connection. If the connection modification request contains a CALL_ID for which there is no call state, then the eNNI-U sends an error indication “invalid / unknown call ID”. Also, if there is no connection state, the eNNI-U sends an error indication “invalid / unknown connection ID”. Once correlated, the eNNI-U continues the connection modification over the E-NNI. The eNNI-U generates a new Path message using the same SESSION object as the original connection but with a different LSP_ID. This allows the original and new connections to share connection resources.

On receiving the connection modification indication, the eNNI-D generates a new Resv message in the reverse direction. This is achieved by generating a new Resv message for the new Path state. It contains the FILTER_SPEC and labels of the corresponding Path message only. Continuing to refresh the previous Resv message, corresponding to the state that is awaiting teardown, until the PathErr with PATH_STATE_REMOVED flag set has been received is RECOMMENDED.

The eNNI-U MUST send a new ResvConf message if it receives a modification confirmation from the network.

At this point, there should be a removal request from the network for the original connection. Upon receipt of the removal request from the network, the eNNI-U should generate a Path message for the original connection with the Delete and Reflect (D&R) bits set in the ADMIN_STATUS object. This results in the graceful removal of the RSVP Path state at the eNNI-U and eNNI-D when the eNNI-D responds with a PathErr message with the PATH_STATE_REMOVED flag set.

The deletion of the original Path state causes the removal of the corresponding Resv state after the PathErr message with the PATH_STATE_REMOVED flag set has been received. Then the Resv corresponding to the deleted state will stop being refreshed.

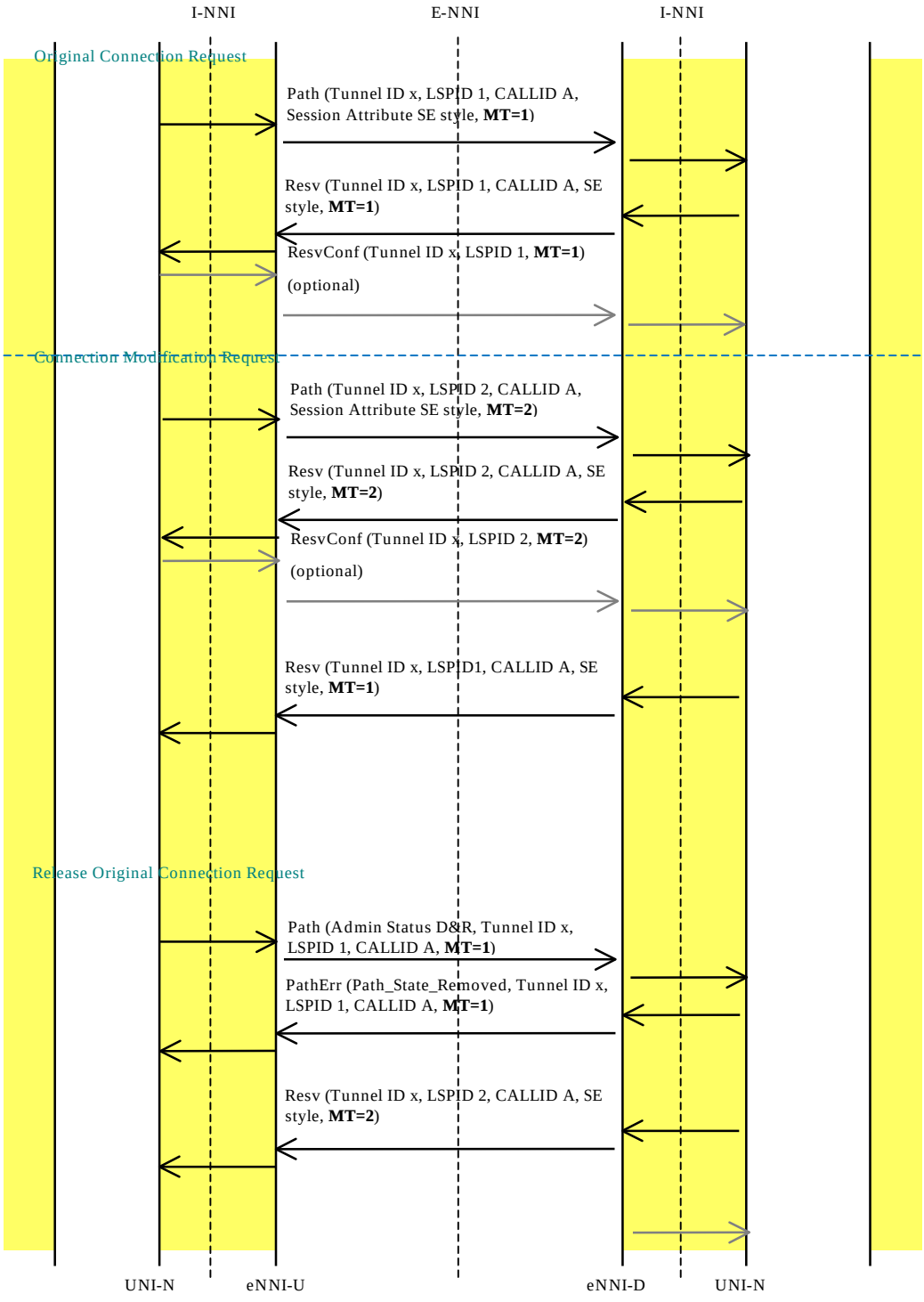


Figure 36: Successful Connection Modification

A failure to increase a connection bandwidth SHOULD result in a PathErr being sent for the Path message requesting more bandwidth. This MUST NOT impact the existing connections, other Path messages, or RSVP states. Figure 37 illustrates a failure to increase the bandwidth of a connection.

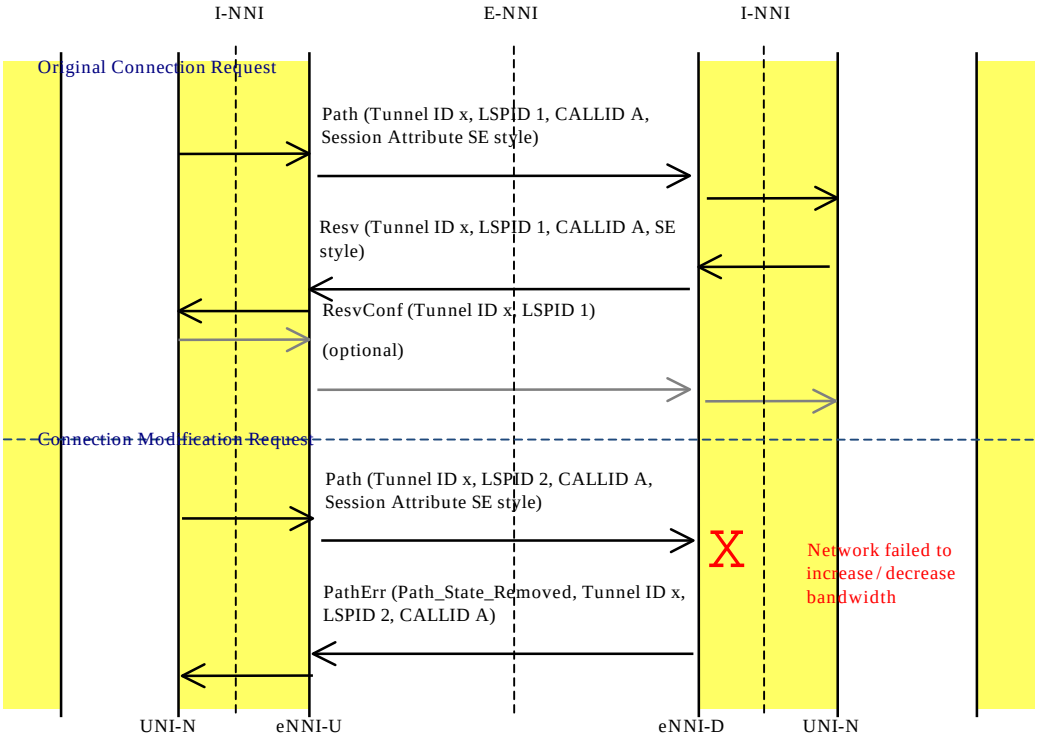


Figure 37: Connection Modification Failure

A bandwidth decrease can be achieved with an identical message flow, although the ResvConf message may not be necessary in this case. The new bandwidth becomes effective at the PathErr stage as opposed to the ResvConf message stage. A failure to decrease a connection bandwidth SHOULD result in a PathErr being sent for the Path message requesting less bandwidth. This MUST NOT impact the existing connections, other Path messages, or RSVP states.

13.6.3 Connection Deletion

13.6.3.1 Graceful Connection Deletion Initiated from the Source or Destination

RSVP allows for deletion of connections using either a single pass PathTear message, or a ResvTear and PathTear message combination. Upon receipt of the PathTear message, a node deletes the connection state and forwards the message. In optical networks, however, it is possible that the deletion of a connection (e.g., removal of the cross-connect) in a node may cause the connection to be perceived as failed in downstream nodes (e.g., loss of frame, loss of light, etc.). This may in turn lead to management alarms and perhaps the triggering of restoration / protection for the connection.

To address this issue, the graceful connection deletion procedure MUST be followed. Under this procedure, an ADMIN_STATUS object with the D-bit set MUST be sent in a Path or Resv message along the connection's path to inform all nodes enroute of the intended deletion, prior to the actual deletion of the connection. The procedure is described in [RFC3473] and shown in Figure 38 and Figure 39.

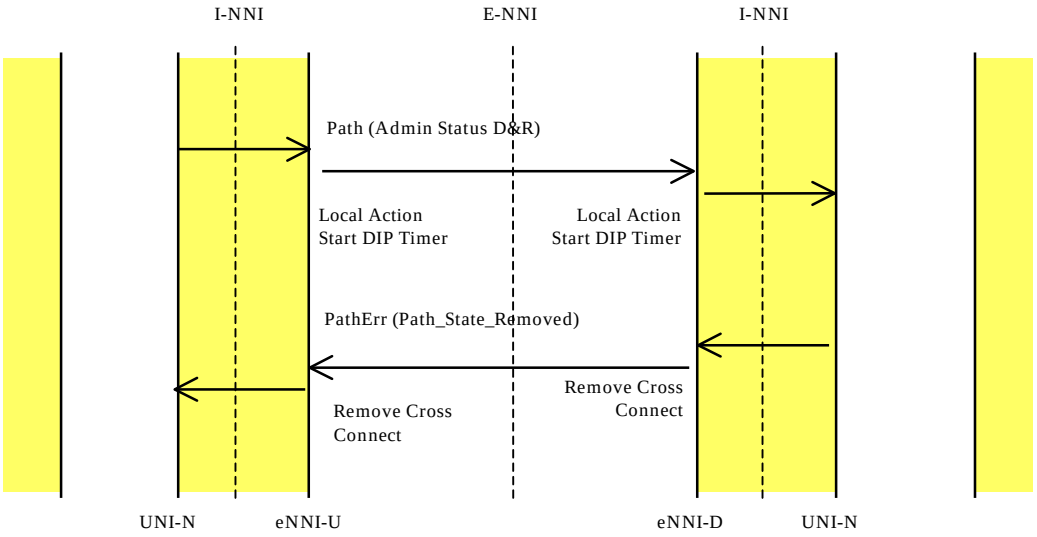


Figure 38: Connection Teardown Initiated by the Source

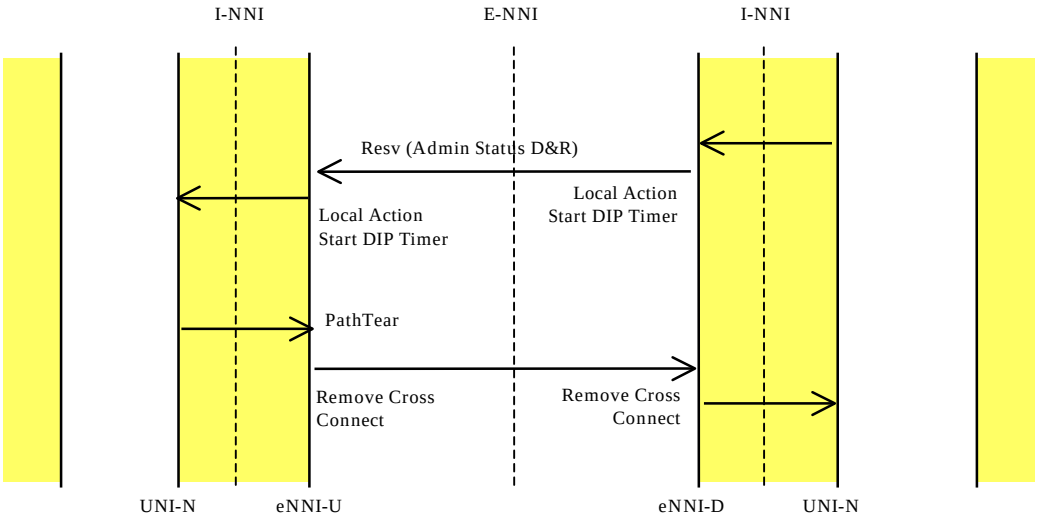


Figure 39: Connection Teardown Initiated by the Destination

13.6.3.2 Graceful Connection Deletion Initiated from the Network

A graceful deletion may also be initiated within the network. In this case, the eNNI-D may receive a connection deletion notification from the network. It is also possible that the eNNI-D can initiate a graceful deletion. In E-NNI Signaling 1.0, the network-initiated graceful deletion notification was signaled via the Path or Resv message with the A&R bits set. The signaling of graceful deletion is changed to use the Notify message in E-NNI Signaling 2.0 to align with [RFC3473].

13.6.3.2.1 Notify Message Support

The Notify message MUST be supported. In E-NNI Signaling 2.0, the Notify message is only used to signal a connection deletion initiated from an E-NNI or network node.

An eNNI-U node MUST include the NOTIFY_REQUEST object in the Path message sent to the eNNI-D. Likewise, an eNNI-D node MUST include the NOTIFY_REQUEST object in the Resv message to the eNNI-U. The Notify Node Address field of the NOTIFY_REQUEST object is set to the SC PC ID of the node generating the object.

When an E-NNI node needs to generate a Notify message, it targets the message to the SC PC ID associated with the Notify Node Address received in the incoming Path or Resv message. All session-specific objects SHALL be set to the appropriate values for the E-NNI connection segment.

An E-NNI node MUST NOT generate a Notify message to a signaling controller from which it did not receive a NOTIFY_REQUEST object. In addition, an E-NNI node SHOULD NOT generate a Notify message if it received a NOTIFY_REQUEST object but the Notify Node Address does not match its neighbor's SC PC ID.

13.6.3.2.2 Network Initiated Graceful Deletion

An E-NNI node MUST support the ability to forward a network initiated graceful deletion notification across the E-NNI interface. In addition, an E-NNI node MAY support the ability to initiate a graceful deletion notification. Network initiated graceful deletion is signaled across the E-NNI 2.0 interface using the Notify message instead of using the Path or Resv message with the ADMIN_STATUS A&R bits set.

An eNNI-D node initiates, or forwards, a graceful deletion notification by sending a Notify message to the eNNI-U. The graceful deletion Notify message contains the ADMIN_STATUS object with the D bit set. The eNNI-D also sends the graceful deletion Notify message when it receives a graceful deletion notification from the network.

In E-NNI Signaling 2.0, the graceful deletion notification SHOULD always be sent upstream to the source node. Upon receipt, the source node initiates the normal graceful deletion procedures as specified in [OIF-UNI-02.0]. The signal flow for a network initiated graceful deletion is shown in Figure 40.

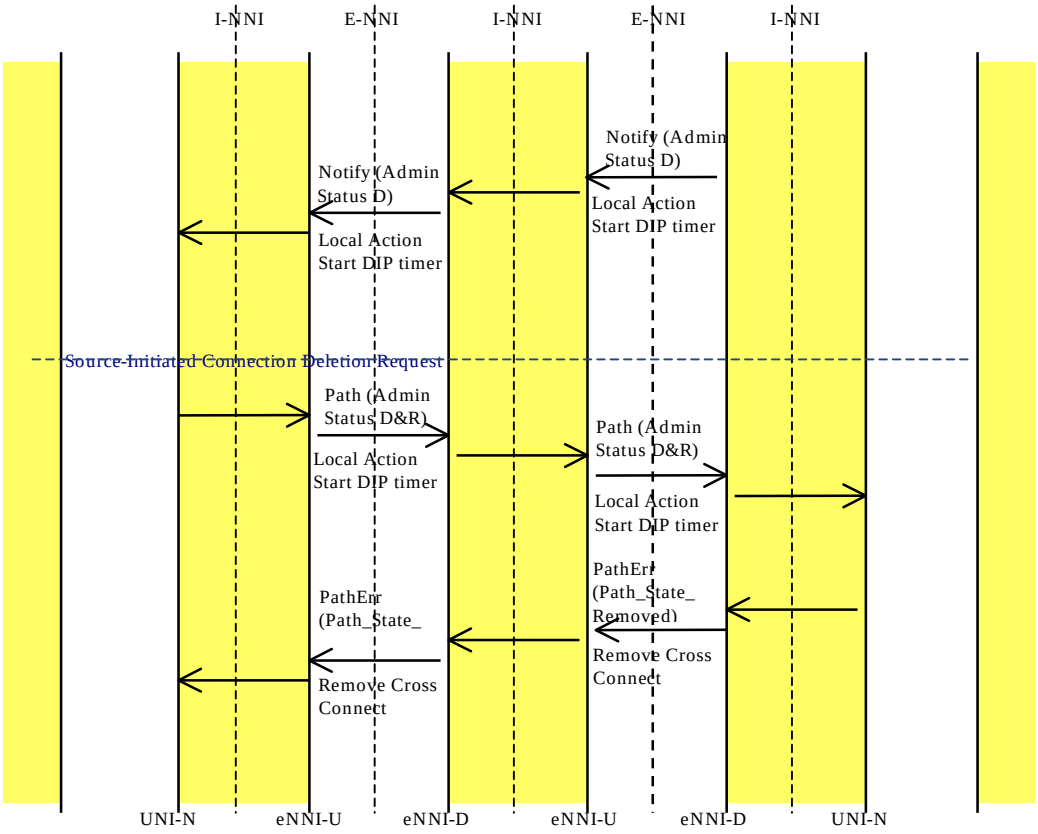


Figure 40: Connection Teardown Initiated by the eNNI-D

13.6.3.2.3 E-NNI Signaling 1.0 Compatibility

An E-NNI 2.0 node MUST NOT send the Notify message for a graceful deletion notification if its neighbor only supports E-NNI Signaling 1.0. Instead, the E-NNI 2.0 node MUST send a Path or Resv message containing the ADMIN_STATUS object with the A&R bits set to signal network graceful deletion.

An E-NNI 2.0 node determines its neighbor's E-NNI version support either by manual configuration or through an automatic discovery process. An E-NNI node MAY assume the neighbor is running E-NNI 1.0 Signaling if it does not receive a NOTIFY_REQUEST object or if the NOTIFY_REQUEST object is received but the Notify Node Address is not equal to the neighbor's SC PC ID.

E-NNI Signaling 1.0 also allows for graceful deletion notifications in the downstream direction. In this case, the eNNI-U SHOULD signal the downstream graceful deletion to an E-NNI 2.0 compliant eNNI-D by using the Notify message containing the ADMIN_STATUS object with the D bit set. Otherwise, it SHOULD send a Path message with the A&R bits set in the ADMIN_STATUS object if the eNNI-D is E-NNI 1.0 compliant.

13.6.3.3 Forced Deletion

An E-NNI node SHOULD support the ability to initiate a forced deletion of a connection. A forced deletion may be necessary to react to events such as:

- Internal network failures, which force the network to terminate connections
- When the "Deletion In Progress" timer object expires

An eNNI-U node initiates a forced deletion by deleting its RSVP states and removing the cross connect. It then sends a PathTear message downstream to the eNNI-D while at the same time signaling a forced deletion in the upstream direction. The eNNI-D, upon receipt of the PathTear message, deletes its RSVP states and removes the cross connect. The eNNI-D continues the signaling of the forced deletion in the downstream direction. This signal flow is shown in Figure 41.

Figure 42 shows a forced deletion initiated by an eNNI-D node. In this case, the eNNI-D deletes its RSVP states and removes the cross connect. The eNNI-D then signals the forced deletion by sending a PathErr message with the "Path_State_Removed" flag set to the eNNI-U and simultaneously signals a forced deletion in the downstream direction. The eNNI-U will delete its RSVP states and remove the cross connect when it receives the PathErr message. The eNNI-U will continue to propagate the forced deletion signal upstream through the network.

It is also possible that the network may generate a forced deletion signal. When an eNNI-U receives the forced deletion signal from the upstream network, it deletes the RSVP states, removes the cross connect, and signals PathTear to the eNNI-D. Likewise, if an E-NNI-D receives a forced deletion signal from the downstream network, it deletes the RSVP states, removes the cross connect, and signals PathErr with the "Path_State_Removed" flag set to the eNNI-U.

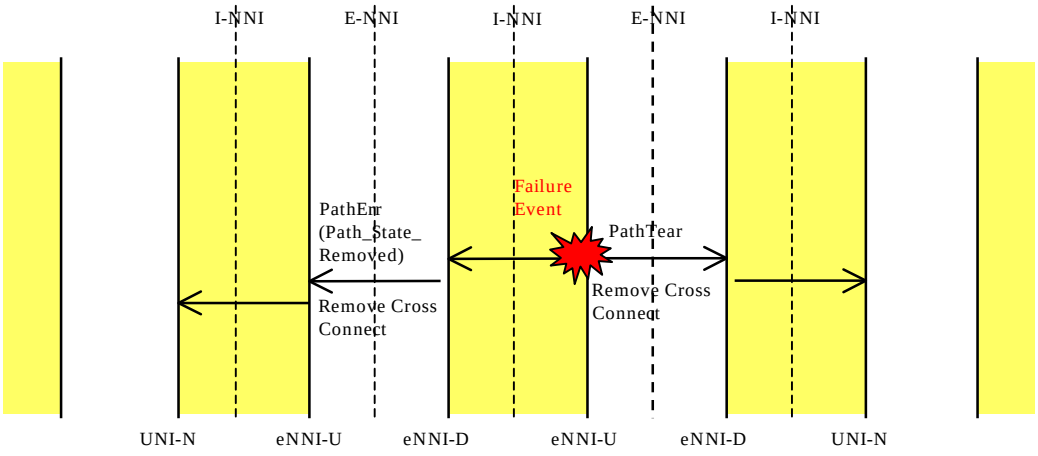


Figure 41: Forced Deletion Initiated by an eNNI-U

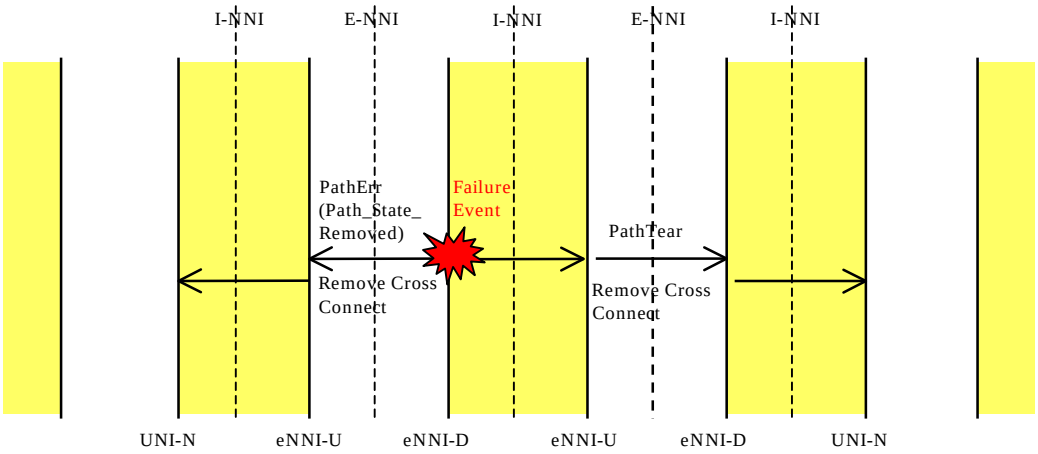


Figure 42: Forced Deletion Initiated by an eNNI-D

13.6.4 Additional RSVP-TE Messages

In addition to the signal flows described in Section 12, RSVP-TE provides the ACK message. This message is only transmitted between eNNI-U and eNNI-D and may be used:

- To obtain an acknowledgement for sent messages. The acknowledgement function can be provided either directly, using the Ack message, or indirectly (via MESSAGE_ID_ACK) when the sent message has a corresponding reply message (that is immediately generated) on a specific link (e.g., Resv / PathErr is Path's corresponding reply message). Figure 43 illustrates an additional ACK for the case of connection setup.

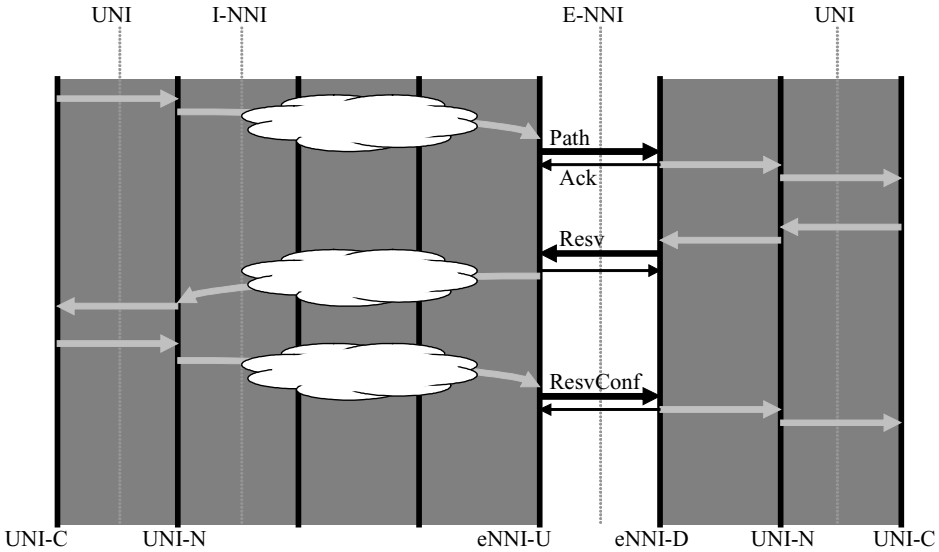


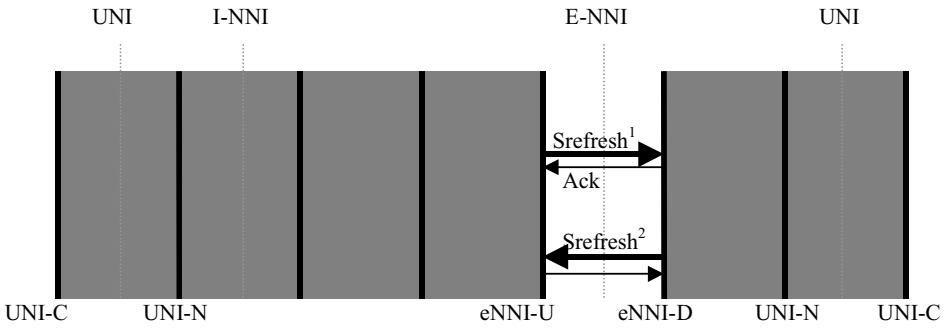
Figure 43: Basic SC Setup Using RSVP-TE

RSVP-TE also defines Hello and Srefresh messages. Both messages have local scope and are specific to the RSVP-TE protocol. The Hello message may be used:

- to ensure an RSVP session is up (using request and acknowledge objects)
- to initiate restart procedures by exchanging recovery and restart timers

The Srefresh message may be used:

- to refresh RSVP-TE state without the transmission of Path or Resv messages. This reduces the amount of information that must be transmitted and processed to maintain connection state synchronization. A Srefresh message carries a list of Message_Identifier fields corresponding to the Path and Resv trigger messages that established the state. Message_Identifier fields are carried in a MESSAGE_ID_OBJECT. Figure 44 illustrates an example of Srefresh used to refresh Path and Resv states.



Note 1: This Srefresh may be used to refresh both Path and Resv state information associated with all connections from eNNI-U to eNNI-D.
Note 2: This Srefresh may be used to refresh both Path and Resv state information associated with all connections from eNNI-D to eNNI-U.

Figure 44: Basic Srefresh Signaling

13.7 RSVP-TE Control Plane Failures

13.7.1 RSVP-TE Signaling Channel Failure

As described in Section 12.2.4, the failure of a signaling channel or control protocol entities MUST NOT result in the deletion of previously established connections. The handling of control state failure (without loss of the forwarding state) is described in [RFC3473] through the support of the RESTART_CAP object[6], which requires the use of Hello messages. Here, in particular, a node MUST support the fault handling procedure described in Section 9 of [RFC3473].

In addition to the behaviors described in Section 12.2.4, RSVP-TE requires an exchange of messages to synchronize the states of established connections. During a signaling channel failure, a self-refresh procedure is executed to prevent state information from expiring. After recovery from the failure, the neighboring control entities initiate an exchange of Hello messages. The Hello messages are used to trigger the process of synchronizing (or recovering) the states of

[6] To support the requirement that a control plane failure does not affect established connections, the Restart Time used in the RESTART_CAP object of the Hello message MUST be set to 0xffffffff. Note that local policy or configuration rules that are set based on management input may override the values specified by the Restart Time.

established connections. This ensures that the states of established connections remain consistent. The following local behaviors apply to nodes impacted by the signaling channel failure:

- A control plane node detecting a signaling channel failure should inform the management system of the failure. The default (control plane) behavior is to enter self-refresh of the call/connection states. The management system may give the control plane specific instructions to override the default behavior, for example, to release certain connections. As an example, possible management system instructions may be to remain in self-refresh mode, or to release certain connections.

- A control plane node (NCC or CC) detecting that one (or more) connections cannot be synchronized with its neighbor (e.g., due to different states for the call or connection) should inform the management system. The default behavior of the control plane should be to retain the connection unless explicitly instructed to release the connection by an external entity. As an example, possible management system instructions may be to delete the connection. Specifics of the interactions between the control plane and management plane are outside the scope of this document.

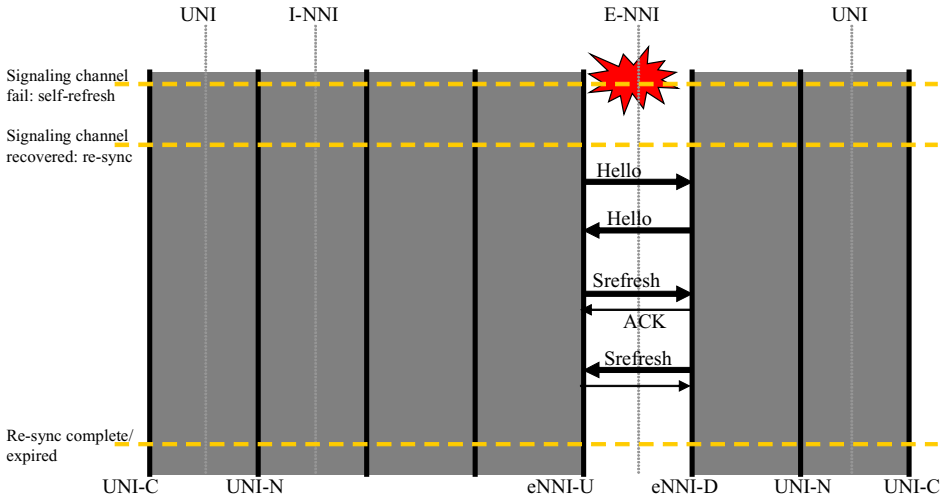


Figure 45: Recovery from Signaling Channel Failure

To ensure RSVP Hellos are supported, a node MUST NOT accept any call request unless a successful control adjacency has been established. A PATH message received from a node that has not sent any Hello message yet MAY be used to trigger the Hello procedure. A node SHOULD respond to unexpected or erroneous Hello messages by setting the Dst_Instance to 0 in the Hello Request or HelloAck object, which indicates that the received message is not accepted.

13.7.2 RSVP-TE Control Plane Failure

In addition to the behaviors described in Section 12.2.5, RSVP-TE requires a message exchange to synchronize the states of established connections. During a control plane node (CC) failure, a self-refresh procedure is executed to prevent state information from expiring. After recovery from the failure, the recovered node (CC) must attempt to restore the state information of established connections from its local persistent storage [G.7713.2]. Subsequent to this, the neighboring control entities initiate exchanges of Hello messages. The Hello messages are used to trigger

synchronizing (or recovering) the states of established connections. This ensures that the states of established connections remain consistent. Thus the following local behaviors can be envisioned for handling control plane node (CC) failure:

- A control plane node (NCC/CC) must provide for persistent storage of call and connection state information. This allows each control plane node (NCC/CC) to recover the states of calls or connections after recovery from a signaling controller entity failure or reboot (or loss of local state in memory). Note that although the restart mechanism allows neighboring control plane nodes (NCCs/CCs) to recover (and thus infer) the states of calls or connections automatically, this mechanism can be used to verify neighbors' states while the persistent storage provides the local recovery of lost state. In this case, per [RFC3473], if during the Hello synchronization the restarting node (NCC/CC) determines that a neighbor does not support state recovery, and the restarting node (NCC/CC) maintains its state on a per neighbor basis, the restarting node (NCC/CC) should immediately consider the Recovery to be complete.

- A control plane node (NCC/CC) detecting that one (or more) connections cannot be synchronized with its neighbor (e.g., due to different states for the call or connection) should inform the management system. The default behavior of the control plane should be to retain the connection unless explicitly instructed to release the connection by an external entity. The management system may give the control plane further instructions on how to handle the non-synchronized connection. As an example, possible management system instructions may be to delete the connection. Specifics of the interactions between the control plane and management plane are outside the scope of this document.

- A control plane node (NCC/CC), after recovering from node failure, may not be able to recover neighbor forwarding adjacency information from its local persistent storage and thus may lose information on forwarding adjacencies. In this case the control plane node (NCC/CC) should query an external controller (e.g., the management system) for information to recover the forwarding adjacency information. Specifics of the interactions between the control plane and management plane are outside the scope of this document.

13.8 Security Note

Note that using the security attribute defined in [G.7713] or the RSVP INTEGRITY object described in [RFC2747] (which is updated by [RFC3097]) for securing the OIF Control Plane is NOT RECOMMENDED because the [G.7713] security attribute is not specifically defined, and the RSVP INTEGRITY object :

- only covers one protocol. A single security solution for all Control Plane protocols is desired.

- does not provide the required confidentiality service or any automated method for exchanging and updating keys.

- specifies MD5 as its only security transform, and MD5, as a hash function, is now considered a weak mechanism.

The rationale given in [RFC2747] for rejecting IPsec does not apply to RSVP-TE as used in the OIF Control Plane.

14 Compatibility with UNI and E-NNI

Any unknown protocol objects shall be handled according to the methods of their specific protocols.

1) In RSVP-TE, (per [RFC2205]) the class number range has three categories for unknown class
    a. 0-127: the message should be rejected with "Unknown Object Class" error
    b. 128-191: the message should not be rejected but the unknown class should be dropped
    c. 192-255: the message should not be rejected and the unknown class should be forwarded without examination and modification

14.1 E-NNI 2.0 Compatibility with UNI

E-NNI 2.0 signaling MUST be compatible with call/connection control originating from or destined to a UNI 2.0 compliant interface.

As noted in Section 4.2, the ASON architecture supports both single and multi-layer scenarios. Multilayer support refers to the ability to handle the adaptation of one CI into another CI. This Implementation Agreement is limited to operations on a single layer network (see [G.800]) at a time.

The scope of this IA with respect to Ethernet is to establish Ethernet UNI services over a network server layer connection. There is no support for Ethernet bearer interfaces. In E-NNI 2.0, Ethernet signaling is limited to providing a forwarding function for messages and objects in client layers (Ethernet and VCAT) to support UNI 2.0.

14.2 E-NNI 2.0 Compatibility with E-NNI 1.0

The E-NNI 2.0 supports all RSVP messages and procedures described in E-NNI 1.0. These messages and procedures are included in this specification for completeness. E-NNI 2.0 also supports the following features described in [OIF-UNI-02.0]:

1. Call Control
2. Sub STS-1 Rate Connections
3. Transport of Ethernet Services
4. Transport of OTN Interfaces
5. Enhanced Security
6. Call Modification

The E-NNI 2.0 specification also modifies certain procedures for existing features:

1. Network Initiated Graceful Deletion
2. Address Separation of Node Id and SC PC ID

3. Hello Procedures Clarification

4. Note on RSVP_HOP

Backwards compatibility considerations for each feature are covered in this section.

An E-NNI implementation MAY allow for manual configuration of its neighbors' version but this is not required.

14.2.1 Call Control

The Call Control feature uses the CALL_ID RSVP object in the Path message (Section 13.2.1), PathErr message (Section 13.2.6), PathTear message (Section 13.2.5), and Resv message (Section 13.2.3). The CALL_ID format is defined in [RFC3474] with further clarifications provided in [OIF-UNI-02.0-RSVP].

The CALL_ID object is mandatory in E-NNI 2.0 and is further discussed in Section 13.5.1. The presence of the CALL_ID object in messages sent to E-NNI 1.0 implementations should not cause any problems because the CALL_ID is supported, though optional, in E-NNI 1.0. Additionally, the Class-Number for the CALL_ID is of the form 11bbbbbb, and, according to [RFC2205], an implementation that receives an unknown object of this type should ignore and forward the object unmodified. If an E-NNI 2.0 implementation receives a Path message without a CALL_ID, that implementation SHOULD insert a CALL_ID, and use that CALL_ID in all messages related to the call.

14.2.2 Sub STS-1 Rate Connections

Sub STS-1 rate connections are requested using the SONET/SDH SENDER_TSPEC [RFC4606] format that is identical to the format used in E-NNI 1.0. This feature is optional under E-NNI 2.0, and nodes must be able to switch at the sub STS-1 rate level to provide this service.

14.2.3 Transport of Ethernet Services

E-NNI 2.0 does not support signaling over Ethernet bearer interfaces, but does support Ethernet services mapped over provisioned or signaled server layer connections. Ethernet Services require a new type of SENDER_TSPEC/FLOWSPEC described in [ETH_PARAM] and two new label formats for Ethernet Private Line and Ethernet Virtual Private Line as described in [OIF-UNI-02.0]. To be able to request this service, all E-NNI nodes involved in signaling at the Ethernet layer MUST support E-NNI 2.0.

As noted in Section 4.2, the scope of this IA with respect to Ethernet is to establish Ethernet UNI services over a network server layer connection. There is no support for Ethernet bearer interfaces.

14.2.4 Transport of OTN (G.709) Interfaces

OTN Interfaces require a new type of SENDER_TSPEC/FLOWSPEC and a new label format described in [RFC4328]. To be able to request this service, all E-NNI nodes on the path MUST support E-NNI 2.0 and switching at the [G.709] OTN (ODUk) layer.

14.2.5 Enhanced Security

Enhanced Security in UNI 2.0 is completely compatible with the security methods specified in this Implementation Agreement, and a single implementation of security can be used for both UNI 2.0 and E-NNI 2.0 signaling.

14.2.6 Call Modification

To be able to modify a call, all E-NNI nodes MUST support E-NNI 2.0 and a common mechanism for the optional call modification extensions.

The first call modification mechanism is adding or removing connections to an existing call. This mechanism only requires support for the CALL_ID object and the ability to add or remove connections associated with this CALL_ID. See Section 14.2.1 for backwards compatibility discussion of the CALL_ID.

The second call modification mechanism is modifying the bandwidth of an existing connection. This mechanism relies on the RSVP make-before-break (MBB) procedure. For connections that support connection modification, the Shared-Explicit (SE) reservation style is required in the SESSION_ATTRIBUTE of the Path message. If the Shared-Explicit style is selected in the Resv message, then the MBB procedure can be used.

The procedure is described in [OIF-UNI-02.0]. However, contrary to [OIF-UNI-02.0], which supports either one or two FILTER_SPEC objects in Resv messages, this specification recommends using two Resv messages, each carrying a single FILTER_SPEC. The Resv message sent for the new Path state contains the FILTER_SPEC and labels of the corresponding Path message only. It is RECOMMENDED to keep refreshing the previous Resv message corresponding to the state that is awaiting teardown until the PathErr with Path_State_Removed flag set has been received.

14.2.7 Network Initiated Graceful Deletion

The UNI 2.0 specification uses the Notify message as opposed to the ADMIN_STATUS object A+R bits in the Path/Resv message to initiate graceful deletion from the network. Network deletion includes deletion initiated by the source and destination UNI-Ns for SC services.

The procedures for graceful deletion initiated by the source or destination UNI-C for SC services, and by the source or destination UNI-Ns for SPC services, remain the same as in E-NNI 1.0. Compatibility with E-NNI for graceful deletion is covered in Section 13.6.3.

14.2.8 Address Separation of Node Id, SC PC ID, and SC PC SCN Address

The E-NNI 2.0 specification has introduced separation of the Node Id, SC PC ID, and SC PC SCN Address identifiers. In E-NNI 1.0, all these identifiers were combined to form the Node Id. For backwards compatibility, it is sufficient for an E-NNI 2.0 implementation to use the same value for Node Id, SC PC ID, and SC PC SCN address. Identifier separation is further described in Section 9.1 of [OIF-UNI-02.0].

In E-NNI 2.0 the SC PC ID MUST be used to provide the CALL ID (see Section 13.5.1). This is a change with respect to the E-NNI 1.0 and UNI specifications, which are both silent on the CALL ID name space to be used.

14.2.9 Hello Procedures Clarification

The E-NNI 2.0 specification states that Hello messages MUST be exchanged prior to accepting new Path messages. Interworking with E-NNI 1.0 requires one of the following:

- The E-NNI 1.0 implementation supports not processing RSVP-TE messages until after a Hello message exchange or;
- The E-NNI 2.0 implementation allows for configuration of the Hello Message behavior to support E-NNI 1.0 implementations.

The E-NNI 2.0 specification mandates the use of the 0xFFFFFFFF value for the RESTART_TIME of the Hello Message RESTART_CAP object and a non-zero value for the RECOVERY_TIME. When interworking with E-NNI 1.0, an E-NNI 2.0 implementation can ignore the RESTART_TIME in the RESTART_CAP object value if it differs from 0xFFFFFFFF. If the RECOVERY_TIME advertised by the E-NNI 1.0 node is 0 when the Hello adjacency is recovered, then the E-NNI 2.0 node can delete the connections to and from E-NNI 1.0.

The E-NNI 2.0 specification is now in line with [RFC3209] and does not allow MESSAGE_ID_ACK or MESSAGE_ID_NACK to be carried in the Hello Message. Should a Hello Message be received containing an embedded MESSAGE_ID_ACK, the implementation should accept that MESSAGE_ID_ACK. An embedded MESSAGE_ID_NACK should be ignored.

14.3 Note on RSVP_HOP

Note that RSVP_HOP types 4 and 5 SHOULD NOT be generated, but MUST be supported if received to support E-NNI 1.0 backward compatibility.

15 References

Note that in many cases references are self-referential. Instead of "... G.8080 [G.8080]...", the text will state "... [G.8080]..."

15.1 ITU-T

– [G.707] ITU-T Rec. G.707 (2003), *Network Node Interface for the Synchronous Digital Hierarchy (SDH)*
– [G.709] ITU-T Rec. G.709 (2003), *Interfaces for the Optical Transport Network (OTN)*
– [G.7712] ITU-T Rec. G.7712 / Y.1703 (2003), *Architecture And Specification Of Data Communication Network*
– [G.7713] ITU-T Rec. G.7713 / Y.1704 (2001), *Distributed Connection Management (DCM)*, Amendment 1 (2005)
– [G.7713.2] ITU-T Rec. G.7713.2 (2003), *DCM Signalling Mechanism Using GMPLS RSVP-TE (DCM GMPLS RSVP-TE)*
– [G.7718] ITU-T Rec. G.7718 / Y.1709 (2005), *Framework for ASON management*
– [G.800] ITU-T Rec. G.800 (2007), *Unified Functional Architecture of Transport Networks*
– [G.805] ITU-T Rec. G.805 (2000), *Generic Functional Architecture of Transport Networks*
– [G.8080] ITU-T Rec. G.8080 / Y.1304 (2006), *Architecture of the Automatic Switched Optical Network (ASON)*Amendment 1, 2008

15.2 OIF

– [ETH_PARAM] *OIF contribution oif2007.291.00, D. Papadimitriou, "MEF Ethernet Traffic Parameters"*, containing text of IETF draft draft-ietf-ccamp-ethernet-traffic-parameters-02

– [OIF-E-NNI-Sig-01.0] *OIF Implementation Agreement* OIF-E-NNI-Sig-01.0 - *Intra-Carrier E-NNI Signaling Specification*, February 2004, http://www.oiforum.com/public/documents/OIF-E-NNI-Sig-01.0-rev1.pdf
– [OIF-UNI1-R2] *OIF Implementation Agreement OIF-UNI-01.0-R2-Common User Network Interface (UNI) 1.0 Signaling Specification Release 2: Common Part*, February 2004, http://www.oiforum.com/public/documents/OIF-UNI-01.0-R2-Common.pdf
– [OIF-UNI-02.0] *OIF Implementation Agreement OIF-UNI-02.0-Common - User Network Interface (UNI) 2.0 Signaling Specification: Common Part*, February 2008, http://www.oiforum.com/public/documents/OIF-UNI-02.0-Common.pdf
– [OIF-UNI-02.0-RSVP] *OIF Implementation Agreement OIF-UNI-02.0-RSVP*, "RSVP Extensions for User Network Interface (UNI) 2.0 Signaling", February 2008, http://www.oiforum.com/public/documents/OIF-UNI-02.0-RSVP.pdf
– [OIF-SEC] OIF-SEP-01.0, *Security Extension for UNI and NNI*, May 2003, http://www.oiforum.com/public/documents/Security-IA.pdf.
– [SecAdd] OIF-SEP-02.1 *Addendum to the Security Extension for UNI and NNI*, March 2006, http://www.oiforum.com/public/documents/OIF-SEP-02_1.pdf
– [SysLog] OIF Implementation Agreement *OIF Control Plane Logging and Auditing with Syslog*, OIF-SLG-01.0, November 2007, http://www.oiforum.com/public/documents/OIF-SLG-01.0.pdf.

15.3 IETF

– [RFC2119] IETF RFC 2119, *Key words for use in RFCs to Indicate Requirement Levels*
– [RFC2205] IETF RFC 2205, *Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification*
– [RFC2747] IETF RFC 2747, *RSVP Cryptographic Authentication*
– [RFC2961] IETF RFC 2961, *RSVP Refresh Overhead Reduction Extensions*
– [RFC3097] IETF RFC 3097, *RSVP Cryptographic Authentication – Updated Message Type Value*
– [RFC3209] IETF RFC 3209, *RSVP-TE: Extensions to RSVP for LSP Tunnels*
– [RFC3471] IETF RFC 3471, *Generalized MPLS - Signaling Functional Description*
– [RFC3473] IETF RFC 3473, *Generalized MPLS Signaling - RSVP-TE Extensions*
– [RFC3474] IETF RFC 3474, *Documentation of IANA Assignments for GMPLS RSVP-TE Usage and Extensions for ASON*
– [RFC3476] IETF RFC 3476, *Documentation of IANA Assignments for LDP, RSVP, and RSVP-TE Extensions for Optical UNI Signaling*
– [RFC3477] IETF RFC 3477, *Signalling Unnumbered Links in RSVP-TE*
– [RFC4201] IETF RFC 4201, *Link Bundling in MPLS Traffic Engineering (TE)*
– [RFC4328] IETF RFC 4328, *GMPLS Signaling Extensions for G.709 Optical Transport Networks Control*
– [RFC4606] IETF RFC 4606, *GMPLS Extensions for SONET & SDH Control*
– [RFC4920] IETF RFC 4920, *Crankback Signaling Extensions for MPLS and GMPLS RSVP-TE*
– [RFC4974] IETF RFC 4974, *Generalized MPLS (GMPLS) RSVP-TE Signaling Extensions*

15.4 T1X1.5

– [T1.105] ANSI T1.105 (1995), *Synchronous Optical Network (SONET) – Basic Description including Multiplex Structure, Rates and Formats*

16 Appendix A: Detailed Example of Message Object Contents

Figure 46 provides an example scenario for call setup and call teardown from the originating side, and call teardown from the destination side. There are two E-NNI interfaces shown, one between

Node A and Node B, and one between Node B and Node C. Node B may represent a domain with an abstract node representation, but no I-NNI signaling is described.

Note that this is an example scenario only, and is provided for information.

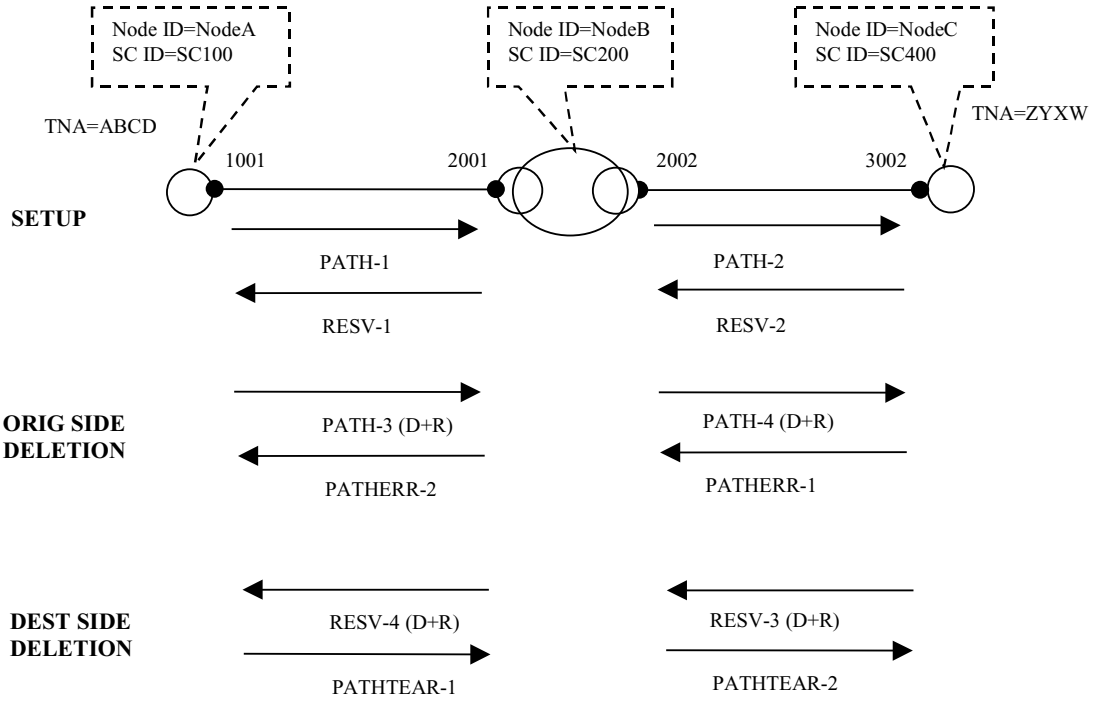


Figure 46: Call setup and teardown example

In the tables that follow, fields shaded in gray show fields whose values have not changed between the initial message (in column 1) and the subsequent message (in column 2).

PATH-1	**PATH-2**
MESSAGE_ID, TIME_VALUES – protocol specific	MESSAGE_ID, TIME_VALUES – protocol specific
SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)	SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)
Destination address: SC200	Destination address: SC300
Tunnel ID: 10001	Tunnel ID: 20001
Extended Tunnel ID: SC100	Extended Tunnel ID: SC200
RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)	RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)
Address: SC100	Address: SC200
Logical interface: 1001	Logical interface: 2002
Interface-Index TLV	Interface-Index TLV
Type = 3	Type = 3
IPv4 Address: NodeA	IPv4 Address: NodeB
Interface ID: 1001	Interface ID: 2002
GENERALIZED_LABEL_REQUEST Class – 4; C-type = 4 (Gen. Label Request)	GENERALIZED_LABEL_REQUEST Class – 4; C-type = 4 (Gen. Label Request)
LSP Encoding Type: SONET/SDH	LSP Encoding Type: SONET/SDH
Switching Type: TDM	Switching Type: TDM
G-PID: 0x0 (unknown)	G-PID: 0x0 (unknown)
SENDER_TEMPLATE Class – 11; C-type = 7 (IPv4)	SENDER_TEMPLATE Class – 11; C-type = 7 (IPv4)
Sender IPv4 address: SC100	Sender IPv4 address: SC200
Sender LSP ID: LSP01	Sender LSP ID: LSP01
SENDER_TSPEC Class – 12; C-type = 4 (SONET/SDH)	SENDER_TSPEC Class – 12; C-type = 4 (SONET/SDH)
Signal Type: 6 – STS3c/VC4	Signal Type: 6 – STS3c/VC4
RCC=NCC=NVC=0; MT = 1; T = 0; P = 0	RCC=NCC=NVC=0; MT = 1; T = 0; P = 0
ERO Class – 20; C-type = 1	ERO Class – 20; C-type = 1
TLV	TLV
Type=4 (unnumbered IF ID)	Type=4 (unnumbered IF ID)
Router ID = NodeB	Router ID = NodeC
Interface ID = 2002	
TLV	
Type=4 (unnumbered IF ID)	
Router ID = NodeC	
UPSTREAM_LABEL Class – 35; C-type = 2 (Generalized Label)	UPSTREAM_LABEL Class – 35; C-type = 2 (Generalized Label)
Generalized Label : SUKLMx	Generalized Label : SUKLMy
NOTIFY_REQUEST Class – 195; C-type = 1 (IPv4)	NOTIFY_REQUEST Class – 195; C-type = 1 (IPv4)
Notify Node Address : SC100	Notify Node Address : SC200
GENERALIZED_UNI Class – 229; C-type = 1	GENERALIZED_UNI Class – 229; C-type = 1
Source TNA	Source TNA
Class = 1; Type = 1	Class = 1; Type = 1
IPv4 address: ABCD	IPv4 address: ABCD
Destination TNA	Destination TNA
Class = 2; Type = 1	Class = 2; Type = 1
IPv4 address: ZYXW	IPv4 address: ZYXW
CALL_ID Class – 230; C-type = 1 (Operator specific)	CALL_ID Class – 230; C-type = 1 (Operator specific)

Source LSR Addr = SC100	Source LSR Addr = SC100
Local ID: 0000 0000 0000 000A	Local ID: 0000 0000 0000 000A

RESV-2	**RESV-1**
MESSAGE_ID, TIME_VALUES – protocol specific	MESSAGE_ID, TIME_VALUES – protocol specific
SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)	SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)
Destination address: SC200	Destination address: SC300
Tunnel ID	Tunnel ID
Extended Tunnel ID: SC100	Extended Tunnel ID: SC200
RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)	RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)
Address: SC200	Address: SC300
Logical interface: 1001	Logical interface: 2002
Interface-Index TLV	Interface-Index TLV
Type = 3	Type = 3
IPv4 Address: NodeA	IPv4 Address: NodeB
Interface ID: 1001	Interface ID: 2002
LABEL Class – 16; C-type = 2 (Gen. Label Request)	LABEL Class – 16; C-type = 2 (Gen. Label Request)
Generalized Label: SUKLMx	Generalized Label: SUKLMy
STYLE Class – 8; C-type = 1	STYLE Class – 8; C-type = 1
Flags: 0x0	Flags: 0x0
STYLE : 0x00000A – Fixed Filter	STYLE : 0x00000A – Fixed Filter
FLOWSPEC Class – 9; C-type = 4 (SONET/SDH)	FLOWSPEC Class – 9; C-type = 4 (SONET/SDH)
Signal Type: 6 – STS3c/VC4	Signal Type: 6 – STS3c/VC4
RCC=NCC=NVC=0; MT = 1; T = 0; P = 0	RCC=NCC=NVC=0; MT = 1; T = 0; P = 0
FILTERSPEC Class – 10; C-type = 7 (IPv4)	FILTERSPEC Class – 10; C-type = 7 (IPv4)
Sender IPv4 address: SC100	Sender IPv4 address: SC200
Sender LSP ID: LSP01	Sender LSP ID: LSP01
NOTIFY_REQUEST Class – 195; C-type = 1 (IPv4)	NOTIFY_REQUEST Class – 195; C-type = 1 (IPv4)
Notify Node Address : SC200	Notify Node Address : SC300
CALL_ID Class – 230; C-type = 1 (Operator specific)	CALL_ID Class – 230; C-type = 1 (Operator specific)
Source LSR Addr = SC100	Source LSR Addr = SC100
Local ID:0000 0000 0000 000A	Local ID: 0000 0000 0000 000A

PATH-3	**PATH-4**
MESSAGE_ID, TIME_VALUES – protocol specific	MESSAGE_ID, TIME_VALUES – protocol specific
SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)	SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)
RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)	RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)
GENERALIZED_LABEL_REQUEST Class – 4; C-type = 4 (Gen. Label Request)	GENERALIZED_LABEL_REQUEST Class – 4; C-type = 4 (Gen. Label Request)
SENDER_TEMPLATE Class – 11; C-type = 7 (IPv4)	SENDER_TEMPLATE Class – 11; C-type = 7 (IPv4)
SENDER_TSPEC Class – 12; C-type = 4 (SONET/SDH)	SENDER_TSPEC Class – 12; C-type = 4 (SONET/SDH)
ERO Class – 20; C-type = 1	ERO Class – 20; C-type = 1
UPSTREAM_LABEL Class – 35; C-type = 2	UPSTREAM_LABEL Class – 35; C-type = 2

(Generalized Label)	(Generalized Label)
NOTIFY_REQUEST Class – 195; C-type = 1 (IPv4)	NOTIFY_REQUEST Class – 195; C-type = 1 (IPv4)
GENERALIZED_UNI Class – 229; C-type = 1	GENERALIZED_UNI Class – 229; C-type = 1
CALL_ID Class – 230; C-type = 1 (Operator specific)	CALL_ID Class – 230; C-type = 1 (Operator specific)
ADMIN_STATUS Class – 196; C-type = 1	ADMIN_STATUS Class – 196; C-type = 1
Admin Status : 0x8000001 (R=D=1)	Admin Status : 0x8000001 (R=D=1)

PATHERR-2	**PATHERR-1**
MESSAGE_ID– protocol specific	MESSAGE_ID– protocol specific
SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)	SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)
SENDER_TEMPLATE Class – 11; C-type = 7 (IPv4)	SENDER_TEMPLATE Class – 11; C-type = 7 (IPv4)
SENDER_TSPEC Class – 12; C-type = 4 (SONET/SDH)	SENDER_TSPEC Class – 12; C-type = 4 (SONET/SDH)
CALL_ID Class – 230; C-type = 1 (Operator specific)	CALL_ID Class – 230; C-type = 1 (Operator specific)
ERROR SPEC Class – 6; C-type = 3	ERROR SPEC Class – 6; C-type = 3
IPv4 Error Node Address: SC200	IPv4 Error Node Address: SC300
Flags: 0x04; Code=0; Value=0	Flags: 0x04; Code=0; Value=0

RESV-4	**RESV-3**
MESSAGE_ID, TIME_VALUES – protocol specific	MESSAGE_ID, TIME_VALUES – protocol specific
SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)	SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)
RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)	RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)
LABEL Class – 16; C-type = 2 (Gen. Label Request)	LABEL Class – 16; C-type = 2 (Gen. Label Request)
STYLE Class – 8; C-type = 1	STYLE Class – 8; C-type = 1
FLOWSPEC Class – 9; C-type = 4 (SONET/SDH)	FLOWSPEC Class – 9; C-type = 4 (SONET/SDH)
FILTERSPEC Class – 10; C-type = 7 (IPv4)	FILTERSPEC Class – 10; C-type = 7 (IPv4)
CALL_ID Class – 230; C-type = 1 (Operator specific)	CALL_ID Class – 230; C-type = 1 (Operator specific)
ADMIN_STATUS Class – 196; C-type = 1	ADMIN_STATUS Class – 196; C-type = 1
Admin Status : 0x8000001 (R=D=1)	Admin Status : 0x8000001 (R=D=1)

PATHTEAR-1	**PATHTEAR-2**
MESSAGE_ID– protocol specific	MESSAGE_ID– protocol specific
SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)	SESSION: Class – 1; C-type = 15 (ipv4 E-NNI)
RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)	RSVP_HOP Class – 3; C-type = 3 (ipv4 IF ID)
SENDER_TEMPLATE Class – 11; C-type = 7 (IPv4)	SENDER_TEMPLATE Class – 11; C-type = 7 (IPv4)
SENDER_TSPEC Class – 12; C-type = 4 (SONET/SDH)	SENDER_TSPEC Class – 12; C-type = 4 (SONET/SDH)
UPSTREAM_LABEL Class – 35; C-type = 2 (Generalized Label)	UPSTREAM_LABEL Class – 35; C-type = 2 (Generalized Label)
CALL_ID Class – 230; C-type = 1 (Operator specific)	CALL_ID Class – 230; C-type = 1 (Operator specific)

17 Appendix B: List of companies belonging to OIF when document is approved

ADVA Optical Networking
Alcatel-Lucent
Altera
AMCC
Analog Devices
Anritsu
AT&T
Avago Technologies Inc.
Avalon Microelectronics
Avanex Corporation
Bookham
Broadcom
Centellax, Inc.
China Telecom
Ciena Corporation
Cisco Systems
ClariPhy Communications
CoreOptics
Cortina Systems
Data Connection
Department of Defense
Deutsche Telekom
Discovery Semiconductors
Emcore
Ericsson
Ethos Networks
ETRI
Eudyna Devices USA Inc.
EXFO Electro-Optical Engineering
Finisar Corporation
Flextronics
Force 10 Networks
France Telecom
Fujitsu
Furukawa Electric Japan
Hitachi
Huawei Technologies
IBM Corporation
IDT
Infinera
Inphi
IP Infusion
JDSU
Juniper Networks
KDDI R&D Laboratories
Kotura, Inc.
LSI Corporation
Marben Products
Mintera
MITRE Corporation
Mitsubishi Electric Corporation
Molex
NEC
NeoPhotonics
Nokia Siemens Networks
Nortel Networks
NTT Corporation
Ofidium
Opnext
Optametra
Optoplex
Picometrix
PMC Sierra
RF Micro Devices
Sandia National Laboratories
Santur
Sierra Monolithics
Silicon Logic Engineering
Soapstone Networks
ST Microelectronics
Sumitomo Osaka Cement
Sycamore Networks
Syntune
Tektronix
Telcordia Technologies
Telecom Italia Lab
Teleoptix
Tellabs
TeraXion
Texas Instruments
Time Warner Cable
TPACK A/S
TriQuint Semiconductor
Tyco Electronics

u2t Photonics AG
Verizon
Vitesse Semiconductor
Yamaichi Electronics Ltd.
ZTE Corporation