Contribution Number: OIF2000.125.7

Working Group: Architecture, OAM&P, PLL, & Signaling Working Groups

TITLE: User Network Interface (UNI) 1.0 Signaling Specification

DATE: October, 1, 2001

Document Status:
Project Name: Multiple OIF Projects
Project Number:

For additional information contact:
The Optical Internetworking Forum, 39355 California Street,
Suite 307, Fremont, CA 94538
510-608-5990 phone ✦ info@oiforum.com

List of Contributors

Osama Aboul-Magd, Nortel
Stefan Ansorge Alcatel
K. Arvind, Tenor Networks
Krishna Bala, Tellium (Chair, Signaling WG)
Sandra Ballarte, Nortel
Ayan Banerjee, Calient
Rick Barry, Sycamore
Debashis Basak, Accelight Networks
Greg Bernstein, Ciena
Curtis Brownmiller, Worldcom
Yang Cao, Sycamore
John Drake, Calient
William Goodson, Lucent
Gert Grammel, Alcatel
Richard Graveman, Telcordia
Eric Gray, Brightwave
Riad Hartini, Caspian Networks
Eric Mannie, Ebone
Raj Jain, Nayna Networks
LiangYu Jia, ONI Systems
Jim Jones, Alcatel
Suresh Katukam, Cisco
Nooshin Komaee, Tellium
Kireeti Kompella, Juniper
Jonathan P. Lang, Calient
Monica Lazer, AT&T
Fong Liaw, Zaffire
Ling-Zhong Liu, Edgeflow
Larry McAdams, Cisco
Yassi Moghaddam, Avici
Wilson Nheu, Agilent
Dimitiri Papadimitriou, Alcatel
Dimitrios Pendarakis, Tellium
Kavi Prabhu, Tenor Networks
Bala Rajagopalan, Tellium (Editor)
Rajiv Ramaswamy, Nortel
Anil Rao, ONI Systems
Robert Rennison, Laurel Networks
Yakov Rekhter, Cisco
Debanjan Saha, Tellium
Arnold Sodder, Tenor Networks
John Strand, AT&T
George Swallow, Cisco
Ewart Tempest, Nortel
Cary Wright, Agilent
Yangguang Xu, Lucent
Yong Xue, UUNET
Tao Yang, Sycamore
Jennifer Yates, AT&T
John Z. Yu, Zaffire
Alex Zinin, Cisco
Zhensheng Zhang, Sorrento

Table of Contents

1 Document Summary

1.1 Working Group(s)

Architecture Working Group
OAM&P Working Group
Signaling Working Group

1.2 Problem Statement

The advent of the automatic switched transport network has necessitated the development of interoperable procedures for requesting and establishing dynamic connectivity (with varying levels of transparency) between clients (e.g., IP, ATM, and SONET devices) connected to the transport network. The development of such procedures requires the definition of the physical interfaces between clients and the transport network, the connectivity services offered by the transport network, the signaling protocols used to invoke the services, the mechanisms used to transport signaling messages, and the auto-discovery procedures that aid signaling. These definitions constitute the *User Network Interface* (UNI), the service control interface between the user devices and the transport network. This document describes the implementation agreement in the OIF on realizing the first version of the UNI, referred to as UNI 1.0.

.

1.3 Scope

The scope of this agreement is to define the set of services, the signaling protocols used to invoke the services, the mechanisms used to transport signaling messages and the auto-discovery procedures that aid signaling, all of which are to be implemented by client and transport network equipment vendors to support UNI 1.0. This document is scoped to allow an early implementation based on reusing existing signaling protocols and auto-discovery mechanisms, along with current and newly available technologies and capabilities in vendor equipment. It should be noted that only signaling for service invocation is within the scope of UNI 1.0. Routing, reachability and address resolution protocols are outside the scope. Also, the UNI 1.0 specification focuses on SONET/SDH connection services. This implementation agreement is not intended to restrict additions of further capabilities in future versions of the UNI.

1.4 Merits to OIF

The UNI 1.0 specification is a key step towards the implementation of an open transport network layer that allows dynamic interconnection of client layers like IP, ATM, SONET and others. This activity supports the overall mission of the OIF.

1.5 Relationship to other Standards Bodies

This document, to the maximum extent possible, utilizes standards and specifications already available from other organizations. Specifically, the SONET/SDH service definitions are based on ITU-T specifications. The signaling protocols and auto-discovery mechanisms are based on work in progress in the IETF on Generalized MPLS and Link Management Protocol (LMP) (the appropriate references are marked “Work in Progress” in Section 14). Although the IETF work is nearing completion, the IETF protocol specifications are subject to change before they become proposed standards. The current protocol adaptations for UNI 1.0 will not be affected by these changes since they fully address the functional requirements for this version of the UNI. However, since the intent of OIF is to develop UNI protocols in close alignment with the IETF work, any changes in GMPLS/LMP proposed standards that could be incorporated in UNI signaling specifications will be included in future versions.

This document also describes certain new capabilities with regard to signaling. It is expected that the OIF would liaison this information to the IETF for formalization.

1.6 Unique Viewpoint

This document is unique in that it consolidates many aspects with respect to the work being done at other standards bodies. In particular, it addresses the optical internetworking requirements for

- Services associated with data clients;
- Rapid provisioning of SONET/SDH framed circuits through optical transport networks; and
- Client-to-transport-network signaling.

2 Introduction

With the development of optical multiplexing using technologies such as Dense Wave Division Multiplexing (DWDM), a new layer has begun to evolve in fiber-optics-based telecommunications networks. This is the optical network layer that provides transport services to interconnect clients such as IP routers, MPLS Label Switching Routers, ATM switches, SONET ADMs, etc. In its initial form, the optical transport network uses SONET/SDH as the framing structure, but the network may not be limited to these clients in the future. The UNI is the service control interface between the transport network and client equipment. The motivation for developing a standard UNI and the value proposition are described in the Carrier Services Framework document [OIF2000.155]. Signaling over the UNI is used to invoke services that the transport network offers to clients. This document defines the services offered over the UNI, the signaling protocols used for invoking the services, the mechanisms for transporting signaling messages and auto-discovery procedures that aid in signaling.

2.1 UNI Activities and Roles

2.1.1 Activities

Five activities are supported across the UNI:

1. Connection establishment (signaling)
2. Connection deletion (signaling)
3. Status exchange (signaling)
4. Auto-discovery (signaling)
5. Use (traffic)

The result of connection establishment is the creation of a connection. The result of connection deletion is the removal of a connection. The result of status exchange is the discovery of connection status. The result of auto-discovery is the discovery of connectivity between the client and the network and the services available from the network. The result of "use" is the exchange of user information over a connection, i.e. this is the traffic exchange.

2.1.2 Roles

For each activity there is a client and a server role. In all cases the server role is provided by (equipment that is) an agent of the server-layer-network, and the client role is provided by an agent of the client-layer-network.

Thus we have 8 roles:

client	client	client	client	client
|	|	|	|	|
Connection establishment	Connection deletion	Status-exchange	Auto-Discovery	Use
|	|	|	|	|
server	server	server	server	server

The same agent need not necessarily perform the five client roles, i.e. they need not necessarily be co-located in the same equipment. Similarly the five server roles need not necessarily be performed by the same agent.

The five client/server interactions need not necessarily take place over the same facility (this is the in-fiber/out-of-fiber discussion, see Section 6). The client and server roles for the UNI-C and UNI-N for these functions are described in Sections 8, 9, 11 and 12.

2.2 Outline of the Specification

This document deals with the following topics:

- Definition of UNI signaling reference configurations: Two sets of configurations covering direct and indirect service invocation are defined.
- Definition of services offered over the UNI.
- Definition of different signaling channel configurations for in-fiber and out-of-fiber signaling.
- Definition of the addressing scheme used under UNI 1.0.
- Definition of the automatic neighbor discovery procedure. This allows a transport network element and a directly attached client device to discover their connectivity automatically and verify the consistency of configured parameters.
- Definition of the service discovery procedure. This allows a client device to determine the parameters of the services offered over the UNI.
- Definition of UNI signaling messages and attributes, and
- Definition of UNI signaling protocols. UNI 1.0 signaling is based on the adaptations of two MPLS signaling protocols, LDP [RFC3036] and RSVP [RFC2205], and their GMPLS extensions [GMPLS SIG, GMPLS CR-LDP, GMPLS RSVP].
- Description of security issues in UNI 1.0.

2.3 Document Organization

This document is organized as follows:

- Section 3 describes the terminology and abbreviations used in the rest of the document
- Section 4 defines the services offered under UNI 1.0
- Section 5 describes the UNI signaling reference configurations
- Section 6 describes the signaling transport mechanisms
- Section 7 describes the addressing scheme under UNI 1.0
- Section 8 describes the automatic neighbor discovery procedures
- Section 9 describes the automatic service discovery procedure
- Section 10 defines the UNI abstract messages and attributes
- Section 11 describes the LDP-based UNI signaling protocol
- Section 12 describes the RSVP-based UNI signaling protocol
- Section 13 describes policy and security capabilities supported under UNI 1.0
- Section 14 contains the references.
- Appendix A describes the ITU terminology and models relevant to this specification

2.4 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].

3 Terminology and Abbreviations

The key terminology and abbreviations used in the rest of the document are summarized below. Additionally, related ITU terminology and functional models are described in Appendix A for informational purpose.

3.1 Terminology

Optical Transport Network or **Transport Network**	An optical transport network is an abstract representation, which is defined by a set of access points (ingress/egress) and a set of network services. The actual implementation is assumed to be composed of a set of transparent or opaque transport network elements such as OEO or all-optical Cross-Connects, Add/Drop Multiplexers (ADM), etc, that are interconnected using point-to-point optical links (single channel or wavelength division multiplexed optical line systems). In this document, the term "Transport Network" is used interchangeably with "Optical Transport Network". Furthermore, these terms are used to refer to the service provider transport network and not the user or client (see below) transport network.
Transport Network Element (TNE)	A network element (within the transport network) having optical interfaces, such as an optical cross-connect (OXC) or an optical add/drop multiplexer.
Port	The hardware interface in an optical or user network element that terminates a bi-directional link between network elements. Examples include OC-48 or OC-192 ports in a TNE.
User or Client	Network equipment that is connected to the transport network for utilizing optical transport services. Examples of clients include IP routers, ATM switches, Ethernet Switches, SDH/SONET Cross-connects, etc.
Connection	A circuit connecting an ingress TNE port and an egress TNE port across the transport network for transporting user signals. The connection may be unidirectional or bi-directional. (This is referred to as "Network Connection" in ITU terminology, see Appendix A)
Service Path or Trail	The user service path is the logical end-end connection between user interfaces. As such, the service path is realized on top of the optical connections and terminates at client termination points. (This is referred to as "Trail" in ITU terminology)
Client-Layer	A layer acting as a client with regard to transport services provided by a server layer (in this case, the transport network). Example of a client layer is IP.
Client-Layer Address	An address used in client-layer protocols. Example is IP addressing in IP clients connected to the transport network.
Transport Network Address	Address of an entity (e.g., a TNE) within the transport network.
Transport Network Assigned (TNA) Address	An address assigned to a client by the transport service provider, either via a protocol or by configuration.
UNI	The user-network interface is the service control interface between a client device and the transport network.
UNI-C	The logical entity that terminates UNI signaling on the client device side.
UNI-N	The logical entity that terminates UNI signaling on the transport network side.
UNI Signaling Channel	This is the logical communication channel between the UNI-C and the UNI-N over which UNI signaling messages are sent.

IP Control Channel	The communication channel over which IP packets are transported between two devices.
In-Fiber Signaling	In-fiber signaling refers to the transport of signaling traffic over a communication channel embedded in the data-bearing physical link. .
Out-of-Fiber Signaling	Out-of-fiber signaling refers to the transport of signaling traffic over a dedicated communication link, separate from the data-bearing link, between the signaling entities.
Signal Type	A SDH/SONET signal type, such as STS-1.
Wavelength Division Multiplexing (WDM)	A technology that allows multiple optical signals operating at different wavelengths to be multiplexed onto a single fiber.

3.2 Abbreviations

DCC	Data Communication Channel
GSMP	Generic Switch Management Protocol
IPCC	IP Control Channel
ISI	Internal Signaling Interface
LDP	Label Distribution Protocol
LMP	Link Management Protocol
LTE	Line Terminating Equipment
MPLS	Multi-Protocol Label Switching
GMPLS	Generalized Multi-Protocol Label Switching
ND	Neighbor Discovery
OC-N	Optical Carrier level N
OH	Overhead
LOH	Line Overhead
MSOH	Multiplex Section Overhead
RSOH	Regenerator Section Overhead
TNA	Optical-Network Assigned
ONE	Transport network Element
RSVP	Resource reSerVation Protocol
RSVP-TE	RSVP with Traffic Engineering extensions
STE	Section Terminating Equipment
STM-M	Synchronous Transport Module level M
STS-N	Synchronous Transport Signal level N
TLV	Type-Length-Value encoding
UNI	User Network Interface
UNI- N	UNI Signaling Agent – Network
UNI-C	UNI Signaling Agent – Client

4 Services Offered over the UNI (Version 1.0)

The primary service offered by the transport network over the UNI is the ability to create and delete connections on-demand. A connection is a fixed bandwidth circuit between an ingress and an egress access point (i.e., a port) in the transport network, with specified framing. The connection can be either unidirectional or bi-directional. Under UNI 1.0, this definition is restricted to consider a connection as being a SONET service of bandwidth STS-1and higher, or SDH service of bandwidth VC-3 and higher. The properties of the connection are defined by the attributes specified during connection establishment.

4.1 UNI 1.0 Signaling Actions

UNI *signaling* refers to the message exchange between a UNI-C and a UNI-N entity to invoke transport network services. Under UNI 1.0 signaling, the following actions may be invoked:

1. *Connection creation*: This action allows a connection with the specified attributes to be created between a pair of access points. Connection creation may be subject to network-defined policies (e.g., user group connectivity restrictions) and security procedures.
2. *Connection deletion*: This action allows an existing connection to be deleted.
3. *Connection status enquiry*: This action allows the status of certain parameters of the connection to be queried.

Connection modification, which allows parameters of an already established connection to be modified, is not supported under UNI 1.0.

4.2 Connection Types

Connection type is defined by a combination of framing (e.g., SONET or SDH), concatenation (contiguous or virtual concatenation), and transparency of the signal type carried. UNI 1.0 specification supports the following combinations. It should be noticed that even though UNI 1.0 signaling supports all these connection types, a given transport network may support only a subset of them.

Framing: SONET/SDH framed signals are supported.

Transparency: With SONET framing, the client may request the network to be "transparent" to different overhead bytes, i.e., the network must not modify these overhead bytes. The following transparency types are supported:

Section transparency: The section, line, and path overhead bytes should not be modified.

Line transparency: The line and path overhead bytes should not be modified.

Path transparency: The path overhead bytes should not be modified.

The corresponding terminology under SDH framing are Regenerator Section (RS), Multiplex Section (MS) and Virtual Container (VC) transparency, respectively.

Signal Types: SONET "elementary" signal types STS-1 SPE, STS-3c SPE, STS-1, STS-3, STS-12, STS-48, STS-192 and STS-768 are supported. SDH "elementary" signal types VC-3, VC-4, STM-0, STM-1, STM-4, STM-16, STM-64 and STM-256 are supported.

Concatenation: In addition to the "non-concatenated" signal types (defined above) contiguous and virtual concatenation of permitted "elementary" signal types are also supported. This applies to VC-3 (only virtual concatenation)/VC-4 for SDH and STS-1 SPE/STS-3c SPE for SONET through signaled connection services. Note: "Support" for virtual concatenation consists of signaling through the UNI from one client to

another that the indicated channels are to be virtually concatenated. The network itself is not involved in virtual concatenation (see Sections 11 and 12.)

Concatenation of these signals is defined as the process of summing the bandwidth of a number of smaller containers into a larger bandwidth container (or a procedure whereby multiple Virtual Containers are associated such that their combined capacity can be used as a single container across which bit sequence integrity is maintained).

Two methods for concatenation are defined, contiguous and virtual concatenation. Both methods provide concatenated bandwidth of X times Container-N at the path termination:

- **Contiguous concatenation** maintains the contiguous bandwidth throughout the whole transport. This requires concatenation functionality at each network element.

- **Virtual concatenation** breaks the contiguous bandwidth into individual VCs, transports the individual VCs and recombines these VCs to a contiguous bandwidth at the end point of the transmission. This requires concatenation functionality only at the path termination equipment. Virtual concatenation is defined in the scope of the UNI 1.0 for the following signals:

    - SDH: VC-3/4 to provide transport for payloads requiring greater capacity than one VC-3/4
    - SONET: STS-1 SPE (and STS-3c SPE) to provide transport for payloads requiring greater capacity than one DS3/T3.

Note that it is possible to perform a conversion between the two types of concatenation. The conversion between virtual and contiguous VC-4 concatenation is defined in ITU-T G.783 (S4 Function).

Contiguous concatenation of X VC-4 signals (VC-4-Xc, X = 4, 16, 64, 256): A VC-4-Xc provides a payload area of X Container-4. The VC-4-Xc is transported in X contiguous AU-4 in the STM-N signal.

SONET equivalent is defined as the contiguous concatenation of X STS-3c SPE signals (STS-3c-Xc, X = 4, 16, 64, 256).

Virtual concatenation of X VC-3/4 signals (VC-3/4-Xv, X = 1 ... 256): A VC-3/4-Xv provides a contiguous payload area of X Container-3/4 (VC-3/4-Xc). The container is mapped in X individual VC-3/4s which form the VC-3/4-Xv. Each VC-3/4 has its own POH where the H4 POH byte is used for the virtual concatenation specific sequence and multi-frame indication.

Each VC-3/4 of the VC-3/4-Xv is transported individually through the network. Due to different propagation delay of the VC-3/4s a differential delay will occur between the individual VC-3/4s.This differential delay has to be compensated and the individual VC-3/4s have to be realigned for access to the contiguous payload area. The realignment process has to cover at least a differential delay of 125 µs.

SONET equivalent is defined as the virtual concatenation of X STS-1 SPE/STS-3c SPE signals (STS-1-Xv SPE, X = 1 … 768 and STS-3c-Xv SPE, X = 1 … 256).

Connection Type Summary

Table 4-1 summarizes the valid combinations of the signal type, the transparency levels and the concatenation types supported by SDH client interfaces:

SDH Signal Type	Transparency	Concatenation	Remark
VC-3	Path	No Concatenation	VC-3 (only VC Transparency)
	Path	Virtual Concatenation	VC-3-Xv (only VC Transparency)
VC-4	Path	No Concatenation	VC-4 (only VC Transparency)
	Path	Contiguous Concatenation	VC-4-Xc (only VC Transparency)
	Path	Virtual Concatenation	VC-4-Xv (only VC Transparency)
STM-0	RS/MS	Not applicable	
STM-1	RS/MS	Not applicable	
	RS/MS	Contiguous Concatenation	Correspond to VC-4 with Transport OH
STM-4	RS/MS	Not applicable	
	RS/MS	Contiguous Concatenation	STM-4c: VC-4-4c (with Transport OH)
STM-16	RS/MS	Not applicable	
	RS/MS	Contiguous Concatenation	STM-16c: VC-4-16c (with Transport OH)
STM-64	RS/MS	Not applicable	
	RS/MS	Contiguous Concatenation	STM-64c: VC-4-64c (with Transport OH)
STM-256	RS/MS	Not applicable	
	RS/MS	Contiguous Concatenation	STM-64c: VC-4-256c (with Transport OH)

Table 4-1 : SDH Connection Types Supported

Table 4-2 summarizes the valid combinations of the signal type, the transparency levels and the concatenation types supported by SONET client interfaces:

SONET Signal Type	Transparency	Concatenation	Remark
STS-1 SPE	Path	No Concatenation	STS-1 SPE (only SPE Transparency)
	Path	Virtual Concatenation	STS-1-Xv SPE (only SPE Transparency)
STS-3c SPE	Path	No Concatenation	STS-3c SPE (only SPE Transparency)
	Path	Contiguous Concatenation	STS-3c-Xc SPE (only SPE Transparency)
	Path	Virtual Concatenation	STS-3c-Xv SPE (only SPE Transparency)
STS-1	Section/Line	Not applicable	
STS-3	Section/Line	Not applicable	
	Section/Line	Contiguous Concatenation	STS-3c (with Transport OH)
STS-12	Section/Line	Not applicable	
	Section/Line	Contiguous Concatenation	STS-12c (with Transport OH)
STS-48	Section/Line	Not applicable	
	Section/Line	Contiguous Concatenation	STS-48c (with Transport OH)
STS-192	Section/Line	Not applicable	
	Section/Line	Contiguous Concatenation	STS-192c (with Transport OH)
STS-768	Section/Line	Not applicable	
	Section/Line	Contiguous Concatenation	STS-768c (with Transport OH)

Table 4-2: SONET Connection Types Supported

Note: STS-Nc SPE with N = 3 * X (i.e. STS-3c-Xc SPE) signals are supported only when both STS-3c SPE elementary signal type and Contiguous Concatenation type are supported.

4.3 Supporting Procedures

The following procedures that support UNI signaling are specified in this document. Among these, the neighbor discovery and service discovery procedures are *optional*, i.e., a compliant UNI 1.0 signaling

implementation need not include these procedures. In this case, the information provided by these procedures must be manually configured.

4.3.1 UNI Neighbor Discovery

The neighbor discovery procedure is fundamental for dynamically establishing the interface mapping between a client and a TNE**.** It aids in verifying local port connectivity between the TNE and client devices. It also allows the UNI signaling control channel to be brought up and maintained. Neighbor discovery procedures for different signaling control reference configurations are described in Section 8.

4.3.2 Service Discovery

Service discovery is the process by which a client device obtains information about available services from the transport network and the transport network obtains information about the client UNI signalling (i.e. UNI-C) and port capabilities. The service discovery protocol is described in Section 9.

4.3.3 Signaling Control Channel Maintenance

UNI signaling requires a control channel between the client-side and the network-side signaling entities. Different control channel configurations are possible, as defined in Section 6. UNI 1.0 supports procedures for maintenance of the control channel under all these configurations, as described in Section 8.

4.3.4 Mandatory and Optional Procedures

As mentioned here above, several procedures are described within the UNI 1.0 specification. Some of them are mandatory (e.g., signaling protocol and control channel maintenance procedures) while others are optional. In order to be UNI 1.0 compliant, a device (UNI-C or UNI-N) must implement all the mandatory features and their corresponding protocol implementations. Among mandatory procedures, there are sometimes choices (e.g., RSVP and LDP). For UNI signaling to work, the UNI-N and the UNI-C must use compatible choices for mandatory procedures. This can only be ensured by prior administrative communication.

The following table summarizes the UNI 1.0 procedures, the mandatory and optional status of these, the choices available for implementing the procedure, and the minimum acceptable implementation for UNI 1.0 compliance:

UNI 1.0 Procedures	**Status**	**Choices**	**Minimum Acceptable Implementation for UNI 1.0 Compliance**
Signaling	Mandatory	RSVP-TE (Sec. 12), LDP (Sec. 11)	RSVP-TE or LDP
Control Channel Realization	Mandatory	In-fiber, Section DCC (Sec. 7) In-fiber, Line DCC (Sec. 7) Out-of-fiber, IP network (Sec. 7) Out-of-fiber, Dedicated SONET/SDH Connection (Sec. 7)	Out-of-fiber, IP network

Framing for in-fiber packet data communication over DCC	Mandatory with In-fiber Section or Line DCC-based control channel realization	PPP framing over HDLC (Sec. 7) PPP framing over ISO 9577 over LAP-D (Sec. 7)	At least one of these
Control Channel Maintenance	Mandatory	LMP procedure for control channel maintenance, for the selected control channel realization (Sec. 8)	LMP procedure for control channel maintenance for the selected control channel realization
In-fiber neighbor discovery	Optional	LMP procedure (Sec. 8)	Manual configuration
Out-of-fiber neighbor discovery	Optional	LMP procedure (Sec. 8)	Manual configuration
Service Discovery	Optional	LMP-based procedure (Sec. 9)	Manual configuration

Table 4-3: Mandatory and Optional Procedures under UNI 1.0

5 UNI Service Invocation Reference Configurations

The reference configurations described in this section indicate different ways in which transport network services may be invoked. There are two service invocation models, one called *direct* invocation and another called *indirect* invocation. Under both models, the client-side and network-side UNI signaling agents are referred to as UNI-C and UNI-N, respectively. In the direct invocation model, the client invokes transport network services directly. The UNI-C functionality is therefore present in the client itself. In the indirect invocation model, an entity called the *Proxy* UNI-C performs UNI functions on behalf of one or more clients. The clients are not required to be co-located with the Proxy UNI-C. Four basic configurations are described in this section, two each for the direct and the indirect invocation models. Other configurations can be formed based on combinations of the four basic configurations. In each of these configurations, UNI signaling messages between the UNI-C and the UNI-N are transported over an IP control channel, as described in Sections 6 and 7. UNI signaling protocols are based on extending RSVP-TE and LDP, as defined in Sections 11 and 12.

5.1 The Direct Invocation Model

Under this model, the UNI-C functionality is implemented in the client itself. There are two basic configurations under this model, as shown in Figure 5-1a,b. The difference between these two configurations is that under the first configuration (Figure 5-1a), the UNI-N functionality is implemented in the transport network element (TNE), whereas in the second (Figure 5-1b) the UNI-N is a proxy. When the UNI-N is implemented in a proxy entity, there is no restriction on where the proxy is located. An internal signaling interface (ISI) within the transport network is used to carry out signaling between the UNI-N and the TNE, as shown in Figure 5.1b. The details of this internal interface are not relevant to the specification of the UNI.

Under the direct invocation model, the UNI neighbor discovery function may be available over each data interface between the client and the TNE. The IP control channel between the UNI-C and the UNI-N can be in-fiber or out-of-fiber, as described in Section 6. The trigger for the client to invoke transport network services may come from a management system in the client network or via traffic engineering decisions made by the client itself. Within the transport network, the services requested over the UNI may be provided through a centralized provisioning system or by the use of distributed protocols. These aspects are not considered in the definition of the UNI.

5.2 The Indirect Invocation Model

Under this model (Figure 5.2a,b), clients invoke transport network services using proxy signaling. An entity called the Proxy UNI-C performs UNI signaling functions on behalf of one or more clients. The clients are not required to be co-located with the Proxy UNI-C. Under this model,

1. The client *may* not implement the automatic neighbor discovery procedure (Section 8). In this case, the UNI-N and the proxy UNI-C must be manually configured with the neighbor information.
2. The Proxy UNI-C performssignaling on behalf of the clients it represents.
3. The Proxy UNI-C and a client it represents may communicate using a proprietary or standard internal signaling interface (ISI). Such interfaces (e.g., GSMP) are beyond the scope of this specification.

The Proxy UNI-C and the UNI-N must run the UNI signaling protocols defined in this specification (Sections 11 and 12). The IP control channel between the Proxy UNI-C and the UNI-N is out-of-fiber, as described in Section 6. The UNI-N entity must be aware of each Proxy UNI-C that is authorized to signal on behalf of clients.

The parameters that must be configured in the Proxy UNI-C for each client it represents include:

- The client-layer address, which can be of any type described in Section 7.
- The TNA addresses assigned to the client.
- The node ID and port ID of the client endpoints connected to the transport network (if automatic neighbor discovery is not implemented).
- The signaling control channel: the out-of-fiber IPCC along with the address to which signaling messages should be sent.
- The service capabilities of the client (e.g. SONET/SDH capabilities).

A Proxy UNI-C may support multiple clients. Furthermore, the two endpoints of an optical connection may be handled by any of the following combinations:

- Proxy UNI-C and UNI-C
- Two separate Proxy UNI-Cs, one for each endpoint
- The same Proxy UNI-C for both endpoints.

There are two basic configurations under this model as shown in Figure 5-2a,b. Figure 5-2a shows the case where the UNI-N functionality is resident in the TNE. Figure 5-2b shows the case where the UNI-N functionality is outside of the TNE, with an internal signaling interface.

5.3 Service Invocation Configurations

Under this specification, it is permissible to use more than one UNI service invocation method in the same network. For example, some clients may use direct invocation while others use indirect invocation. Similarly, on the network side, it is permissible to have some TNEs with UNI signaling capability while others utilize proxy agents. Finally, clients at each end of a connection are allowed to use different service invocation methods.

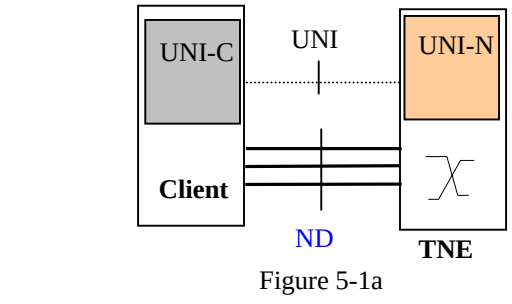


Figure 5-1a

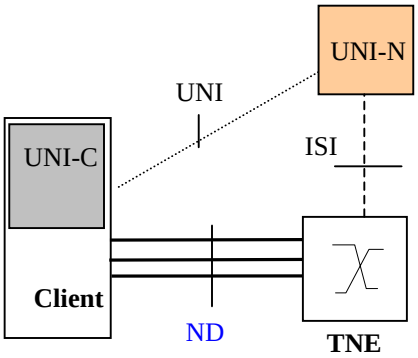


Figure 5-1b

Figure 5-1a,b: Direct Service Invocation Configurations

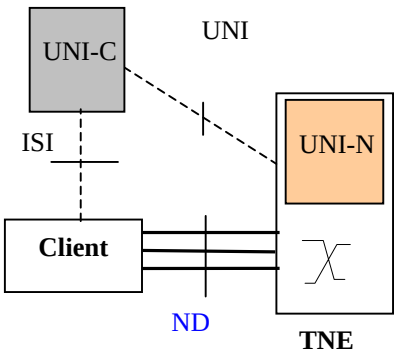


Figure 5-2a

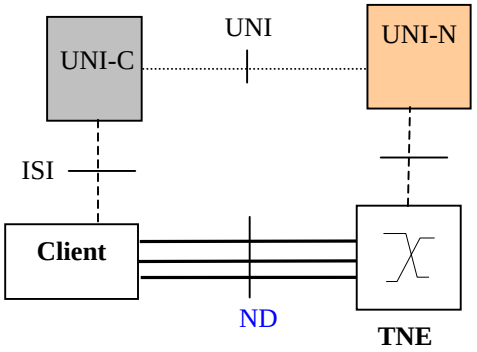


Figure 5-2b

6 Signaling Transport Configurations

As per the UNI service invocation reference configurations described in Section 5, a control channel is required between the UNI-C and the UNI-N to transport signaling messages. The UNI 1.0 specification supports the following types of signaling transport configurations:

In-fiber: The signaling messages are carried over a communication channel embedded in the data-carrying optical link between the client and the TNE. This type of signaling applies only to the direct service invocation model depicted in Figure 5-1a.

Out-of-fiber: The signaling messages are carried over a dedicated communication link between the UNI-C and the UNI-N, separate from the data-bearing optical links. This type of signaling applies to the direct service invocation model shown in Figure 5-1b, as well as the indirect service invocation model shown in Figure 5-2a,b.

In both these cases, a control channel must support the transport of IP packets. Hence, such a control channel is called an IP control channel or IPCC. UNI signaling messages themselves are carried over IP. For example, the UNI signaling protocol could be based on RSVP, in which case IP packets carrying RSVP messages will flow over the IPCC.

Under UNI 1.0, an IPCC may be realized in one of the following ways:.

6.1 In-fiber Signaling over SONET/SDH Line or Section DCC Bytes

This configuration is shown in Figure 6-1. Here, the client and the TNE are connected by one or more bi-directional SONET/SDH links. Each bi-directional link consists of a pair of unidirectional links, one from the client to the TNE, and another from the TNE to the client. Certain SONET/SDH overhead bytes in *one or more* such bi-directional links are then used to realize bi-directional IPCC(s).

An in-fiber IPCC is logically equivalent to a point-to-point link. If there are multiple such links (IPCCs) available between a client and a TNE, an IP packet may be transmitted on any one of them as per a local decision procedure.

Under this specification, an in-fiber IPCC must be realized using SONET line/SDH Multiplex Section (MS) or SONET section/SDH Regenerator Section (RS) DCC (Data Communication Channel) bytes:

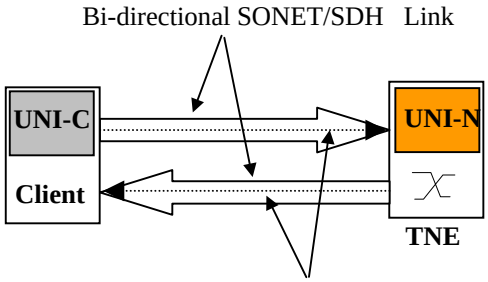


Figure 6-1: In-Fiber, Embedded Signaling Channels

- **Section (RS) DCC:** This consists of D1, D2, and D3 section overhead bytes. Used as a transparent sequence of bytes, these three bytes provide a 192 Kbps message channel.
- **Line (MS) DCC:** This consists of D4–D12 line overhead bytes. Overall available bandwidth is 576Kbps.

Two options are defined under UNI 1.0 for encapsulating IP packets over DCC bytes. These are shown in Figure 6-2. Under the first option, IP packets are encapsulated using PPP in HDLC like framing as per [RFC1662], with bit stuffed framing. Under the second option, IP over PPP over ISO 6577 with LAP-D framing is used. In both cases, only PPP encapsulation must be used with no other PPP procedures (i.e., LCP and NCP need not be executed). A client device or transport network element must support at least one of these encapsulation methods, and may support both. When both methods are supported in the client device or TNE for a given IPCC, it must be possible to select one of them by manual configuration in the client and/or the TNE.

The automatic discovery by the client and TNE of the other's IP address (to send UNI signaling packets), and the maintenance of IPCCs, etc., are described in Section 8.

6.2 Out-of-Fiber Signaling

Out-of-fiber signaling applies to all four reference configurations described in Section 5. Two options are available for this under UNI 1.0, as described next. Under both these options, an out-of-fiber IPCC is realized by establishing IP connectivity between the UNI-C and the UNI-N independent of the number of data links between the corresponding client and the TNE.

6.2.1 Out-of-Fiber Signaling over an External IP Transport Network

In this case, UNI signaling messages are carried over an external IP transport network. As shown in Figure 6-3, the UNI-C and the UNI-N have one or more points of attachment to IP transport networks. The nature of these networks is not specified in this document. The IP interface addresses of the UNI-C and the UNI-N to each of these networks is also shown. Control messages may be sent over either network without coordination. But each entity, UNI-C and UNI-N, must be configured with the IP address of the other to be able to send control messages. Under UNI 1.0, it is assumed that the IP networks through which an IPCC is realized are secure. That is, no additional procedures are specified to ensure secure signaling message transfer over external control networks.

6.2.2 Out-of-Fiber Signaling over a Dedicated Signaling Channel

In this case, a dedicated, bi-directional SONET/SDH connection may be established between the UNI-C and the UNI-N. This connection is then used as the IPCC, and the signaling messages are sent in the payload. Under this specification, the establishment of the connection and its usage are controlled by suitable manual configuration at the peer entities. There could be more than one bi-directional IPCC established in this manner. Any one of them may be used for sending a particular signaling message. This specification requires that IP packets sent over an IPCC realized using a dedicated connection should use PPP over SONET/SDH, as defined in [RFC 2615].

6.3 Signaling Transport Realization

Although three distinct signaling transport options are recognized, this specification does not preclude the implementation of more than one type of transport link between a client and a TNE. For instance, a client and a TNE could have both in-fiber and out-of-fiber IPCCs. Similarly, a TNE could have different types of IPCCs with different clients. Thus, configuration is required in clients and TNEs to indicate which type of IPCC is implemented for which sets of data links.. Procedures for keeping track of the status of IPCCs between a client and a TNE are described in Section 8.

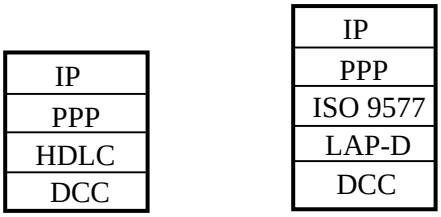


Figure 6-2: DCC Encapsulation Options

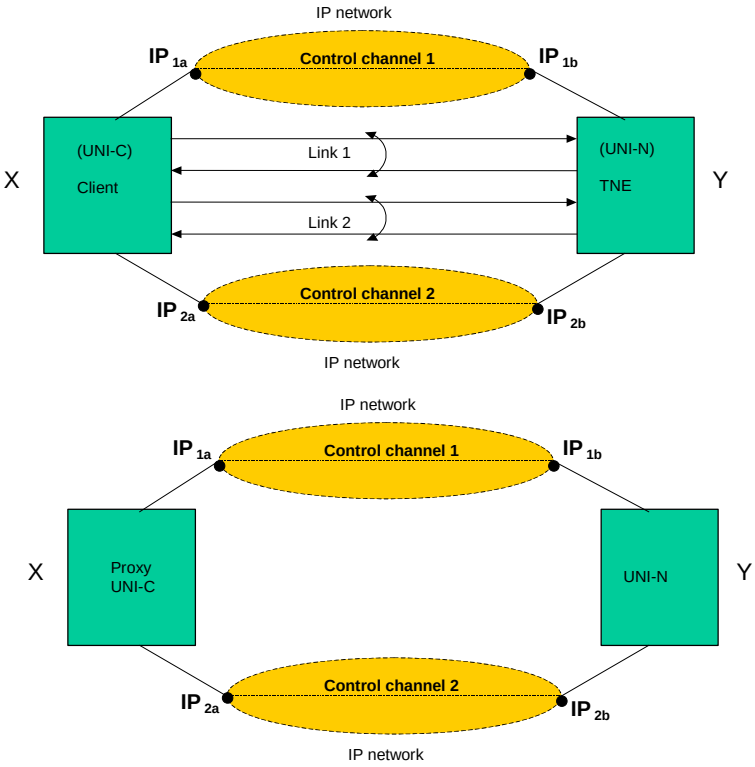


Figure 6-3: Out-of-Fiber Signaling Configuration

7 Addressing

7.1 Control Entities Requiring Identification for UNI Operation

UNI addressing enables the identification and reachability of various entities associated with the optical transport network and the connection control plane. In order to describe the multiple address spaces involved in the operation of the UNI protocol, it is necessary to introduce the entities that need to be identified, as well as the relevant terminology.

As described in Section 6, signaling protocol messages are exchanged over IP control channels. To recapitulate,

- **(IP) Control Channel (IPCC)**: This is a data channel over which IP packets are transported between UNI-C and UNI-N nodes. An IPCC may be realized in a number of ways and a pair of UNI-C and UNI-N nodes may be connected by multiple control channels (used either concurrently or in a primary-backup configuration) as described in Section 6.

- **Signaling Channel**: This is the logical channel over which signaling messages are exchanged between UNI-C and UNI-N. The signaling channel is realized using the collection of all IPCCs between the two UNI peers, as well as the signaling protocol entities at the respective peers. A signaling channel failure could be the result of the concurrent failure of all IPCCs or the failure of a signaling entity.

Figure 7-1 shows an example configuration of a UNI-C and a UNI-N, co-located with a client and a TNE, respectively, and connected by four data links. Each of the data links could be used to realize an in-fiber IPCC as described in Section 6. The two nodes also have an out-of-fiber IPCC (see Section 6), realized over an external Ethernet. In the context of IP networking, each IPCC represents a point-to-point link connecting the two nodes and may be either *numbered* or *unnumbered*. In the former case, each IPCC is identified by a separate IP address at each end. In the latter case, an individual IPCC is not assigned IP addresses. Rather, it is assigned an interface index on each side, and each end is identified by the combination of the interface index and an IP address assigned to the node.

As shown in Figure 7-1, the UNI signaling channel is layered on top of one or more IPCCs. The signaling channel itself must be uniquely identified, in a way that is independent of changes in the status of the underlying IPCC(s). This is achieved using the Node Identifier (Node ID), defined in Section 7.2.

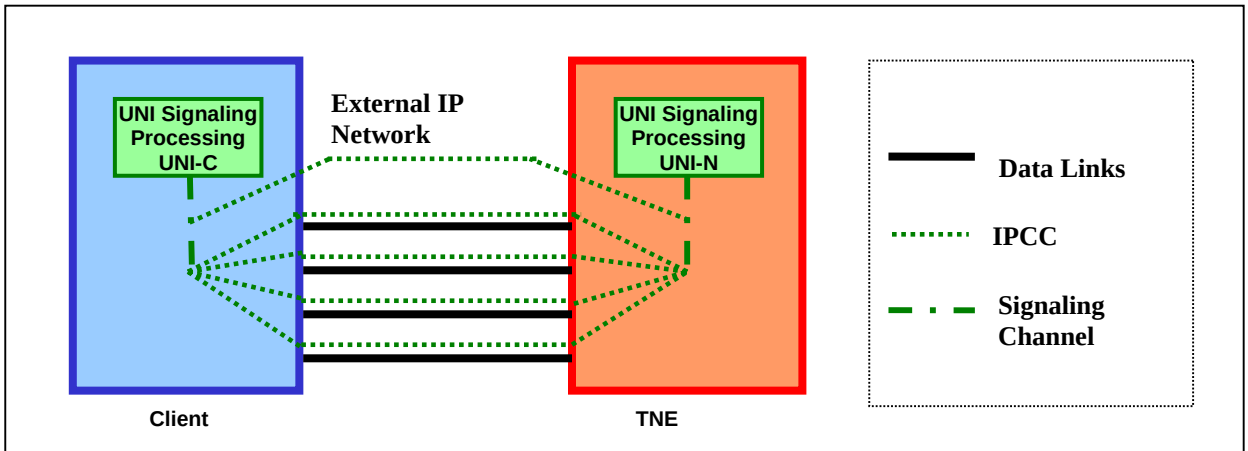


Figure 7-1: Example configuration illustrating IPCCs and the Signaling Channel between two UNI Peers

7.2 UNI Address Spaces

The following address spaces are relevant to UNI signaling:

1. **Internal transport network addresses**: These are addresses of nodes within the transport network, used for internal routing, provisioning and network management purposes. These addresses are shown in Figure 7-2, as TNE A and TNE B. These addresses will not be exported to network clients and are not the subject of standardization within the OIF. Thus, they are outside the scope of this specification.

2. **UNI-N and UNI-C Node Identifier (Node ID)**: This is an IP address that identifies connection control plane termination in a UNI-C or a UNI-N. The Node ID is invariant to changes in the address or status of underlying IPCCs. Under UNI 1.0, the Node ID is an IPv4 address, which must be unique within the domain of operation of the communicating UNI-C and UNI-N nodes. The Node ID need not be a globally unique IP address, but it must be unique only in the domain within which each node can uniquely identify the respective peer it is communicating with. Note that this requirement is trivially satisfied if the Node ID is a globally unique IPv4 address. More likely is the scenario where Node IDs are drawn from the private IPv4 address space. Migration to IPv6 addressing for the Node ID will depend on the proliferation of IPv6 capable UNI-C and UNI-N nodes. Node ID is identified in Figure 7-2 within the nodes shown. Node ID is assigned by the network operator managing the node.

3. **(IP) Control Channel Identifier (CCID)** : As discussed in Section 7.1, the endpoints of an IPCC have to be identified for the purposes of neighbor discovery, service discovery and IPCC maintenance. This identifier is called the *Control Channel Identifier or CCID*. An IPCC may be numbered, in which case it is identified by an IPv4 address at each end. If it is unnumbered then it is identified by the combination of the Node ID and an interface index at each end. In the case of numbered IPCC, migration to IPv6 addressing will depend on the proliferation of IPv6 capable UNI-C and UNI-N nodes.

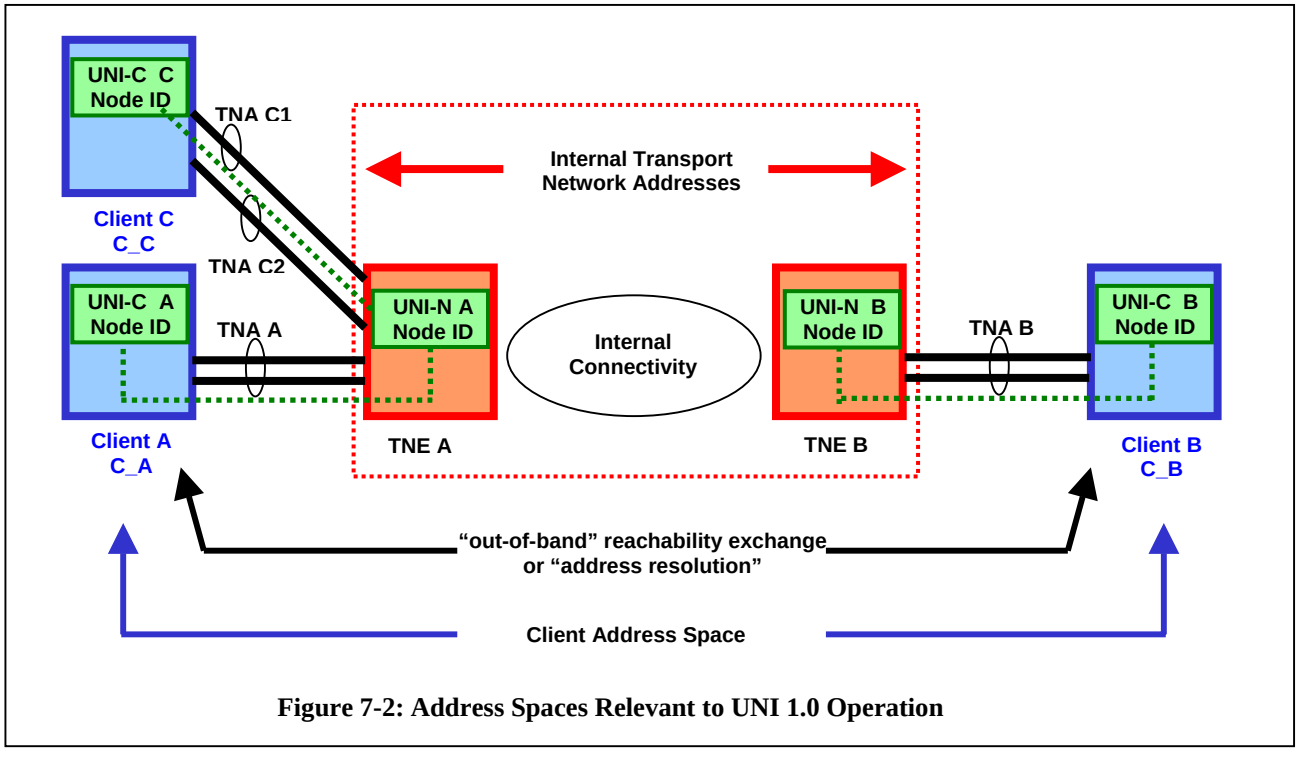


Figure 7-2: Address Spaces Relevant to UNI 1.0 Operation

4. **Transport Network Assigned (TNA) address**: UNI connection endpoints are identified by *Transport Network Assigned* (TNA) addresses, shown in Figure 7-2. Each TNA is a *globally unique* address assigned by the transport network to *one or more* data bearing links connecting a UNI-N and a UNI-C. *The basis on which the transport network operator assigns TNA addresses to data links is a matter of transport network operator policy*. The structure of TNA addresses is discussed in Section 7.4.

5. **Client addresses:**These are denoted by C_A, C_B and C_C in Figure 7-2. These follow the addressing schemes used in client networks (e.g., IP, ATM, etc) and are not directly exchanged in signaling messages. The UNI 1.0 specification realizes a separation between client and TNA address spaces. Client addresses are therefore outside the scope of the UNI 1.0 specification.

7.3 Logical Port Identifiers

As described above, a single TNA address may denote one or more data links. An individual data link within a group of links sharing the same TNA address is identified by a *logical port identifier* at each end. Assignment of logical port identifiers is a *local decision of the node at each end*. Thus, each data link connecting a client and a TNE will be associated with two logical port identifiers, one on the client side and one on the TNE side. For UNI 1.0 a logical port identifier is encoded as a 32-bit integer and must be unique within a node.

Each logical port identifier corresponds on a one-to-one basis to a physical port. The numbering schemes used for logical port identifiers and physical ports may, however, be different. This allows a network operator to avoid exposing its internal port numbering scheme to clients. Additionally, an operator may maintain the same logical port identifier for a data link to a given client device, regardless of the identity of the physical port within the TNE on which the link is terminated.

Figure 7-3 shows a client and a TNE with multiple data links between them. The corresponding logical port identifiers on each side are denoted as, CP_i and NP_i, respectively. Correct operation of UNI signaling relies on *unambiguous mapping between the client and TNE logical port identifiers*, as these are carried in connection establishment and tear-down messages. This mapping can be established either

- Automatically, using the LMP neighbor discovery and link verification procedures described in Section 8, or
- Manually, by configuring the mapping in the corresponding UNI-C and UNI-N entities.

Under this specification, a UNI-C is required to allow manual configuration of the port mapping information. Specifically, the UNI-C is required to utilize the mapping mandated by the UNI-N for each port. For example, in Figure 7-3, the UNI-C must be able to be configured to use NP_1 (assigned by the UNI-N) instead of CP_1 (assigned internally) to identify the first data link shown.

7.4 Structure of TNA Addresses

TNA addresses are used to identify the endpoints of a UNI connection. The TNA address space should be large enough to accommodate global networks and it should be capable of supporting hierarchies and summarization.

UNI 1.0 TNA addresses are capable of accommodating either of the IP or NSAP families. For example, the following three types of TNA addresses are supported by the UNI 1.0 signaling protocols:

1. IPv4 addresses (32 bits)
2. IPv6 addresses (128 bits)
3. NSAP addresses (160 bits) (The NSAP format is structured according to ISO/IEC 8348, 1993 (identical to ITU X.213, 1992)).

7.5 Role of TNA Addresses in UNI Signaling

TNA addresses are carried in UNI signaling messages to uniquely identify the data endpoints of a connection. When the same TNA address is assigned to more than one data link terminating on a client, each data link is additionally qualified using a *logical port identifier*. In this case, the combination of TNA address and logical port identifier is required to uniquely identify a data link. Furthermore, depending on the nature of the selected interface and the requested service, de-multiplexing information such as channel must be selected. This information is carried in signaling messages as a "label". A "label" indicates the detailed multiplexing information required to identify logical channels within the scope of a given data link. These details are described in Sections 11 and 12.

Under UNI 1.0, connections between heterogeneous types of TNA addresses are allowed. For example, the source endpoint could be identified by an IPv4 address, while the destination endpoint is identified by an NSAP address. Any interworking that may be required in this case is beyond the scope of the UNI specification. Details of how these addresses are carried in signaling messages are given in Sections 11 and 12.

The means by which the transport network maintains the mapping between TNAs and internal transport network addresses is beyond the scope of the UNI 1.0 specification.

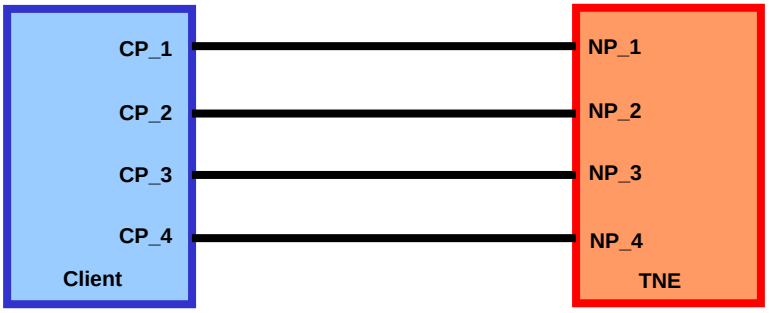


Figure 7-3: Cient and TNE Logical Port Identifier Mapping

8 Neighbor Discovery and IP Control Channel Maintenance

8.1 Overview

The neighbor discovery procedures defined in this section allow TNEs and directly attached client devices to determine the identities of each other and the identities of remote ports to which their local ports are connected. The IP control channel (IPCC) maintenance procedures defined in this section allow TNEs and clients to continuously monitor and maintain the list of available IP control channels.

Neighbor discovery and IPCC maintenance procedures are *optional* under UNI 1.0. If neighbor discovery is not implemented in the client and the TNE as described in this section then the neighbor and remote port identities MUST be manually configured in the corresponding UNI-C and UNI-N. If the control channel maintenance procedure, as described in this section, is not implemented then the signaling protocol used must implement its own procedure for this. The implementation of the neighbor discovery as described in this section, however, avoids the need for manual configuration of neighbor and port connectivity information. Therefore, it eliminates the potential errors that may arise from relying on such configuration. Furthermore, it provides a means to automatically detect inconsistencies in the physical wiring (e.g., when the transmit and receive sides of a port are connected to two different remote ports.

8.2 Scope of UNI 1.0 Neighbor Discovery

Figure 8-1 illustrates peers at different SONET layers, Section Terminating Equipment (STE), Line Terminating Equipment (LTE) and Path Terminating Equipment (PTE). The neighbor discovery procedures defined in this section are targeted for SONET LTE peers. Typically, one of them is the client device and the other is the TNE.

8.3 Outline

A TNE may be connected to multiple clients and vice versa, but for the definition of neighbor discovery and IPCC maintenance procedures, it is adequate to consider a single TNE and a directly connected client. Such a pair may be connected by a number of data links. For UNI 1.0, these MUST be SONET/SDH links. Furthermore, there may be one or more IPCCs between the client and the TNE. Such an IPCC can be realized in-fiber or out-of-fiber, as defined in Section 6. In all cases, the procedures defined in this section require the ability in TNEs and clients to send and receive certain messages in-fiber on the data links.

The protocol mechanisms defined in this section are based on the Link Management Protocol (LMP) [LMP]. Under LMP, there are two basic steps defined for IPCC maintenance:

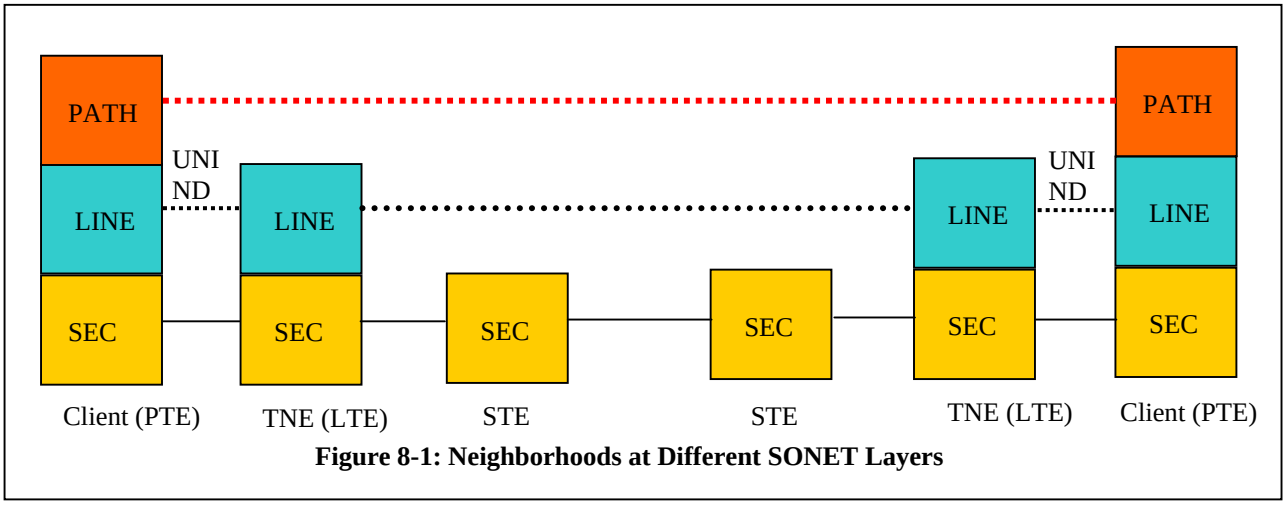


Figure 8-1: Neighborhoods at Different SONET Layers

1. *Configuration*: The basic parameters are established in the TNE and the client for each IPCC between them.
2. *Hello*: A Hello protocol instance is initiated for each IPCC in the client and the TNE. This protocol serves to monitor the availability of the control channel.

In the case of in-fiber signaling over DCC bytes (where an IPCC is embedded in each data link), the port connectivity information can also be exchanged during the IPCC configuration procedure. In the case of out-of-fiber configuration, a third step is defined in LMP for verifying port connectivity:

3. *Link Verification*: To determine port connectivity, Test messages are transmitted in-fiber over each data link. Control messages transmitted over IPCCs are used to coordinate the Test procedure.

It is expected that multiple ports connecting a TNE to a client will be aggregated into one or more logical links, each identified by a TNA (see Section 7). As such, it is useful to have a procedure to verify the parameters of the aggregated link such as bandwidth, link encoding type, and switching capability. The *Link Property Correlation* procedure of LMP is used for this. Under LMP terminology, the aggregated logical link is referred to as a Traffic Engineering (TE) link.

The following identifiers are used in LMP procedures:

- *Node Identifier* (Node ID): This is an IP address that identifies the node (TNE or the client). For example, the Node ID of a router client could be its Router ID. See also Section 7.2.
- *Interface ID.* This is a 32-bit logical identifier for each port.
- *Control Channel ID* (*CCID*): A locally unique identifier must be assigned for each IPCC configured at a node. This identifier is called the CCID, and it is a 32-bit number that is unique within the scope of a Node ID. As per this definition, the same control channel between the TNE and the client may be assigned different CCIDs by each side. In the case of in-fiber signaling over DCC bytes, the CCID MUST be the same as the Interface identifier corresponding to the port. See also Section 7.2.
- *Link ID:* For UNI 1.0, this corresponds to the TNA address. In LMP, different address types (e.g., IPv4, IPv6, etc.) are supported using different LINK_ID C-Types. C-Type 4 (reserved for OIF) is used to support NSAP addresses.

In the following sections, the LMP messages and procedures for neighbor discovery and IPCC maintenance under UNI 1.0 are described.

8.4 LMP Messages for Neighbor Discovery and IPCC Maintenance

All LMP messages are sent as IP packets (except, in some cases, the Test message which may be limited by the transport mechanism) and they can be sent over in-fiber or out-of-fiber signaling channels. To support neighbor discovery and IPCC maintenance under UNI 1.0, only a subset of LMP messages is required. Table 8-1 summarizes the LMP messages supported under UNI 1.0. Messages 1-4 are LMP control channel maintenance messages and they carry the CCID of the IPCC over which they are sent. Messages 5-16 are LMP data-link-related messages. Other than the Test message (Message 10), all messages are carried over the IPCC. The Test message is always sent in-fiber over each data link between the client and the TNE.

Message Number	LMP Message
1	Config
2	ConfigAck
3	ConfigNack
4	Hello
5	BeginVerify
6	BeginVerifyAck
7	BeginVerifyNack
8	EndVerify
9	EndVerifyAck
10	Test
11	TestStatusSuccess
12	TestStatusFailure
13	TestStatusAck
14	LinkSummary
15	LinkSummaryAck
16	LinkSummaryNack

Table 8-1: LMP Messages for UNI neighbor discovery and IPCC maintenance

8.5 Neighbor Discovery and IPCC Maintenance Procedures

8.5.1 Overview

The neighbor discovery procedure is used by a client (TNE) to determine the identities of the TNEs (clients) connected to the remote end of each data link and the manner in which the data links are interconnected. To this end, the neighbor discovery procedure allows the client and the TNE to exchange their Node IDs, determine the mapping between local and remote port identification (referred to as local Interface IDs and remote Interface IDs) and correlate the configured parameters of the corresponding data links. The IPCC maintenance procedure allows a pair of signaling peers to establish and maintain control channel connectivity for UNI signaling., Figure 8-2 illustrates a port-mapping table that is maintained at a device running UNI neighbor discovery. The local Inteface IDs and "other information" are configured when a port is provisioned. The latter may also be exchanged as part of the LMP LinkSummary message exchange. All of the remote side information (shaded) is determined automatically.

Node ID	Interface ID	Other Info. (e.g., port speed, etc.)	Remote Node ID	Remote Interface ID	Other Info.
32-bit address	1		32-bit address	5	
32-bit address	2		32-bit address	8	
32-bit address	3		Unknown	Unknown	

Figure 8-2: Example Port Table

In addition to the port-mapping table, the status of the IPCCs (including the local/remote CCIDs) must be maintained.

The three steps outlined in Section 8.3 for neighbor discovery and IPCC maintenance procedures are described in more detail below.

8.5.2 Control Channel Configuration

The control channel configuration step consists of exchanging three LMP messages: Config, ConfigAck and ConfigNack. A node (TNE or client) that initiates the configuration of an IPCC MUST send a Config message to its peer over the IPCC. The Config message contains one or more LMP objects. For UNI 1.0, the Config message MUST include the HelloConfig object, which contains the HelloInterval and HelloDeadInterval (defined in [LMP]). These parameters can be negotiable or non-negotiable, as indicated in the HelloConfig object. A node receiving a Config message must respond with a ConfigAck or a ConfigNack message, indicating the acceptance or the rejection of parameters in the Config message. The reception of a ConfigAck message by the sender of the Config message completes the configuration process. The reception of a ConfigNack message, on the other hand, may terminate the configuration process unsuccessfully, or may result in the transmission of another Config message with revised parameters.

A Config message may be sent concurrently by the client and the TNE to each other. In this case, the Client MUST stop sending Config messages and respond to the received Config message. The TNE MUST ignore the received Config messages and wait for a ConfigAck or a ConfigNack message. This procedure supercedes the contention resolution procedure defined in [LMP]. The manner in which a Config message is addressed depends on the signaling transport mechanism:

In-Fiber:

Config messages SHOULD be sent to the Multicast address (224.0.0.1). The ConfigAck and ConfigNack messages MUST be sent to the Source IP address found in the IP header of the received Config message.

Out-of-Fiber:

Config messages MUST be sent to the neighbor's IPCC address. This is configured at both ends of the control channel.

8.5.2.1 Verifying Port Connectivity

Port connectivity verification is an important part of neighbor discovery. It is used to verify the physical connectivity provided by the data channels and to exchange the Interface IDs. With in-fiber configuration, the port connectivity verification is built into the LMP configuration procedure. In the case of out-of-fiber configuration, a separate procedure is needed. These are described below:

In-Fiber:

In this configuration, there is a one-to-one mapping between a control channel and a data port. Therefore, the IPCC SHOULD be assigned the same value as the Interface ID. As such, port connectivity verification can be done during the configuration procedure as shown in Figure 8-3. Here, the TNE initiates the configuration process by sending a Config message. In addition to the HelloConfig TLV, the Config message MUST include the TNE's Node ID and the CCID (equal to the Interface ID). In response, the client returns a ConfigAck message containing its own Node ID and CCID (equal to the Interface ID) as well as the Remote Node ID and the Remote CCID. These messages allow the TNE and the client to populate the "Remote Node ID" and the "Remote Interface ID" fields of the port table (Figure 8-2).

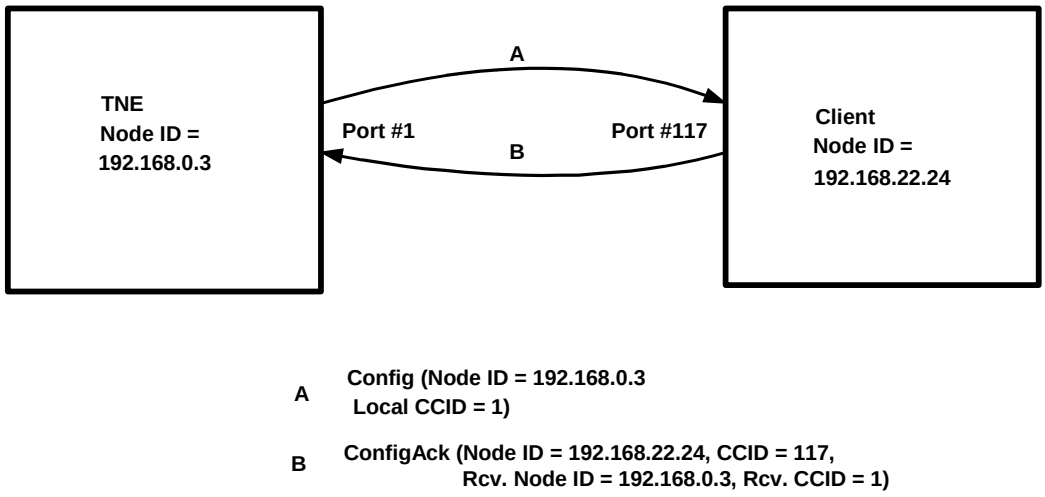


Figure 8-3: Port Connectivity Verification

Out-of-Fiber:

In this configuration, there is not a 1-to-1 mapping between a control channel and a data port. Therefore, the IPCC configuration procedure is not sufficient to determine data-port connectivity, and a separate procedure is required. To coordinate the procedure between the client and the TNE, an IPCC MUST be established between them before Port verification is initiated. Port verification is initiated by a node (client or TNE, called "initiator") by sending a BeginVerify message to the remote node over the IPCC. In response to the BeginVerify message, the remote node may send either a BeginVerifyAck or BeginVerifyNack message to the initiator. If a BeginVerifyAck message is sent, then it MUST include a 32-bit VerifyId to uniquely identify the specific verification instance to the remote node. The VerifyId is copied in all other verification-related messages sent by both the initiator and the remote node. A BeginVerifyNack message sent from the remote node indicates that the receiver is unable or unwilling to do the link verification procedure.

During the testing procedure, link verification Test messages are sent in-fiber by the initiator over each data link to the remote node. To support different mechanisms to transport the Test messages (e.g., using overhead bytes or the payload), a Verify Transport Mechanism field is included in the BeginVerify message and a Verify_Transport_Response field is included in the BeginVerifyAck message. The testing of each data link can be sequential, or multiple data links may be tested concurrently, depending on the capabilities of the nodes. To end the procedure, the initiator sends an EndVerify message to the remote node. The remote node then returns an EndVerifyAck message.

The verification procedure is as follows. After the initiator and the remote node agree to begin testing the ports, the initiating node sends a "Test" message over each unidirectional fiber link. Upon receipt of the "Test" message, the receiver correlates the Test message to a particular verification procedure instance by using the VerifyID, and maps the remote Interface ID to the corresponding local value. The receiver then sends its local Interface ID back to the initiator in a TestStatusSuccess message over the control channel. If the receiver fails to find a Test message within a specified time interval, it sends a TestStatusFailure message to the initiator. Finally, the initiator returns a TestStatusAck message to its peer to acknowledge the Success or the Failure message.

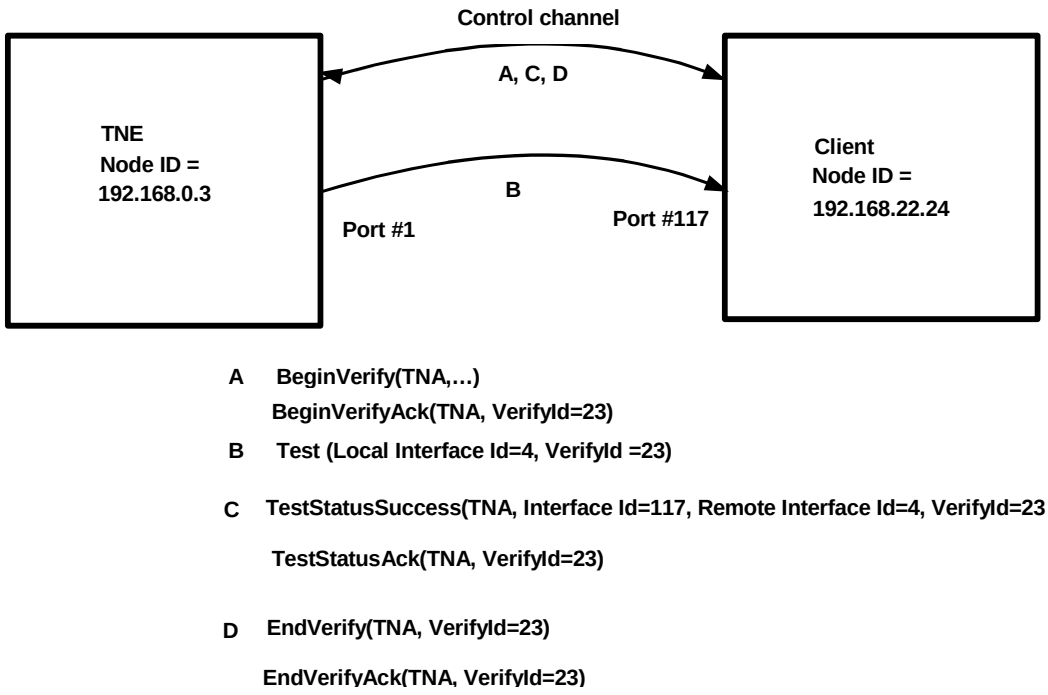


Figure 8.3: Port Connectivity Verification

8.5.3 The Hello Protocol and IPCC Maintenance

Once an IPCC is activated between the Client and TNE, the two nodes exchange Hello messages to maintain IPCC connectivity. Each node (client or TNE) executing the Hello protocol must periodically send an LMP Hello message over the IPCC. The destination address of the IP packet MUST be the address learned in the Configuration procedure (i.e., the Source IP address found in the IP header of the received Config message). The periodicity of the Hello message is governed by the HelloInterval which is established during the configuration phase.

If a node is sending Hellos but does not receive any Hellos during the HelloDeadInterval period, the corresponding IPCC MUST be declared down. In this case, the Configuration step (Section 8.5.2) may be attempted again at a later time.

8.5.4 LMP and SONET Failure Detection Mechanisms

While LMP messaging helps determine the Up/Down status of IPCCs, the status of an IPCC should also be updated using other available information (e.g., SONET alarms and signal quality information). Indeed, the latter may allow failures to be detected at a faster timescale compared to LMP. LMP Hellos, however, are still essential since they determine whether an IP control channel can be realized over a link.

8.5.5 Selection of a Physical Channel for Sending IP Control Messages

The port table maintained using LMP allows a node to determine the set of available IPCCs at any given instant. There may be many IPCCs with "Up" status to a given neighbor. To send an IP packet to that neighbor, the channel manager must first match the IP destination address with a neighbor's address, and select an IPCC whose status is "Up" to send the packet. The algorithm by which an IPCC is selected is not

specified. There need not be any coordination between a pair of neighbors in selecting a specific IPCC for sending signaling messages.

A node may choose an IPCC to a neighbor and keep using it until its status changes. In such an event, the node may choose any other IPCC to the neighbor whose status is "Up". If there is no such IPCC then UNI signaling messages cannot be sent to the neighbor. (A network management event may also be generated to note the absence of a control channel).

8.5.6 Link Property Correlation

In addition to IPCC maintenance and link verification, the link property correlation function is defined to ensure proper link configuration between the TNE and the Client. Link property correlation is done using the LinkSummary message exchange of LMP. Link property correlation MUST be done before the link is brought up and MAY be done at any time a link is UP and not in the Verification process.

The LinkSummary message exchange is used

- to aggregate multiple data links into a logical link (identified by the TNE).
- exchange, correlate, or change link parameters, and
- exchange, correlate, or change Interface Ids.

The LinkSummary exchange is defined using the LinkSummary, LinkSummaryAck, and LinkSummaryNack messages (see [LMP] for message formats).

The LinkSummary message MUST be periodically transmitted by a node until either the node receives a LinkSummaryAck or LinkSummaryNack message, or a timeout expires without receipt of a LinkSummaryAck or a LinkSummaryNack message. If a node receives a LinkSummary message from a remote node and the Interface Id mappings in the message match those that are stored locally then the two nodes have agreement on the verification procedure (see Section 8.5.4). If the verification procedure is not used then the LinkSummary message can be used to verify agreement on manual configuration.

8.5.7 Detecting Inconsistent Physical Connectivity

Detecting inconsistent physical connectivity is an important feature of neighbor discovery and is built into the port verification procedures of LMP. Such an inconsistency arises when the transmit side of a bi-directional port is connected to one remote port and the receive side is connected to another. The detection of inconsistency in connectivity is described below for the in-fiber and the out-of-fiber IPCC cases:

In-Fiber:

Inconsistency is detected by a node when either

a) ConfigAck (or Nack) message is not received in response to a Config message that has been sent, or
b) ConfigAck or Nack message received does not contain the expected "Remote Node ID" or "Remote CCID".

Out-of-Fiber:

Inconsistency is detected by a node when either

a) TestStatusFailure message is received, or
b) TestStatusSuccess message is received containing an unexpected Interface ID (either Remote Interface ID or Local Interface ID).

Detection of inconsistency for is illustrated in the examples given in Section 8.5.9.

8.5.8 Neighbor Discovery Examples

8.5.8.1 Neighbor Discovery with In-Fiber IPCC Configuration

Consider a client and a TNE connected by multiple data links, each configured to carry control channel traffic in-fiber as described in Section 6.1. As per the neighbor discovery procedures defined in this section,

- The client and the TNE execute the control channel configuration procedure over each IPCC. As per this procedure, the Interface IDs are exchanged (in the CCID field) the configuration is verified (any inconsistency is identified).

- The client and the TNE begin executing the Hello protocol over each IPCC.

8.5.8.2 Neighbor Discovery with Out-of-Fiber IPCC Configuration

Consider a client and a TNE connected by multiple component links, with one or more out-of-fiber IPCCs defined as per Sections 6.2 and 6.3. As per the procedures defined in this section,

1. Each IPCC is associated with a specific neighbor, and a set of data links (and hence ports).
2. The client and the TNE execute the control channel configuration procedure over each IPCC.
3. The client and the TNE begin executing the Hello protocol over each IPCC.
4. The client selects an IPCC leading to the TNE. It then sends a BeginVerify message over the IPCC. The BeginVerify message contains the identity of one or more of data links (i.e., one or more Interface IDs) that are to be verified and the interval (in msec), called VerifyInterval, between the transmission of successive Test messages;
5. When the TNE receives a BeginVerify message and it is ready to receive in-fiber Test messages,
   - it selects a transport mechanism for the Test messages
   - assigns a unique VerifyID to the Test procedure
   - determines the interval (in msec), called VerifyDeadInterval, that it will wait to receive Test messages, and
   - sends this information along with a BeginVerifyAck message (over an IPCC) back to the client;
6. When the client receives a BeginVerifyAck"message from the network equipment, it begins transmitting Test messages periodically to the TNE, including the Interface ID and VerifyID in-fiber over each tested link;
7. Upon receiving a Test message, the TNE records the received Interface ID, and returns a TestStatusSuccess message (over the IPCC), including the client Interface ID and the corresponding TNE Interface ID**.** If, however, a Test message is not detected by the TNE within VerifyDeadInterval, it sends a TestStatusFailure message (over the IPCC).
8. Upon receiving the TestStatusSuccess message over the IPCC, the client will be able to detect inconsistencies in physical connectivity. If an inconsistency is not identified then the client marks the link as "Discovered" and records the TNE Interface ID.
9. When all the links have been tested, the client sends an EndVerify message (over the IPCC) to indicate that the testing has been completed. Subsequently, the TNE sends an EndVerifyAck (over the IPCC).

After the neighbor discovery process is completed, any OH bytes that are used for this purpose may be put back to normal operation. The scheme is essentially non-intrusive, and allows for bit transparent operation with SONET/SDN signals.

With the above process, both the client and the TNE maintain the complete list of link identifier mappings, i.e., Remote Interface ID and Local Interface ID for each port. These mappings are then checked for consistency using the LinkSummary exchange.

8.5.9 Neighbor Discovery Examples

8.5.9.1 Example 1

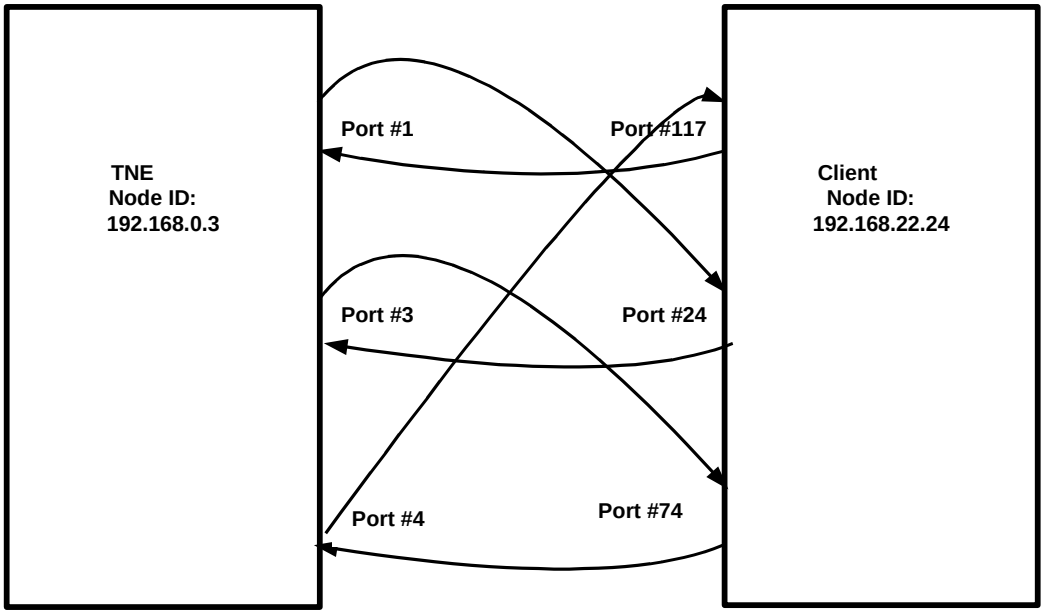


Figure 8.4: Detection of Inconsistent Physical Connectivity

In-Fiber:

In this example (Figure 8-4), inconsistent connectivity is detected by a node when a ConfigAck or ConfigNack message is received that does not contain the expected Remote Node ID or Remote CCID. This is shown as follows:

Step 1: TNE sends the Client a Config message (Local CCID = Interface ID = Port #1, Node ID = 192.168.0.3) out of Port #1. This message is received at the client on Port #24.

Step 2: Client sends the TNE a ConfigAck message (Local CCID = Interface Id = Port#24, Node ID = 192.168.22.24, Remote CCID = Port#1, Remote Node ID = 192.168.0.3) out of Port #24. This message is received at the TNE on Port #3.

At this point, the TNE detects inconsistent wiring.

Out-of-Fiber:

In this example (Figure 8-4), inconsistency is detected by a node when a TestStatusSuccess message is received containing an unexpected Interface ID. Note that Test messages are the ONLY messages transmitted in-fiber; all other messages are transmitted over the IPCC. For this example, we assume that the link verification is done sequentially. This is shown below where identification information transmitted in a message is included within brackets ().

The following procedure is initiated at the TNE:

Step 1: The TNE sends the client a BeginVerify message (TNA address, 3 data links to be tested, MessageID).
Step 2: The client sends the TNE a BeginVerifyAck message (TNA address, VerifyID=432, Received MessageID)
Step 3: The TNE sends the client a Test message (Verify ID=423, Interface ID = Port #1) out of Port #1. This message is received by the Client on Port #24.
Step 4: The client sends the TNE a TestStatusSuccess message (VerifyID=423, Interface ID = Port #24, Received Interface ID = Port #1, MessageID)
Step 5: The TNE sends the client a TestStatusAck message (VerifyID=423, Received MessageID)
Step 6: Step 3-5 are repeated for Ports #3 and #4.
Step 7: The TNE sends the client an EndVerify message (VerifyID=423, MessageID)
Step 8: The client sends the TNE and EndVerifyAck message (VerifyID=423, Received MessageID).

The above steps are also initiated at the Client as follows:

Step 1: The client sends the TNE a BeginVerify message (TNA address, 3 data links to be tested, MessageID).
Step 2: The TNE sends the client a BeginVerifyAck message (TNA address, Verify Id=727, Received MessageID)
Step 3: The client sends the TNE a Test message (Verify Id=727, Interface Id = Port #117) out of Port #117. This message is received by the TNE on Port #1.
Step 4: The TNE sends the client a TestStatusSuccess message (VerifyID=727, Local Interface ID = Port #1, Remote Interface Id = Port #117, MessageID)
Step 5: The client sends the TNE a TestStatusAck message (VerifyID=727, Received MessageID)
Step 6: Step 3-5 are repeated for Ports #24 and #74.
Step 7: The client sends the TNE an EndVerify message (VerifyID=727, MessageID)
Step 8: The TNE sends the client and EndVerifyAck message (VerifyID=727, Received MessageID).

By now, the inconsistency in connectivity is detected at both nodes.

8.5.9.2 Example 2

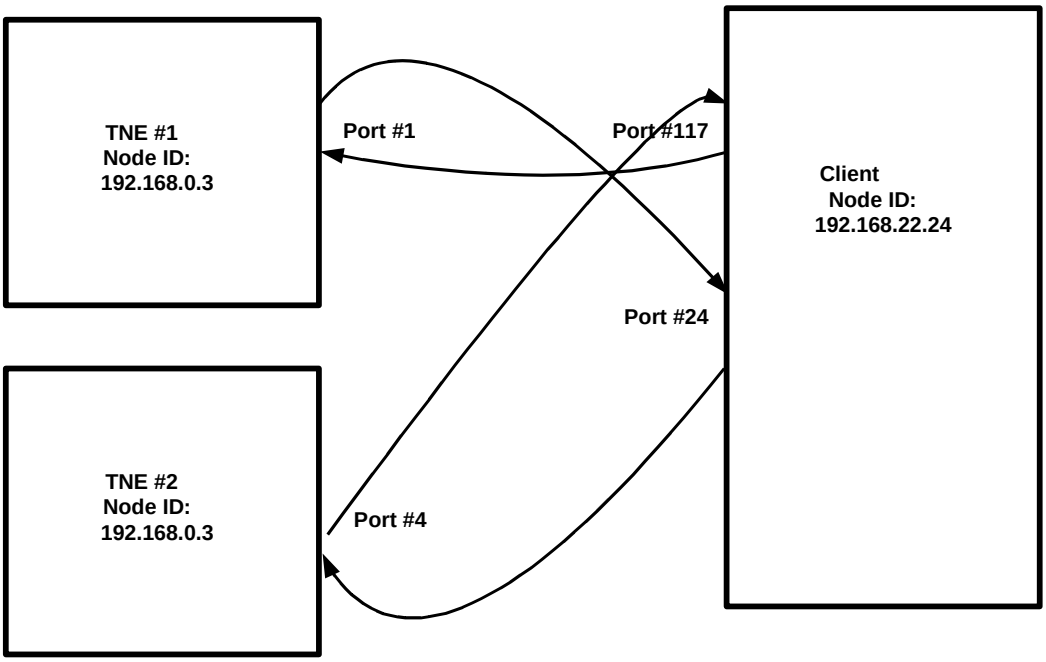


Figure 8.5: Detection of Inconsistent Physical Connectivity

In-Fiber:

In this example (Figure 8-5), inconsistency is detected by the TNEs, which note the absence of a ConfigAck (or ConfigNack) message in response to a Config message that has been sent. This is illustrated below:

Step 1: TNE #1 sends the client a Config message (Local CCID = Interface Id = Port #1, Node ID = 192.168.0.3) out of Port #1. This message is received by the client over Port #24.

Step 2: The client sends the TNE a ConfigAck message (Local CCID = Interface Id = Port #24, Node ID = 192.168.22.24, Remote CCID = Port #1, Remote Node ID = 192.168.0.3) out of Port #24. This message is received on Port #4 of TNE #2.

At this point, the TNE #1 will never receive the ConfigAck message and therefore can deduce that there is a problem with the control channel.

Out-of-Fiber:

In this example (Figure 8-5), a node detects the inconsistency when a TestStatusSuccess message is received containing an unexpected Interface ID. Note that Test messages are the ONLY messages transmitted in-fiber. All other messages are transmitted over the IPCC. This is described below, where identification information transmitted in a message is included within brackets ().

The following procedure is initiated at the TNE:

Step 1: The TNE sends the client a BeginVerify message (TNA address, 1 data link to be tested, MessageID).
Step 2: The client sends the TNE a BeginVerifyAck message (TNA address, Verify Id=200, Received MessageID)
Step 3: The TNE sends the client a Test message (Verify Id=200, Interface Id = Port #1) out of Port #1. This message is received by the client over Port #24.
Step 4: The client sends the TNE a TestStatusSuccess message (VerifyID=200, Local Interface Id = Port #24, Remote Interface Id = Port #1, MessageID)
Step 5: The TNE sends the client a TestStatusAck message (VerifyID=200, Received MessageID)
Step 6: The TNE sends the client an EndVerify message (VerifyID=200, MessageID)
Step 7: The client sends the TNE an EndVerifyAck message (VerifyID=200, Received MessageID).

The above steps are also initiated at the client as follows:

Step 1: The client sends the TNE a BeginVerify message (TNA address, 1 data link to be tested, MessageID).
Step 2: The TNE sends the client a BeginVerifyAck message (TNA address, Verify Id=350, Received MessageID)
Step 3: The client sends the TNE a Test message (Verify Id=350, Interface Id = Port #24) out of Port #24. This message is received by the TNE over Port #1.
Step 4: After waiting VerifyDeadInterval msec without receiving the Test message, the TNE sends the client a TestStatusFailure message (VerifyID=350, MessageID)
Step 5: The client sends the TNE a TestStatusAck message (VerifyID=350, Received MessageID)
Step 6: The client sends the TNE an EndVerify message (VerifyID=350, MessageID)
Step 7: The TNE sends the client and EndVerifyAck message (VerifyID=350, Received MessageID).

By now, the inconsistency in fiber connectivity is detected at both nodes.

8.5.9.3 Example 3

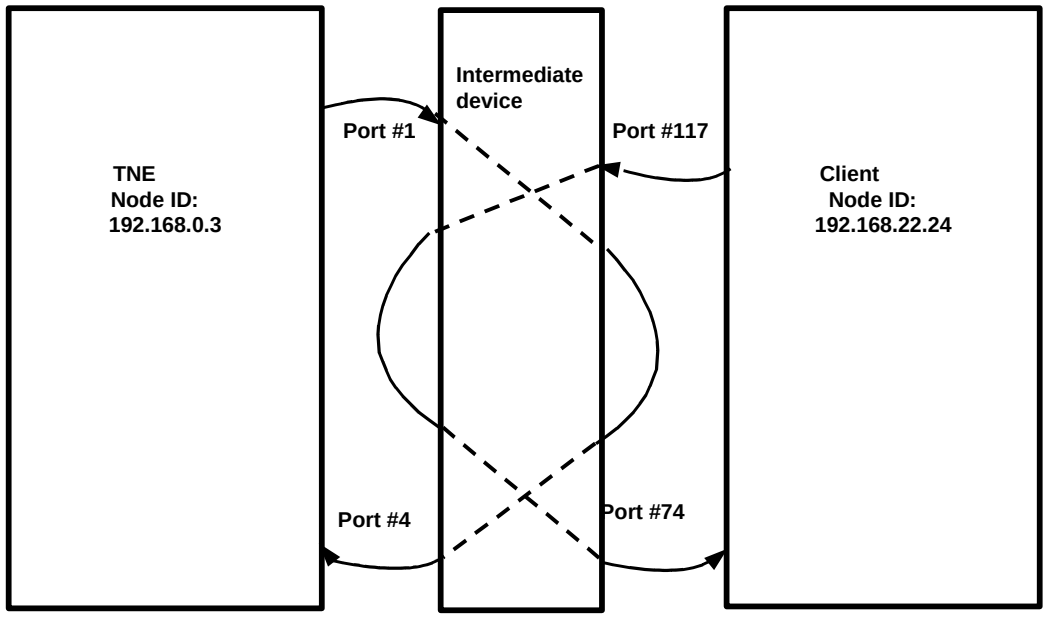


Figure 8.6: Detection of Inconsistent Physical Connectivity

In-Fiber:

In this example (Figure 8-6), the inconsistency is detected by TNEs by the reception of a Config message with the same Node ID as the receiving node. This is described below:

Step 1: TNE #1 sends the client a Config message (Local CCID = Interface Id = Port #1, Node ID = 192.168.0.3) out of Port #1. This message is received by the TNE over Port #4. Similarly,

Step 1: The client sends the TNE a Config message (Local CCID = Interface Id = Port #117, Node ID = 192.168.22.24) out of Port #117. This message is received over Port #74 by the client.

Both the TNE and client are thus aware of the misconfiguration by receiving of a Config message with the same Node ID.

Out-of-Fiber:

In this example (Figure 8-6), inconsistent wiring is detected by a node when a TestStatusFailure message is received. Note that Test messages are the ONLY messages transmitted in-fiber. All other messages are transmitted over the IPCC. This is described below, where identification information transmitted in a messages is included within brackets ().

The following procedure is initiated at the TNE:

Step 1: The TNE sends the client a BeginVerify message (TNA address, 1 data link to be tested, MessageID).
Step 2: The client sends the TNE a BeginVerifyAck message (TNA address, Verify Id=75, Received MessageID)
Step 3: The TNE sends the client a Test message (Verify Id=75, Interface Id = Port #1) out of Port #1. This message is received by the TNE over Port #4.
Step 4: After waiting VerifyDeadInterval msec without receiving the Test message, the TNE sends the client a TestStatusFailure message (VerifyID=75, MessageID).
Step 5: The TNE sends the client a TestStatusAck message (VerifyID=75, Received MessageID).
Step 6: The TNE sends the client an EndVerify message (VerifyID=75, MessageID).
Step 7: The client sends the TNE and EndVerifyAck message (VerifyID=75, Received MessageID).

These steps are also executed by the client as follows:

Step 1: The client sends the TNE a BeginVerify message (TNA address, 1 data link to be tested, MessageID).
Step 2: The TNE sends the client a BeginVerifyAck message (TNA address, Verify Id=42, Received MessageID).
Step 3: The client sends the TNE a Test message (Verify Id=42, Local Interface Id = Port #117) out of Port #117. This message is received by the client over Port #74.
Step 4: After waiting VerifyDeadInterval msec without receiving the Test message, the TNE sends the client a TestStatusFailure message (VerifyID=42, MessageID).
Step 5: The client sends the TNE a TestStatusAck message (VerifyID=42, Received MessageID).
Step 6: The client sends the TNE an EndVerify message (VerifyID=42, MessageID).
Step 7: The TNE sends the client and EndVerifyAck message (VerifyID=42, Received MessageID).

By this time inconsistent fiber connectivity is detected by both nodes.

9 Service Discovery

9.1 Overview

Service discovery is the procedure by which a UNI-C indicates the capabilities of the client device it represents to, and obtains information concerning transport network services from the UNI-N. Depending on the service invocation model used (see Figures 5-1 and 5-2), the UNI-C and UNI-N may or may not be co-located with the client device and the TNE, respectively. In the latter case, the service discovery procedure must include information about each of the clients and TNEs represented by the UNI-C and UNI-N, respectively.

The service discovery procedure is optional. Therefore, when it is not used, information that is otherwise provided by this procedure has to be manually configured in the UNI-C and the UNI-N. Service discovery runs over an IPCC (see Section 6) and it is related to neighbor discovery (see Section 8).

9.2 Overview

The following service attributes are defined (the first two are attributes of the signaling and data channels, while the last two are services of the transport network):

1. **Signaling protocols and UNI version supported**: Whether RSVP and/or LDP-based signaling is supported, and which version of UNI signaling is supported.
2. **Client port-level service attribute**: Describes the link type (i.e. SONET/SDH), the signal types, the transparency levels and the concatenation types supported over each client interface.
3. **Transparency service support**: Describes the standardized SONET/SDH transparency service(s) supported by the transport network.
4. **Network routing diversity support**: Whether the transport network supports routing node, link and Shared Risk Link Group (SRLG) diversity.

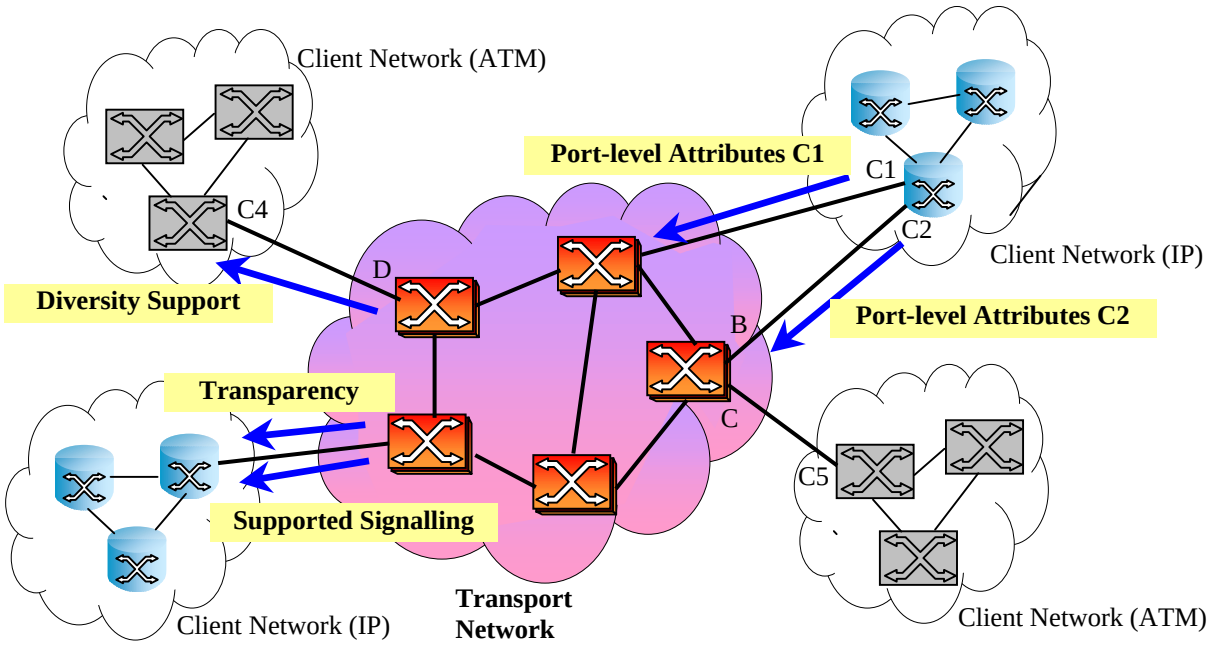


Figure 9-1: Service Discovery and Attribute Exchanges

Additionally, it is recognized that all transport networks may not support some of the service attributes such as the on-demand transparency (see section 9.3.3.1) and some network routing diversity levels. The support for these specific services, however, may be determined during connection creation.

9.3 Service Discovery Procedure

The service discovery procedure is invoked after neighbor discovery is complete, and it is run over an IPCC. This procedure consists of a sequence of message exchanges between the UNI-C and the UNI-N as shown in Figure 9-2. From the figure, it can be seen that

1. Three types of messages are used for service discovery: ServiceConfig, ServiceConfig Ack and ServiceConfig Nack. These messages are defined in sub-section 9.5.
2. Three types of ServiceConfig messages are defined. As described below, these messages contain different information elements in the LMP object format [LMP].
3. ServiceConfig messages are sent from both the UNI-C to the UNI-N, as well as from the UNI-N to the UNI-C.

The service discovery messages are exchanged over an IPCC (in-fiber or out-of-fiber). The destination IP address in these messages is set to the Node ID of the recipient, i.e. the unicast IPv4 address assigned to the UNI-C or the UNI-N. The service discovery message exchange procedure is based on LMP Config message exchange [LMP].

The following ServiceConfig objects are currently defined under UNI 1.0:

- Signaling Protocols (see Section 9.4.1)
- Client Port-Level Service Attributes (see Section 9.4.2)
- Network Transparency and TCM Monitoring (see Section 9.4.3.1)
- Network Diversity (See Section 9.4.3.2)

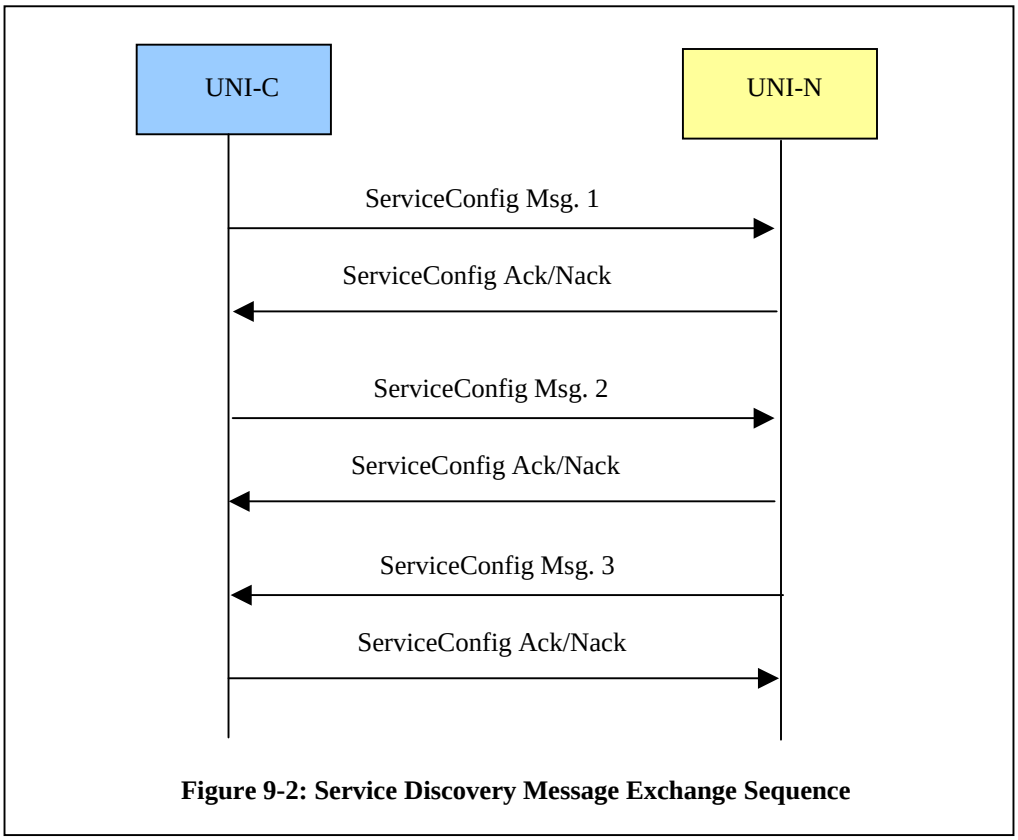


Figure 9-2: Service Discovery Message Exchange Sequence

The three ServiceConfig messages exchanged between the UNI-C and the UNI-N contain the following objects, respectively:

ServiceConfig Message #	ServiceConfig Message Object	Message Direction
1	Signaling Protocol	UNI-C to UNI-N
2	Client Port-Level Service Attributes	UNI-C to UNI-N
3	Network Transparency & TCM Monitoring, Network Diversity	UNI-N to UNI-C

Table 9-1: ServiceConfig Messages

There can be more than one exchange of ServiceConfig Message #2 and the corresponding ServiceConfig Ack/Nack. A ServiceConfig Message #3, is sent by the UNI-N to the UNI-C after the latter has received the Client Port-Level Service Attributes corresponding to all the links between the TNE it represents and the clients represented by the UNI-C.

When one of these message exchanges (Figure 9-2) is not supported the corresponding information must be provided through manual configuration. Moreover, the ServiceConfig message MUST be periodically transmitted until:

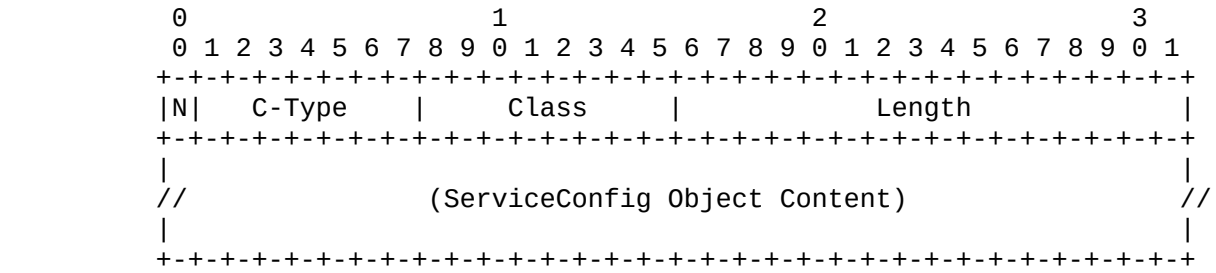
– either a ServiceConfigAck or ServiceConfigNack message is received, or
– a timeout expires and no ServiceConfigAck or ServiceConfigNack message has been received

As specified in [LMP], both the retransmission interval and the timeout period are local configuration parameters.

9.4 ServiceConfig Objects

ServiceConfig messages are built using objects. Each object is identified by its Object Class and Class-type. These objects can be either negotiable or non-negotiable (identified by the N bit in the header of the object). Negotiable objects can be used to let the devices agree on certain values. Non-negotiable Objects are used for announcement of specific values that do not need or do not allow negotiation. The ServiceConfig object content describes the supported services and attributes at the UNI.

The format of the ServiceConfig object is as follows:



N (1 bit)

The N flag indicates if the object is negotiable (N=1) or non-negotiable (N=0).

C-Type (7 bits)

Class-type within an Object Class. A C-Type value is defined for each ServiceConfig object.

Class (8 bits)

The Class indicates the Object type. The Class value for the ServiceConfig Object type is 51.

Length (16 bits)

The Length field indicates the length of the Object in bytes.

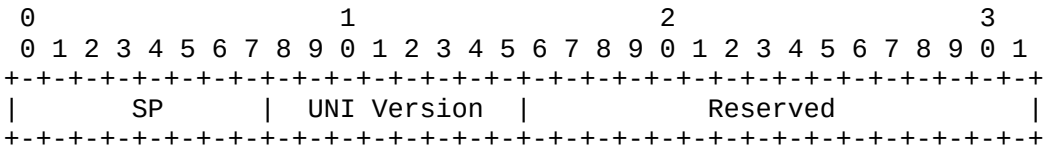
ServiceConfig Object Content

The following ServiceConfig Object Contents are currently defined:

- Signaling Protocols (see Section 9.4.1): C-Type = 1
- Client Port-Level Service Attributes (see Section 9.4.2): C-Type = 2
- Network Transparency and TCM Monitoring (see Section 9.4.3.1): C-Type = 3
- Network Diversity (See Section 9.4.3.2): C-Type = 4

9.4.1 Supported Signaling Protocols

The Supported Signaling Protocols object (Class = 51, C-Type = 1)- indicates the supported protocols at the UNI. This object is sent from the UNI-C to the UNI-N. Its format is described as follows:



This object contains the following fields.

N Bit

This bit should be set to 0 if only one signaling protocol is supported; it should be set to 1 if the protocol choice is negotiable. This bit doesn't apply to the UNI version field.

Signaling Protocols -SP (8-bits)

This field is defined as a vector of flags (bit 1 is the low order bit), which identify the UNI signaling protocols supported by the sender. The following values are currently defined:

- Flag 1 (bit 1) set to 1 if the sender supports RSVP based UNI signaling
- Flag 2 (bit 2) set to 1 if the device supports LDP based UNI signaling

UNI Version (8-bits)

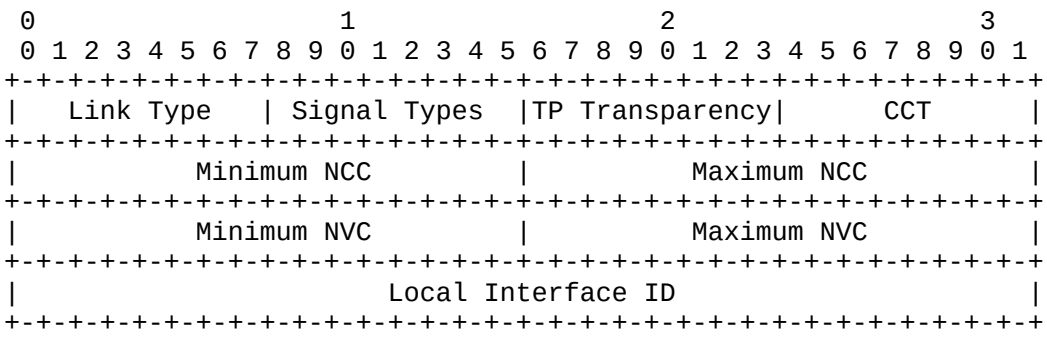
This 8-bit integer field indicates the supported version of the UNI. It is set to 0x01 to indicate UNI version 1.0.

If there is a signaling protocol or UNI version mismatch, the UNI-C (or the UNI-N) should allow a restart while requiring only the service discovery process to be re-initiated.

9.4.2 Client Port-Level Service Attributes

The Client Port-Level Service Attributes object (Class = 51, C-Type = 2) describes the encoding types, the "elementary" signal types, the transparency levels and the concatenation types supported by each client interface as well as the minimum and maximum number of components that can be concatenated (contiguously or virtually) on the indicated interface. This detailed information enables the UNI-N to dynamically determine the port-level capabilities of the clients attached to the corresponding TNE and consequently process connection requests from UNI-C(s).

This object is sent from the UNI-C to the UNI-N for each interface connecting the client node (represented by the UNI-C) to the TNE (represented by the UNI-N). Its format is as follows:



The Client Port-Level Service Attribute object contains the following fields:

N Bit

This bit must be set to 0 to indicate non-negotiability of the values.

Link Type (8 bits)

The Link type field identifies the encoding type supported on the link. This field must take one of the following values (as defined in [GMPLS SIG]):

Value	Link Encoding Type
5	SDH ITU-T G.707
6	SONET ANSI T1.105

Signal Types (8 bits)

This field indicates all the possible signals supported by a given component interface for the corresponding non-concatenated signal. This parameter is interpreted in relation to the Link Encoding Type (Link Type field). In fact these values indicate the most usual SONET/SDH interface types currently available.

Permitted values for SDH links are as follows:

Value	Signal Type
5	VC-3
6	VC-4
7	STM-0
8	STM-1
9	STM-4
10	STM-16
11	STM-64
12	STM-256

Permitted values for SONET links are the following:

Value	Signal Type
5	STS-1 SPE
6	STS-3c SPE
7	STS-1

8	STS-3
9	STS-12
10	STS-48
11	STS-192
12	STS-768

Note: a dedicated signal type is assigned to a SONET STS-3c SPE instead of coding it as a contiguous concatenation of three STS-1 SPE.

TP Transparency (8 bits)

This field is defined as a vector of flags (bit 1 is the low order bit) which indicates the Termination Point (TP) Transparency level(s) supported by the client component interface:

- Flag 1 (bit 1) set to 1 if Path/VC Overhead Transparency (default)
- Flag 2 (bit 2) set to 1 if Line/MS Overhead Transparency
- Flag 3 (bit 3) set to 1 if Section/RS Overhead Transparency
- Flag 4 to Flag 8 (bit 4 to 8) set to 0 and reserved for future use

Note: This field explicitly specifies the Transparency "support" from the client side which is completely orthogonal to the Transparency level supported by the transport network as defined in Section 9.4.3.1.

Contiguous Concatenation Types – CCT (8 bits)

This field is defined as a vector of flags (bit 1 is the low order bit) which indicates the contiguous concatenation type(s) supported by the client interface:

- Flag 1 (bit 1) set to 1 if contiguous concatenation is supported as defined in T1.105/G.707
- Flag 2 to Flag 8 (bit 2 to 8) set to 0 and reserved for future use

When no contiguous concatenation is supported the CCT field = 0 (default value).

Minimum Number of Contiguously concatenated Components – Minimum NCC (16 bits)

This field (used jointly with the CCT field) specifies the minimum number of contiguously concatenated components applicable to a VC-4 or STS-3c SPE (as detailed in Section 4) signal. This number depends on the hardware interface capabilities.

When the client interface does not support contiguous concatenation, then Minimum NCC must be set to zero when sent and ignored when received. When the client interface supports contiguous concatenation, then the Minimum NCC value must be at least equal to 1.

Maximum Number of Contiguously concatenated Components – Maximum NCC (16 bits)

This field (used jointly with the CCT field) specifies the maximum number of contiguously concatenated components applicable to a VC-4 or STS-3c SPE (as detailed in Section 4) signal. This number depends on the hardware interface capabilities. When the client interface does not support contiguous concatenation, then Maximum NCC must be set to zero when sent and ignored when received. When the client interface supports contiguous concatenation, then the Maximum NCC value must be at least equal to 1.

Minimum Number of Virtually concatenated Components – Minimum NVC (16 bits)

This field specifies the minimum number of virtually concatenated components applicable to a VC-3/VC-4 or STS-1 SPE/STS-3c SPE (as detailed in Section 4). This number depends on the hardware interface capabilities.

When the client interface does not support virtual concatenation, then Minimum NVC must be set to zero when sent and ignored when received. When the client interface supports virtual concatenation, then the Minimum NVC value must be at least equal to 1.

Maximum Number of Virtually concatenated Components – Maximum NVC (16 bits)

This field specifies the maximum number of virtually concatenated components applicable to a VC-3/VC-4 or STS-1 SPE/STS-3c SPE (as detailed in Section 4). This number depends on the hardware interface capabilities.

When the client interface does not support virtual concatenation, then Maximum NVC must be set to zero when sent and ignored when received. When the client interface supports virtual concatenation, then the Maximum NVC value must be at least equal to 1.

Local Interface ID (32 bits)

The encoded Local Interface ID of the client interface whose capabilities this object refers to.

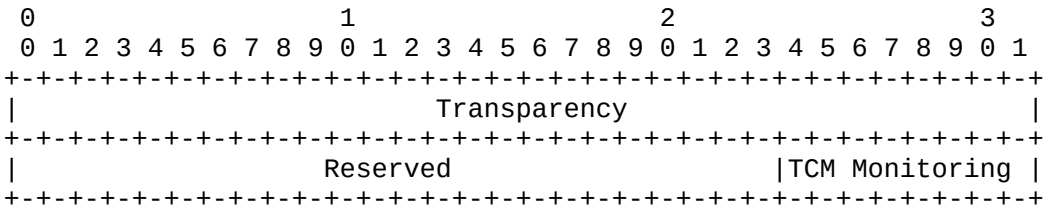
Tables 4-1 and 4-2 (see Section 4) summarize the valid combinations of the signal type, the transparency levels and the concatenation types supported by SDH and SONET client interfaces, respectively.

9.4.3 Network Service Attributes

The following network services attributes are currently defined:

9.4.3.1 Network Transparency and TCM Monitoring

The Network Transparency Support and TCM Monitoring object (Class = 51, C-Type = 3) indicates the transparency level supported within the transport network. Its format is defined as:



This object contains the following fields.

N Bit

This bit must be set to 0 to indicate non-negotiability of the values.

Transparency (32 bits)

The Transparency field is a 32-bits vector of flag (see [GMPLS SONET]) indicating the transparency service level supported within the transport network:

- Flag 1 (bit 1) set to 1 if Standard SOH/RSOH transparency as defined in T1.105/G.707
- Flag 2 (bit 2) set to 1 if Standard LOH/MSOH transparency as defined in T1.105/G.707

- Flag 3 to 32 (bit 3 to 32) set to 0 and reserved for future use

Note: Standard Path Overhead (POH) bytes transparency is defined as the default behavior (path overhead bytes are not modified) and obtained by setting any flag to 0.

Bit 1 is the low order bit. Other flags (bits 3 to 32) are reserved for future use. They should be set to zero when sent, and must be ignored when received. A flag is set to one to indicate that the corresponding transparency type is requested.

Standard SOH/RSOH transparency (i.e. means that the transport network should not modify Section/Regenerator Section, Line/Multiplex Section and Path Overhead bytes.

Standard LOH/MSOH transparency means that the transport network should not modify Line/Multiplex Section and Path Overhead bytes.

Standard POH transparency means that the transport network should not modify Path Overhead bytes (default behavior).

Note: transparency service is only applicable when using the following Signal Types (ST's): STM-0, STM-1, STM-4, STM-16, STM-64, STM-256 for SDH connections, STS-1, STS-3, STS-12, STS-48, STS-192, and STS-768 for SONET connections. At least one transparency type must be specified when requesting such a signal type into the connection setup request (see Sections 10, 11 and 12).

Reserved (24 bits)

This field is reserved for future use. These bits must be set to zero when 0 and ignored when received

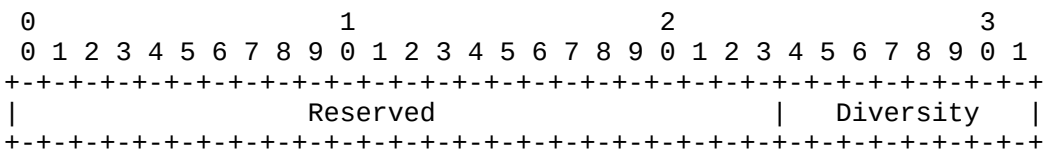
TCM Monitoring (8 bits)

The TCM Monitoring field is defined as vector of flags (bit 1 is the low order bit). Within the scope of the UNI 1.0, the following values are currently considered:

- Flag 1 (bit 1) set to 1 indicates the “transparent” support of Tandem Connection Monitoring (TCM) at the UNI.
- Flag 2 to 8 (bits 2 to 8) are set to 0 and reserved for future use.

9.4.3.2 Network Diversity

The Network Diversity object (Class = 51, C-Type = 4) indicates the network routing diversity level provided by the transport network. This object is sent from the UNI-N to the UNI-C. Its format is as follows:



The object contains the following fields:

N Bit

This bit must be set to 0 to indicate non-negotiability of the values.

Reserved (24 bits)

This field is reserved for future use. These bits must be set to zero when sent and ignored when received

Diversity (8 bits)

The Diversity field is defined as vector of flags (bit 1 is the low order bit) which lists all the "diversity types" supported by the UNI-N (see Section 10). In the scope of the UNI 1.0, diversity is only valid within a single routing domain.

The currently defined Diversity flags are the following:

- Flag 1 (bit 1) set to 1 when Node diversity is supported
- Flag 2 (bit 2) set to 1 when Link diversity is supported
- Flag 3 (bit 3) set to 1 when SRLG diversity is supported
- Flag 4 to 8 (bit 4 to 8) set to 0 and reserved for future use

Default behavior (no diversity supported by the transport network) is obtained by setting all flags to 0.

9.5 Service Discovery LMP Messages

The ServiceConfig (MsgType = 50), ServiceConfigAck (MsgType = 51) and ServiceConfigNack messages (MsgType = 52) are defined as LMP messages preceded by the LMP common header (see [LMP]).

9.5.1 The ServiceConfig Message

The ServiceConfig message (MsgType = 50) is an LMP message sent in an IP packet. This message has the following format:

<ServiceConfig Message> ::= <Common Header> <Node ID> <Message ID> <ServiceConfig>, where:

<Node ID> is the LOCAL_NODE_ID Object (Class = 3, C-Type = 1) as defined in [LMP]
<Message ID> is the Type 1 MessageId Object (Class = 9, C-Type = 1) as defined in [LMP]
<ServiceConfig> is the ServiceConfig Object Class (Class = 51) as defined in Section 9.5.2

9.5.2 ServiceConfig Object

The ServiceConfig object (Class = 51) indicates the supported services at the UNI (See section 9.4 for more details) . The ServiceConfig Object content was defined in section 9.3 for the support of Signaling protocol (section 9.4.1), Client port level service attributes (section 9.4.2), Transparency and TCM Monitoring (section 9.4.3.1) and Network routing diversity (section 9.4.3.2).

9.5.3 The ServiceConfigAck Message

The ServiceConfigAck message (MsgType = 51) indicates that ALL the parameters (both negotiable and non-negotiable) received in the ServiceConfig message are acceptable to the receiving node. The ServiceConfigAck message is an LMP message sent in an IP packet, with the IP destination address set to the Node ID received in the corresponding ServiceConfig message. The ServiceConfigAck message has the following format:

<ServiceConfigAck Message> ::= <Common Header> <Node ID> <Message ID> , where:

<Node ID> is the LOCAL_NODE_ID Object (Class = 3, C-Type = 1) as defined in [LMP]
<Message ID> is the Type 2 MessageId Object (Class = 10, C-Type = 1) as defined in [LMP]

9.5.4 ServiceConfigNack Message

The ServiceConfigNack message (MsgType = 52) is used to acknowledge receipt of the ServiceConfig message, to indicate which (if any) non-negotiable parameters are unacceptable, and to propose alternate values for the negotiable parameters. The ServiceConfigNack message an LMP message sent in an IP packet, with the IP destination address set to the Node ID received in the corresponding ServiceConfig message. The ServiceConfigNack message has the following format:

<ServiceConfigNack Message> ::= <Common Header> <Node ID> <Message ID> <ServiceConfig>, where:

<Node ID> is the LOCAL_NODE_ID Object (Class = 3, C-Type = 1) as defined in [LMP]
<Message ID> is the Type 2 MessageId Object (Class = 10, C-Type = 1) as defined in [LMP]
<ServiceConfig> is the ServiceConfig Object Class (Class = 51) as defined in Section 9.5.2

9.5.5 ServiceConfig Finite State Machine (FSM) at UNI-C and UNI-N

The ServiceConfig FSM defines the states and logic of operation of the service discovery procedure. As noted in Figure 9-2, the service discovery procedure begins when the UNI-C sends a ServiceConfig message #1. With a positive response from the UNI-N, the UNI-C sends one or more ServiceConfig messages #2 (A negative response results in the failure of service discovery). Finally, when the UNI-N has received all the ServiceConfig messages #2, it sends a ServiceConfig message #3. This procedure is captured in the FSMs, which also takes into account the exception states.

9.5.5.1 Data Structures

The following data structures are maintained at the UNI-C and UNI-N:

Retransmission Timer, *r*: This configurable timer is set by a node sending a ServiceConfig message. The timer is reset if an Ack or Nack is received for the message. The expiry of this timer results in the retransmission of the Service Config message. The default value of this timer is 1 sec.

Number of Retransmissions, *n*: This configurable parameter indicates the maximum number of retransmissions of ServiceConfig messages allowed. The default value of this number is 3. A node considers the service discovery procedure to have failed if a response is not received from the peer after *n* retransmission attempts.

Completion Timer: This configurable timer is set by the UNI-C when it has no more ServiceConfig messages to send to the UNI-N, i.e., all ServiceConfig messages (#1 and #2) have been sent, and for each ServiceConfig message sent,

- an Ack has been received,
- a Nack has been received and no alternate values for parameters can be proposed, or
- no response has been received after *n* retransmissions.

The UNI-C expects to receive a ServiceConfig message from the UNI-N before this timer expires. The UNI-C stops waiting for such a message from the UNI-N if the timer expires.

This timer is set by the UNI-N when it has successfully received ServiceConfig message #1. The UNI-N expects to receive all ServiceConfig messages #2 from the UNI-C before this timer expires. The default value of this timer is 3 minutes.

ServiceConfig Success Flag: This flag is set when the node executing the FSM determines that service discovery has been successfully completed. Intial value is FALSE.

9.5.5.2 ServiceConfig States

The ServiceConfig FSM can be in one of the states described below. Every state corresponds to a certain condition of the procedure and is usually associated with a specific type of ServiceConfig message that is transmitted to the peer UNI agent.

CCUp	Represents the active state of the logical Control Channel (CC). In this state, at least one IPCC is available and LMP ServiceConfig messages can be exchanged over it. In this state, the UNI-C sends ServiceConfig Message #1 and UNI-N receives the same (As such, the ServiceConfig Message#1 can be considered as a request to begin service discovery).
CCDown	Represents the down state of the Control Channel, i.e., no IPCC is available to the peer entity. In this state, LMP ServiceConfig messages can not be exchanged.
ServiceConfigSnd	In this state, the node (UNI-N or UNI-C) periodically sends a ServiceConfig message, and is expecting the other side to reply with either a ServiceConfigAck or ServiceConfigNack message.
ServiceConfigRcv	In this state, the node (UNI-N or UNI-C) is waiting for acceptable ServiceConfig message from the remote side.

9.5.5.3 ServiceConfig Events

The service discovery procedure is described in terms of FSM states and events. The underlying protocols and software modules generate ServiceConfig events. Each event has its number and a given symbolic name. The ServiceConfig events are as follows:

Event #	Event Symbolic Name	Description
1	EvCCUp	This is an externally triggered event bringing the logical CC to the active state. It implicitly indicates that the Service Discovery procedure negotiation can begin. This event, for instance, can follow a successful Neighbor Discovery procedure.
2	EvCCDown	This event is generated when there is an indication that no IPCCs are available. Therefore, this is an external event.
3	EvServiceConfigOK	This event indicates a ServiceConfigAck message has been received from the peer UNI entity.
4	EvServiceConfigErr	This event indicates a ServiceConfigNack message has been received from the peer UNI entity.
5	EvNewServiceConfig	New ServiceConfig message was received from the peer UNI entity.
6	EvAllServiceConfigSent	This event indicates the completion of sending ServiceConfig messages at the UNI-C or UNI-N. That is, it indicates that all ServiceConfig messages have been sent. Furthermore, for each ServiceConfig message sent, • an Ack has been received, • or a Nack has been received and no alternate values for parameters can be proposed
7	EvAllServiceConfigRcv	This event indicates that all ServiceConfig messages have been received at UNI-C or UNI-N. Furthermore, for each ServiceConfig message received, • an Ack has been sent,

		• a Nack has been sent with no alternate values for rejected parameters, or • the completion timer has expired.
8	EvNoServiceConfigSent	This event indicates that no response (ServiceConfigAck or ServiceConfigNack) has been received after *n* retransmissions.
9	EvServiceConfig1Rcv	This event indicates that the UNI-N has received Service Config Message #1 and has sent a Service Config Ack in return.

9.5.5.4 ServiceConfig FSM Description at the UNI-C

Figure 9-3 illustrates the ServiceConfig FSM at the UNI-C. The following actions are performed during state transitions (Note: The actions do not cover retransmissions of messages)

	States			
Events	*CCdown*	*CCup*	*ServiceConfigSnd*	*ServiceConfigRcv*
1	Action 1	No Action	No Action	No Action
2	No Action	Action 2	Action 2	Action 2
3	Error	No Action	Action 6	No Action
4	Error	Action 3	Action 7	No Action
5	Error	Action 4	No Action	Action 8
6	Error	No Action	No Action	No Action
7	Error	No Action	Error	Action 8
8	Error	Action 5	Action 5	Error

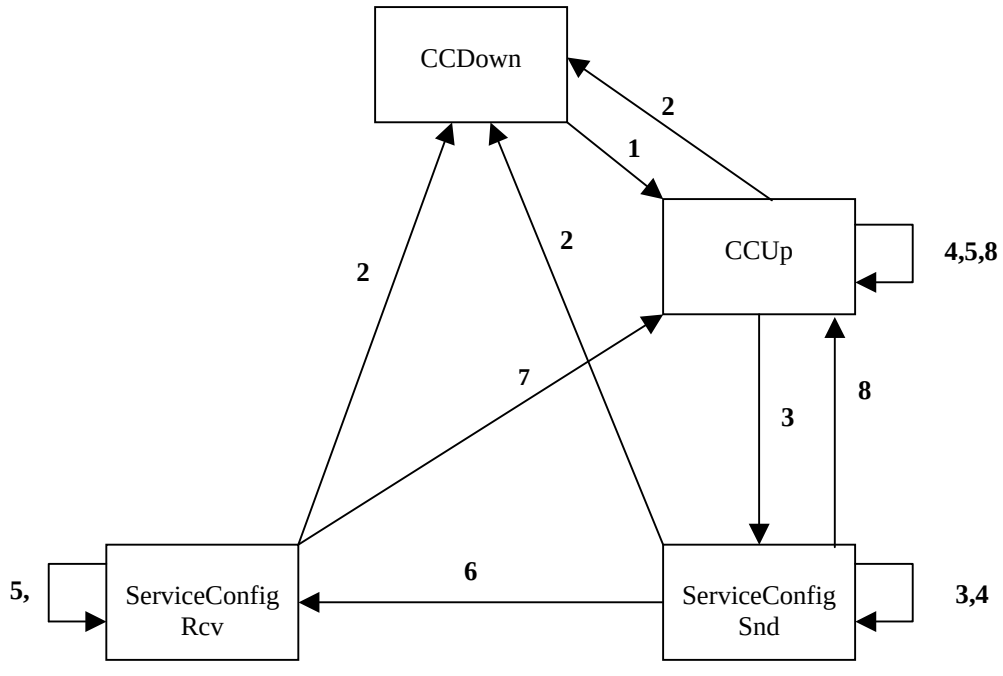


Figure 9-3: Service Config FSM at UNI-C

Action 1: ServiceConfig message #1 is sent to UNI-N.

Action 2: All data structures are reset to initial values and the availability of control channel is awaited.

Action 3: ServiceConfig Message #1 has only non-negotiable parameters. Thus, the receipt of a Nack implies that service discovery cannot proceed. All data structures are reset to initial values and Service Discovery is attempted after a pre-configured delay.

Action 4:

- If the ServiceConfig Success Flag is set to TRUE, and the message from the UNI-N has the same Network monitoring, TCM and Diversity information as recorded before (see Action 8) then a ServiceConfigAck is sent to the UNI-N.
- Otherwise, a new ServiceConfig message has been received and it should be ignored.

Action 5: All data structures are reset to initial values. ServiceConfig Message #1 is sent again after a pre-configured waiting time.

Action 6: If ServiceConfigAck has been received pertaining to the client port level service attributes of all the client links to the TNE, Event 6, EvServiceConfigSent, is generated. The completion timer is also started.

Action 7:

- If alternate parameters can be proposed then a new ServiceConfig message with these parameters are sent.
- Otherwise, if there are no more ServiceConfig messages remain to be sent, Event 6, EvServiceConfigSent, is generated. The completion timer is also started.

Action 8:

- If parameters received in the Service Config message from UNI-N are acceptable then the parameters are recorded and a Service Config Ack to UNI-N is sent. The Service Config Success Flag is set to TRUE. Event 7, EvAllServiceConfigRcv, is generated.
- If the parameters received are unacceptable then alternate values for negotiable parameters are proposed and sent in a Nack message to UNI-N.
- If no alternate values are acceptable then a Nack message to UNI-N is sent and the Service Config Success Flag is set to FALSE Event 7, EvAllServiceConfigRcv, is generated.

Error: Indicates protocol implementation error. The <state, event> combination should not occur.

9.5.5.5 Service Config FSM Description at the UNI-N

Figure 9-4 illustrates the ServiceConfig FSM at the UNI-N. The following actions are performed during state transitions (Note: The actions do not cover retransmission of messages):

	States			
Events	*CCdown*	*CCup*	*ServiceConfigSnd*	*ServiceConfigRcv*
1	No Action	No Action	No Action	No Action
2	No Action	Action 1	Action 1	Action 1
3	Error	Error	Action 6	Error
4	Error	Error	Action 7	Error
5	Error	Action 2	No Action	Action 4
6	Error	Error	No Action	Error
7	Error	Error	No Action	Action 5
8	Error	Error	Action 8	Error
9	Error	Action 3	No Action	No Action

Action 1: All data structures are reset to initial values and for the availability of control channel is awaited..

Action 2:

- If the parameters in the ServiceConfig Message # 1 are acceptable then a ServiceConfigAck is sent to the UNI-C. Event 9, EvServiceConfig1Rcv, is generated.
- If the parameters are not acceptable then a ServiceConfigNack is sent to the UNI-C.

Action 3: The completion timer is started.

Action 4:

- If the client port-level service attributes received in the ServiceConfig message from UNI-C are acceptable then the parameters are recorded and a Service Config Ack to UNI-C is sent. If Service Config messages with client port level service attributes have been received for all the links to the client then Event 7, EvServiceConfigRcv, is generated.
- If the parameters received are unacceptable then alternate values for negotiable parameters are proposed and sent in a Nack message to UNI-C.
- If no alternate values are acceptable then a Nack message to UNI-C is sent. If Service Config messages with client port level service attributes have been received for all the links to the client then Event 7, EvServiceConfigRcv, is generated

Action 5: A ServiceConfig message is sent to the UNI-C with the Network transparency and TCM monitoring, and the Diversity objects.

Action 6: Event 6, EvServiceConfigSent event is generated. The Service Config Success Flag is set to

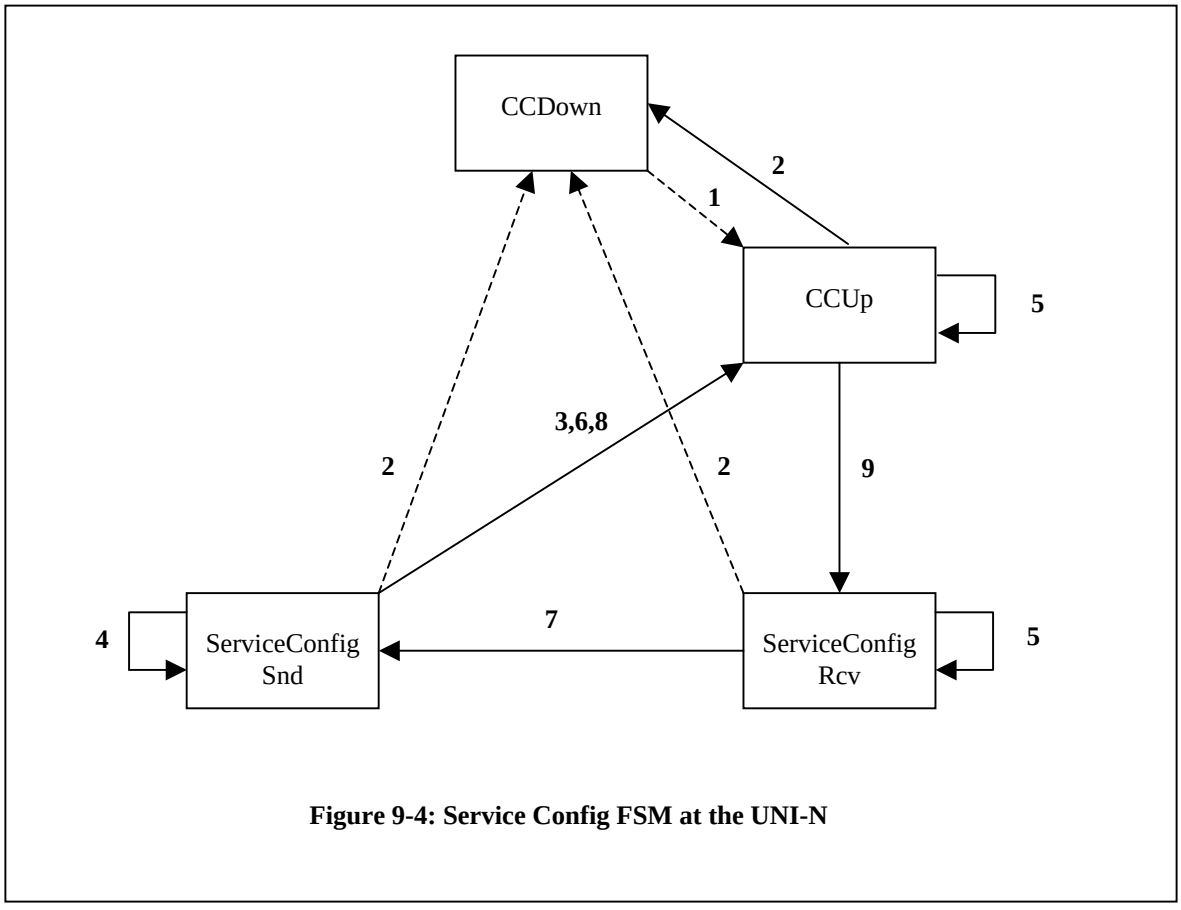


Figure 9-4: Service Config FSM at the UNI-N

TRUE.

Action 7:

- If alternate parameters can be proposed then a new ServiceConfig message with these parameters are sent.
- Otherwise, Event 6, EvServiceConfigSent, is generated, and the Service Config Success Flag is set to FALSE Data structures are reset to initial values.

Action 8: The Service Config Success Flag is set to FALSE. Data structures are reset to initial values.

Error: Indicates protocol implementation error. The <state, event> combination should not occur.

10 UNI Abstract Messages

The UNI signaling messages are described in this section. These messages are denoted "abstract" since the actual realization depends on the signaling protocol used. Sections 11 and 12 describe LDP and RSVP-TE signaling messages corresponding to the abstract messages defined here. In the following description, the terms "initiating UNI-C" and "terminating UNI-C" are used to identify the entities at the two ends of a connection that initiate and terminate a given signaling action. The initiating UNI-C for a given signaling action need not necessarily be the UNI-C that originally requested the connection. For instance, the initiating UNI-C for a status enquiry or deletion may be different from the initiating UNI-C for the original connection request. Note that for proxy signaling, the same proxy UNI-C could be used for both end points of the connection (see Section 5).

The following abstract messages (Table 10-1) are currently defined:

Message No.	Abstract Message Description	Message Direction
1	Connection Create Request	UNI-C→UNI-N & UNI-N→UNI-C
2	Connection Create Response	UNI-N→UNI-C & UNI-C→UNI-N
3	Connection Create Confirmation	UNI-C→UNI-N & UNI-N→UNI-C
4	Connection Delete Request	UNI-C→UNI-N & UNI-N→UNI-C
5	Connection Delete Response	UNI-N→UNI-C & UNI-C→UNI-N
6	Connection Status Enquiry	UNI-C→UNI-N & UNI-N→UNI-C
7	Connection Status Response	UNI-N→UNI-C & UNI-N→UNI-C
8	Notification	UNI-N→UNI-C

Table 10-1: UNI Messages

A list of UNI attributes is associated with each UNI message. Some of these attributes are required (mandatory) for a given signaling message, while others are optional. These are described in the following sections. In addition, a network may not support all of the requested attributes. In such cases, the network shall send appropriate indications to the client in response messages. The encoding of these attributes is not defined in this section since this too depends on the particular signaling protocol used. Refer to Sections 11 and 12 for attribute encoding under the LDP and RSVP-TE signaling protocols, respectively.

The manner in which the UNI abstract messages are mapped to actions *within* the transport network, and the signaling protocol used within the transport network to realize the actions are outside the scope of this specification.

The UNI abstract messages are described in detail below.

10.1 Connection Create Request

The connection create request is used by a client to request a connection between specified source and destination clients. The connection request additionally defines the set of attributes that describe the service requirements for the connection.

The connection create request is sent from
1. the initiating UNI-C to UNI-N to request the creation of a connection;
2. the UNI-N to the terminating UNI-C to indicate an incoming connection request.

The mandatory (M) and optional (O) attributes carried in this message are shown below. Not all of the attributes are applicable in each of the cases 1 (UNI-C->UNI-N) and 2 (UNI-N->UNI-C) above. The

"Applicability" column indicates the case in which the corresponding parameter is applicable. The section in which each attribute is described is also indicated.

Attributes	Applicability	Reference
Source TNA Address (M)	Cases 1 and 2	Section 10.9.1.1
Source Logical Port Identifier (M)	Case 1	Section 10.9.1.2
Source Generalized Label (O)	Case 1	Section 10.9.1.3
Destination TNA Address (M)	Cases 1 and 2	Section 10.9.1.1
Destination Logical Port Identifier (O)	Cases 1 and 2	Section 10.9.1.2
Destination Generalized Label (O)	Cases 1 and 2	Section 10.9.1.3
Local connection ID (M)	Cases 1 and 2	Section 10.9.1.4
Contract ID (O)	Case 1	Section 10.9.4.1
Encoding Type (M)	Cases 1 and 2	Section 10.9.2.1
SONET/SDH traffic parameters (M)	Cases 1 and 2	Section 10.9.2.2
Directionality (O)	Cases 1 and 2	Section 10.9.2.3
Generalized Payload Identifier (O)	Cases 1 and 2	Section 10.9.2.4
Service Level (O)	Cases 1 and 2	Section 10.9.2.5
Diversity (O)	Cases 1 and 2	Section 10.9.3.1

10.2 Connection Create Response

The connection create response is used to acknowledge the establishment of the connection to the UNI-C that initiated the connection request. The corresponding client may then start transmission of data on the established connection.

The connection create response is sent from
1. the terminating UNI-C to UNI-N to accept or reject an incoming connection create request;
2. the UNI-N to the initiating UNI-C to indicate the successful creation of (or failure to create) a connection requested previously.

The attributes carried in this message are listed below, together with the applicability in each of the cases above:

Attributes	Applicability	Reference
Source Logical Port Identifier (O)	Case 2	Section 10.9.1.2
Source Generalized Label (O)	Case 2	Section 10.9.1.3
Destination Logical Port Identifier (O)	Cases 1 and 2	Section 10.9.1.2
Destination Generalized Label (O)	Cases 1 and 2	Section 10.9.1.3
Local Connection ID (M)	Cases 1 and 2	Section 10.9.1.4
Connection Status (M)	Cases 1 and 2	Section 10.9.5.1
Error Code (O)	Cases 1 and 2	Section 10.9.5.2

10.3 Connection Create Confirmation

The connection create confirmation is used to acknowledge the establishment of the connection to the UNI-C that terminated the connection create request. The corresponding client may then start transmission of data on the established connection.

The connection create confirmation is sent from
1. the initiating UNI-C to UNI-N to acknowledge completion of the connection establishment;

2. the UNI-N to the terminating UNI-C to indicate that the connection has been successfully created and that the corresponding client may start transmitting data over the connection.

The attributes carried in this message are listed below, together with the applicability in each of the cases above:

Attributes	Applicability	Reference
Source Logical Port Identifier (O)	Case 1	Section 10.9.1.2
Source Generalized Label (O)	Case 1	Section 10.9.1.3
Destination Logical Port Identifier (O)	Cases 1 and 2	Section 10.9.1.2
Destination Generalized Label (O)	Cases 1 and 2	Section **Error!**
Local Connection ID (M)	Cases 1 and 2	Section 10.9.1.4
Connection Status (M)	Cases 1 and 2	Section 10.9.5.1
Error Code (O)	Cases 1 and 2	Section 10.9.5.2

10.4 Connection Delete Request

The connection delete request is used to initiate the deletion of a connection. If the connection deletion is being initiated by a UNI-C, then this UNI-C should maintain the connection control state and the corresponding client should maintain the data plane until after the connection deletion has been acknowledged. This avoids alarms being generated by the client at the other end of the connection. The UNI-C terminating the connection deletion request may delete the connection state upon receipt of the connection delete request (the corresponding client may delete the data plane state).
The network may also wish to delete a connection (a forced deletion) due to scenarios such as:

- Internal network failures, which force the network to terminate connections
- A deletion response is not received by the UNI-N within a pre-defined timeout

When the network initiates a connection delete request, a connection delete response is not required and the UNI-N may terminate all connection control state and the data path as the connection delete request is initiated.

The connection delete request is sent from
1. the initiating UNI-C to UNI-N to delete a connection;
2. the UNI-N to the terminating UNI-C to indicate deletion by far end;
3. the UNI-N to a UNI-C to indicate the deletion of a connection by the network.

The attribute carried in this message and its applicability are listed below:

Attributes	Applicability	Reference
Local Connection ID (M)	All cases	Section 10.9.1.4

Note that connection deletion can be initiated by either ends of a connection, not just the end that originally initiated the connection request.

10.5 Connection Delete Response

The connection delete response message signals the completion of the connection deletion procedure. The UNI-C that initiated the connection deletion may delete connection control state upon receipt of the deletion response (the corresponding client may delete the data plane state of the connection).

The connection delete response is sent from
1. the terminating UNI-C to UNI-N to acknowledge an incoming connection delete request;
2. the UNI-N to the initiating UNI-C to indicate the successful deletion of the connection as requested.

The attributes carried in this message and their applicability are listed below:

Attributes	Applicability	Reference
Local Connection ID (M)	Cases 1 and 2	Section 10.9.1.4
Connection Status (M)	Cases 1 and 2	Section 10.9.5.1
Error Code (O)	Cases 1 and 2	Section 10.9.5.2

10.6 Connection Status Enquiry

The connection status enquiry is used to query the state and attributes of a given connection.

The connection status enquiry is sent from
1. the initiating UNI-C to UNI-N to enquire about the status and/or the attributes of one or more connections owned by the UNI-C;
2. the UNI-N to either UNI-C to enquire about the status and/or the attributes of one or more connections owned by the UNI-C.

The attributes carried in this message and their applicability are listed below:

Attribute	Applicability	Reference
TNA Address (O)	Cases 1 and 2	Section 10.9.1.1
Zero or more local Connection IDs (M)	Cases 1 and 2	Section 10.9.1.4

Multiple local connection IDs may be included in a single connection status enquiry. If a TNA Address but no local connection ID is specified then the attributes of all connections owned by the TNA is returned. Otherwise, the status of the indicated connection(s) is returned.

Note that a connection status enquiry can be initiated by the UNI-C at either end of a connection, not just by the UNI-C that initiated the original connection request.

10.7 Connection Status Response

The connection status response returns the status of the specified connection and the associated attributes.

The connection status response message is sent from
1. the UNI-N to the UNI-C to indicate the status of connection attributes as requested previously;
2. the UNI-C to UNI-N to indicate the status of connection attributes as requested previously.

The attributes carried in this message and their applicability are listed below. If the status of multiple connections is being returned, the contents shown must be repeated for each connection. The “Local Logical Port Identifier” and the “Local Generalized Label” indicate the port and channel information at the side (source or destination) where the Connection Status Response message is received. In other words, this message does not include the remote port and channel information.

Attributes	Applicability	Reference
Local Connection ID (M)	Cases 1 and 2	Section 10.9.1.4
Connection Status (M)	Cases 1 and 2	Section 10.9.5.1
Source TNA Address (O)	Cases 1 and 2	Section 10.9.1.1
Destination TNA Address (O)	Cases 1 and 2	Section 10.9.1.1
Local Logical Port Identifier (O)	Cases 1 and 2	Section 10.9.1.2
Local Generalized Label (O)	Cases 1 and 2	Section 10.9.1.3

Contract ID (O)	Case 1	Section 10.9.4.1
Encoding Type (O)	Cases 1 and 2	Section 10.9.2.1
SONET/SDH traffic parameters (O)	Cases 1 and 2	Section 10.9.2.2
Directionality (O)	Cases 1 and 2	Section 10.9.2.3
Generalized Payload Identifier (O)	Cases 1 and 2	Section 10.9.2.4
Service Level (O)	Cases 1 and 2	Section 10.9.2.5
Diversity (O)	Cases 1 and 2	Section 10.9.3.1
Error Code (O)	Cases 1 and 2	Section 10.9.5.2

10.8 Notification

The notification message is sent autonomously by a UNI-N to either UNI-C to indicate a change in the status of the connection (e.g., unrestorable connection failure). The attributes carried in this message are listed below:

Attributes	Applicability	Reference
Local Connection ID (M)	UNI-N→UNI-C	Section 10.9.1.4
Connection Status (M)	UNI-N→UNI-C	Section 10.9.5.1
Error Code (M)	UNI-N→UNI-C	Section 10.9.5.2

10.9 Description of Attributes

The attributes are classified into identification-related, service-related, routing-related, policy-related and miscellaneous. The encoding of these attributes would depend on the signaling protocol used and are described in Sections 11 and 12. In this section, the attributes are described in a general manner.

10.9.1 Identification-Related Attributes

10.9.1.1 Transport Network Assigned (TNA) Address

Transport Network Assigned (TNA) Addresses are assigned by the service provider to one or more data links (see Section 7). For UNI 1.0, the TNA may be an IPv4, IPv6 or an NSAP address.

10.9.1.2 Logical Port Identifier

The Logical Port Identifier is an index that indicates a client or TNE port (Section 7).

10.9.1.3 Generalized Label

The Generalized Label is a structure that indicates a multiplexed channel within the signal carried over the specified data link. For instance, this identifier could indicate an STS-48 channel within an STS-192 link. In general, this structure can indicate several levels of multiplexing. The encoding of this structure is left up to the signaling protocol specification (see Sections 11 and 12).

10.9.1.4 Local Connection ID

The Local Connection ID identifies a connection locally at a UNI. The local connection ID must be unique over a given UNI, but is not required to be unique across the entire network. The local connection ID is assigned by the UNI-C that initiates a connection create request at the initiating end, and by the UNI-N at the terminating end. Thus, a local connection ID is carried in each connection create request message sent from the initiating UNI-C to the UNI-N, and in each connection create message sent from the UNI-N to the terminating UNI-C. The local connection ID values generated for the same connection at the initiating and

terminating end need not be the same. The UNI-N must verify the uniqueness of any UNI-C-assigned local ID (at the initiating end).

10.9.2 Service-Related Attributes

10.9.2.1 Encoding Type

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

- SONET T1.105
- SDH G.707

10.9.2.2 SONET/SDH traffic parameters

10.9.2.2.1 Signal type

The signal type defines the elementary signal on which multiple transforms can be applied successively to build the final signal being requested for the connection. The elementary signal types are defined in Section 4.

10.9.2.2.2 Contiguous Concatenation Type (CCT)

The contiguous concatenation type indicates the type of SONET/SDH contiguous concatenation to apply on the elementary signal.

10.9.2.2.3 Number of Contiguous Components (NCC)

The number of contiguous components indicates the number of identical SONET/SDH SPEs/VCs that are requested to be contiguously concatenated, as specified in the CCT field.

10.9.2.2.4 Number of Virtual Components (NVC)

The Number of Virtual Components field indicates the number of identical signals that are to be virtually concatenated. These signals can be either identical elementary signal SPEs/VCs, or identical contiguously concatenated signals. In the latter case, this attribute allows the virtual concatenation of contiguously concatenated signals to be requested, for instance the virtual concatenation of several STS-3c SPE's, or any STS-Xc SPE's (to obtain an STS-Xc-Yv SPE).

10.9.2.2.5 Multiplier

The Multiplier field indicates the number of identical signals that form the final signal constituting the connection. These signals can be identical elementary signals, identical contiguously concatenated signals, or identical virtually concatenated signals. Note that all of these signals belong to the same connection.

10.9.2.2.6 Transparency

The transparency field is a vector of flags that indicates the type of transparency requested. Several flags can be combined to provide different types of transparency. Not all combinations are necessarily valid. See Section 4 for more details.

10.9.2.3 Directionality

The Directionality attribute indicates whether the connection is uni-directional or bi-directional. Default is bi-directional.

10.9.2.4 Generalized Payload Identifier

The Generalized Payload Identifier (G-PID) indicates the payload carried within the established connection (i.e., identifies the client layer of the connection).

10.9.2.5 Service Level

The Service Level attribute indicates a class of service. A carrier may specify a range of different classes of service (e.g. gold, silver, bronze) with predefined characteristics (e.g. restoration plans). The pre-defined service types correspond to different types of network restoration (e.g. no restoration, 1+1 protection), connection set-up and hold priorities, reversion strategies for the connection after failures have been repaired, and retention strategies. The definition of the different service classes and their default values are set by the service provider.

10.9.3 Routing-Related Attributes

10.9.3.1 Diversity

For a new connection being created, this attribute indicates a list of *n* existing connections sourced from the same UNI-C with which diversity within the transport network is required. This attribute contains *n* items of the form <*diversity type*, *local connection ID*>, where the diversity type is selected from the following set:

- Node diverse. The new connection SHALL NOT use any network nodes that are in the path of the connection denoted by *local connection ID*.
- Link Diverse. The new connection SHALL NOT use any network links that are in the path of the connection denoted by *local connection ID*.
- SRLG diverse. The new connection SHALL NOT use any link that has the same SRLG as those in the path of the connection denoted by *local connection ID*.
- Shared Path. The new connection SHALL use the same links as those used by the connection denoted by *local connection ID*.

Note that a client may request connections that must be SRLG diverse, but not node diverse. To request node *and* SRLG diversity (for example), the client should have two separate entries in the list of diversity requirements - one for node diversity and one for SRLG diversity.

10.9.4 Policy-Related Attributes

10.9.4.1 Contract ID

This identifier is assigned by the service provider and configured in clients. This is not interpreted by the clients.

10.9.5 Miscellaneous Attributes

10.9.5.1 Connection Status

This indicates the status of a connection. The following values are defined:

- Connection active
- Connection does not exist

- Connection unavailable
- Connection pending

10.9.5.2 Error Code

The error code is used to describe the errors resulting from connection actions. Errors that may generally arise during connection establishment under CR-LDP and RSVP are defined as part of the specification of these protocols. In addition, the following error codes are defined specific to the UNI:

- Unauthorized sender (policy error)
- Unauthorized receiver (policy error)
- Service level not available
- Diversity not available
- G-PID unsupported
- Invalid / unknown connection ID

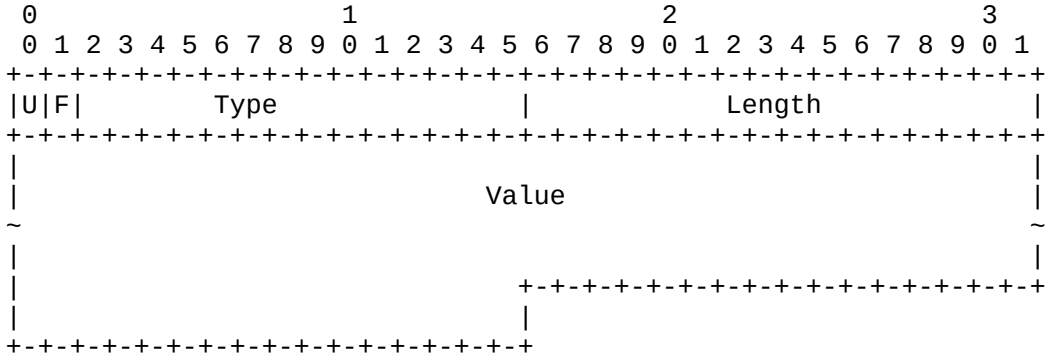
11 LDP Extensions for UNI Signaling

11.1 Overview

The Label Distribution Protocol (LDP) [RFC3036] has been defined for distributing labels among Label Switched Routers (LSRs) in a Multi-Protocol Label Switching (MPLS) network. Two LSRs which directly communicate using LDP to exchange labels are known as "LDP Peers". An "LDP Session" (realized over a TCP connection) is required between peer LSRs. LDP procedures permit LSRs to establish Label Switched Paths (LSPs) through an MPLS network by mapping network-layer routing information directly to data-link layer switched paths.

LDP associates a Forwarding Equivalence Class (FEC) with each LSP it creates. The FEC associated with an LSP specifies which packets are "mapped" to that LSP at the ingress LSR.

All LDP messages have a common structure that uses a Type-Length-Value (TLV) encoding scheme as shown below. The number of bits allocated for each field is as shown. The Value part of a TLV-encoded object, or TLV for short, may itself contain one or more TLVs. The Length field specifies the length of the Value field in octets. The meaning of U and F bits is defined in [RFC3036].



There are four categories of LDP messages:

- Discovery messages, used to announce and maintain the presence of an LSR in a network.
- Session messages, used to establish, maintain, and terminate sessions between LDP peers.
- Advertisement messages, used to create, change, and delete label mappings for FECs.
- Notification messages, used to provide advisory information and to signal error information.

Discovery messages provide a mechanism whereby LSRs indicate their presence in a network by sending a Hello message periodically. These messages are transmitted over UDP to the LDP port. The IP multicast address corresponding to "all routers on this subnet" is used as the destination IP address. When an LSR chooses to establish a session with another LSR (whose address is learned via the Hello message), it uses the LDP initialization procedure over TCP. Upon successful completion of the initialization procedure, the two LSRs become LDP peers, and may begin exchanging advertisement messages.

LDP uses TCP transport for session, advertisement and notification messages; i.e., for everything but the UDP-based discovery mechanism.

The LDP Messages defined in [RFC3036] are:

Message Name	Function
Notification Message	LSR notification of advisory or error information
Hello Message	Peer discovery
Initialization Message	LDP session establishment
KeepAlive Message	Monitors the integrity of the LDP session transport connection
Address Message	Advertise interface addresses
Address Withdraw Message	Withdraw previously advertised addresses
Label Mapping Message	Advertise FEC-label binding
Label Request Message	Request a binding (mapping) for a FEC
Label Abort Request Message	Abort an outstanding request
Label Withdraw Message	Breaks up the mapping between the FEC and the labels
Label Release Message	Signals the no need for specific FEC-label mapping.

LDP can operate in a number of modes depending on label distribution mode (*independent* or *ordered*), label retention mode (*conservative* or *liberal*), and label advertisement mode (*downstream on demand* or *downstream unsolicited*). These are defined in [RFC3036].

11.2 Use of LDP for UNI Signaling

In extending LDP for UNI signaling, there have been two main guiding principles. Firstly every effort has been made to limit the introduction of new LDP messages. Indeed, new messages have been limited to only those required to support the status enquiry function of UNI signaling.

Secondly care has been taken not to violate the LDP semantics as defined in [RFC3036]. Therefore, LDP UNI extensions could be easily implemented as simple additions to the existing LDP implementations.

The mode of operation of the LDP at the UNI is downstream on demand label advertisement with ordered control.

11.3 LDP Session Initialization

An LDP session is established between a pair of UNI peers, i.e., a UNI-C and a UNI-N. The UNI-C MUST play the active role during the LDP session initialization phase. Moreover,

- There will be a single LDP session between the UNI-C and the UNI-N, regardless of the number of data links between the corresponding client and TNE.

- In the case of proxy signaling, there will be a single LDP session between a proxy UNI-C and UNI-N regardless of the number of clients on whose behalf the proxy agent signals.

The LDP Hello and Extended Hello SHALL be used for neighbor discovery as specified in [RFC3036].

An LDP session starts when the UNI-C sends an LDP Initialization message to the UNI-N over a TCP connection. The UNI-N follows the procedure specified in section 2.5.3 of [RFC3036] for the passive LSR. It replies with an Initialization message to propose the parameters it wishes to use. The Initialization message also includes a Contract ID TLV. Contract ID is assigned by the network provider and identifies the service contract agreement between the client and the operator. The format of the Contract ID is opaque to LDP signaling.

The LDP session at the UNI MUST be maintained as per procedures described in section 2.5.6 of [RFC3036].

11.4 Connection Create using LDP

The connection create request under LDP signaling is implemented using the Label Request message as defined in [RFC3036]. The Label Request message is sent from the source UNI-C to the source UNI-N. At the other end, the Label Request message is sent from the destination UNI-N to destination UNI-C. The connection creation signaling sequence is shown in Figure 11-1.

The connection create request might indicate a number of attributes. Extensions are needed to the Label Request message to support the signaling of these attributes.

Connection create requests usually specify bi-directional connections. Following GMPLS signaling procedures [GMPS Sig], a bi-directional connection is signaled by using the Upstream Label TLV in the Label Request message. Reception of the Label Request message by the destination UNI-C signifies that the resources to establish the connection with the specified attributes are available in the network. It does not, however, imply that the connection is available for data transport. Specifically, the configuration of intermediate cross-connects may not have occurred yet. This process is started when the destination UNI-C sends a Label Mapping message in response to the Label Request. The Label Mapping message contains the identity of the data link to be used and a generalized label (see Section 10). It is required that the Label Mapping message be sent only after the corresponding client device has configured its resources (including any cross-connects) to activate the connection. If it so desires, the destination UNI-C MAY indicate in the Label Mapping message that a reservation confirmation indication is needed. This indication is sent by the source UNI-C to the source UNI-N and from the destination UNI-N to the destination UNI-C. It indicates to the destination UNI-C that the corresponding client is free to send data over the newly established connection. The reservation confirmation indication is implemented using the LDP Notification message, with the status code “reserve confirm”. Similarly, the connection becomes available for the source client when the source UNI-C receives a Label Mapping message from the source UNI-N.

Contention for labels may occur when both the UNI-C and the UNI-N choose the same data link and generalized label for two bi-directional connections being concurrently set up (i.e., the UNI-C is the source for one of these connections and the destination for the other). To resolve such contention, the node with the higher Node ID MUST issue a NOTIFICATION message with a “Routing problem/Label allocation

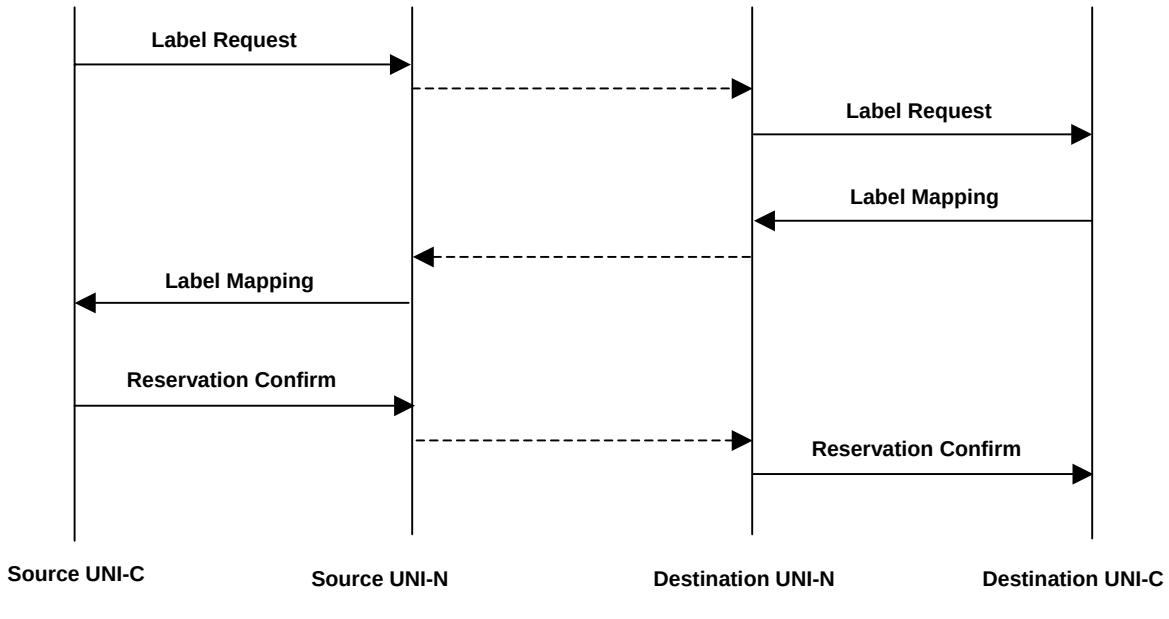


Figure 11-1: Connection Create Message Sequence

failure" indication to the other. Upon receipt of such an error, a node SHOULD try to allocate a different Upstream label (and a different Suggested Label, if used) to the bi-directional connection. If no other resources are available, however, the node must proceed with standard error handling.

The connection create request might fail for a number of reasons, e.g. no bandwidth available, no physical connectivity, SLA violation, connection rejected by far end UNI-C, etc. In these cases, the failure is indicated to the source UNI-C using the LDP Notification message with the status code reflecting the reason for the failure. Figure 11-2 shows the signaling message sequence when the connection create request is rejected by the network.

Should a client desire to abort the connection creation process after sending the Label Request message, an LDP Abort message MUST be sent as defined in section 3.5.9. of [RFC3036]. Specifically the Message ID used in the Label Request Message (see Section 11.8) is used in the Abort Message as the temporary local connection identifier.

11.5 Connection Deletion Using LDP

LDP employs two mechanisms for an LSR (label switched router) to inform its peer to stop using a particular label. The first method is based on the use of the Label Withdraw message. A node sends a Label Withdraw message to its peer to indicate that the latter should stop using a specific label that the node had previously advertised. The second method is based on the use of Label Release message. A node sends this message to its peer to indicate that it no longer needs a specific label that had previously been advertised by the peer.

The UNI LDP extensions make use of the Label Release and Label Withdraw messages for connection deletion. The choice of which message to use depends on the entity that initiates the deletion. The Label Withdraw message is used when connection deletion is in the upstream direction (e.g., by the destination UNI-C). As per the LDP procedure in section 3.5.10 of [RFC3036], a Label Release message is sent in the downstream direction (e.g., by the source UNI-C) to acknowledge the delete request.

The Label Release message is used when connection deletion is in the downstream direction (e.g., by the source UNI-C). In this case the delete request is confirmed in the upstream direction by the use of LDP Notification message with the status code "delete success" (e.g., by the destination UNI-C). Figures 11-3 and 11-4 show graceful connection deletion requested by the source UNI-C and the destination UNI-C, respectively.

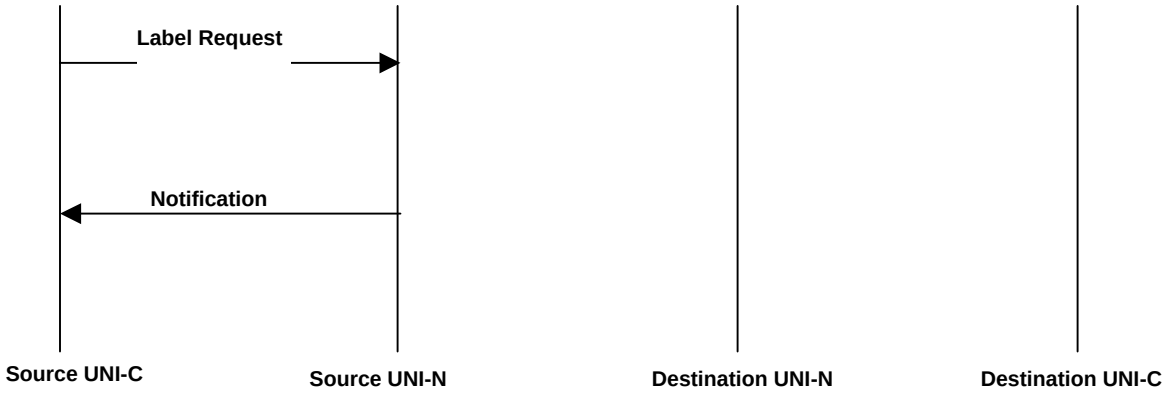


Figure 11-2: Connection Set-up Rejection by the Network

Figure 11-3 shows that the delete request is preceded by an LDP Notification message with Admin Status TLV [GMPLS SIG] included. The reason for this is as follows. In optical networks, the loss of signal will propagate faster than the delete request. Thus, a downstream network node may detect loss of signal and incorrectly trigger connection restoration. The Admin Status TLV is used to prepare the connection for deletion within the network as well as at the destination client by changing the administration status of the connection as indicated. This message SHOULD be sent by the source UNI-C to initiate connection deletion.

Figure 11-5 shows a connection deletion initiated within the network (e.g., a forced deletion triggered by the management layer). In this case, both Label Withdraw and Label Release messages are used to initiate deletion at the clients as shown.

11.6 Failure Detection and Recovery Using LDP

One of the requirements on signaling is that the failures in the control plane (e.g., loss of the signaling channel) should not impact existing data plane connections. To satisfy this requirement, a mechanism MUST exist to detect a signaling communication failure. Furthermore, a recovery procedure SHALL guarantee connection state integrity at both the UNI-C and the UNI-N.

The LDP Keep Alive mechanism MUST be used to detect signaling communication failures between a UNI-C and a UNI-N unless an alternative mechanism is in place to detect such failures more efficiently. During the signaling communication failure, all active connections are maintained (the bearer connection is active) and all transient connections (in-progress) are cleared.

Upon signaling communication re-establishment (i.e. re-establishment of LDP Keep Alive) a resynchronization period is required in order to synchronize connection state information across the UNI. This synchronization SHALL be performed prior to processing new connection requests.

The resynchronization period starts when the UNI-C either queries UNI-N for the states of all connections associated with a particular data link, or deletes the connections (implicitly or explicitly). In the former case, the UNI-N will respond with the appropriate status information. Connection state, once created, remains within the network until implicitly or explicitly deleted by a UNI-C entity or by the network operator. The responsibility is on the UNI-C to resynchronize the connection state with the UNI-N. The UNI-N does not initiate synchronization with the UNI-C.

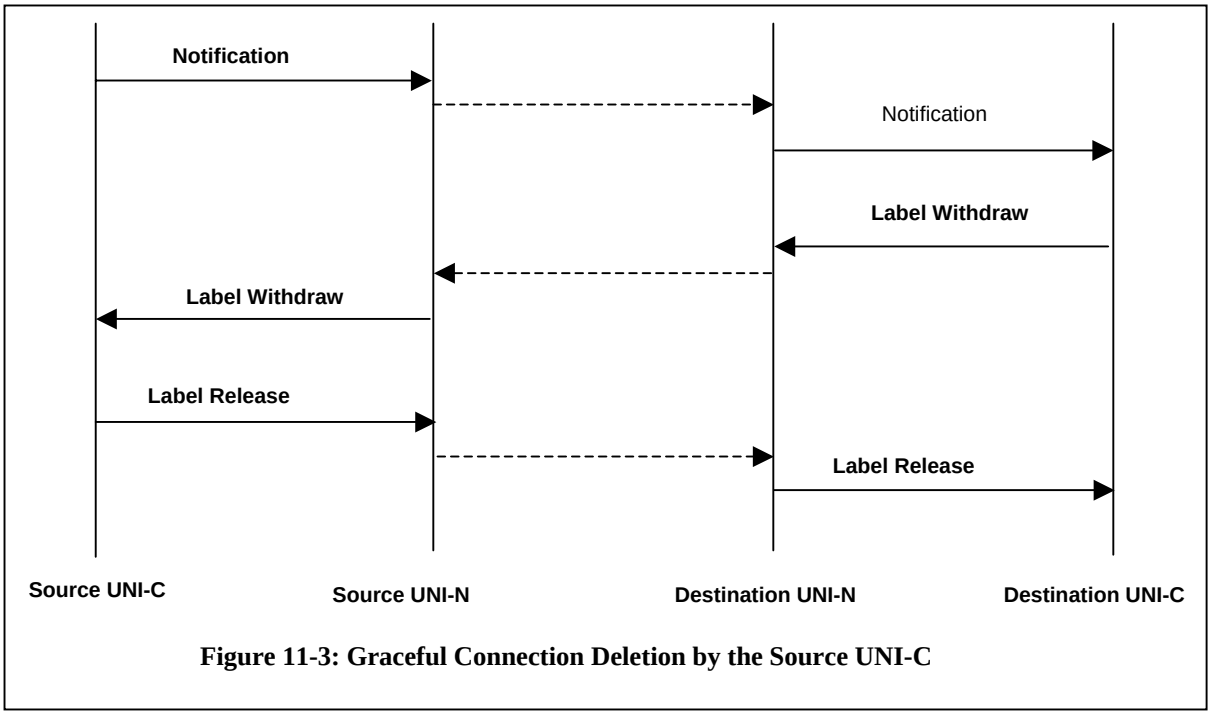


Figure 11-3: Graceful Connection Deletion by the Source UNI-C

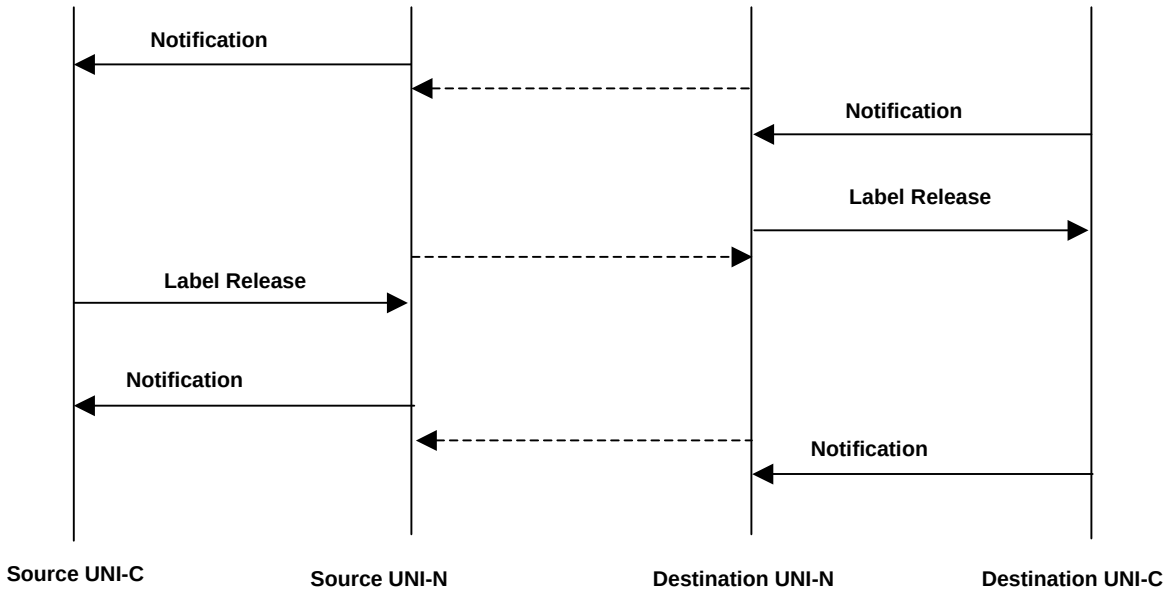


Figure 11-4: Graceful Connection Deletion by the Destination UNI-C

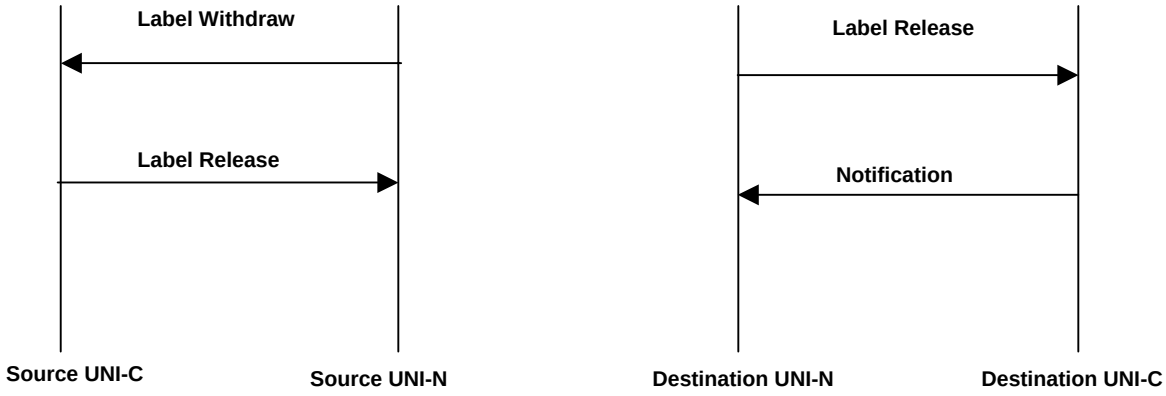


Figure 11-5: Connection Deletion Initiated by the Network

There are three cases to consider during recovery:

1. Both the UNI-C and the UNI-N have retained connection information during the control plane failure. In this case, the recovery procedure consists of the UNI-C sending Status Enquiry messages to the UNI-N, requesting summary connection information.
2. All connection information is lost by the UNI-C during a control plane failure. In this case, the recovery procedure requires the UNI-C to query the UNI-N about the status of all connections to rebuild its connection state information. This is done by using the Status Enquiry message to get detailed connection information based on LDP session, data link or TNA address.
3. All connection information is lost by the UNI-C and the UNI-C decides to delete these connections on:

- A per LDP session basis.
- A specific data link
- All data links that share a specified TNA address

In this case, the synchronization is a simple restart process that can be achieved by sending a Label Release message with the corresponding TNA and optional logical port parameters. Figure 11-6 shows the scenario for graceful and forced deletion of all connection (for the purpose of simplicity, the diagram assumes that the connections to be deleted all terminate on the same destination client). Graceful deletion is indicated by the destination UNI-N, which sends a Notification message to the destination UNI-C indicating the intention to delete. This Notification message is OPTIONALLY sent in response to the Label Release message received from the source UNI-C. When multiple destinations are involved, a separate Notification message SHALL be sent per destination.

Resynchronization is deemed to be complete between a given UNI-C and a UNI-N when the state of each connection known by the UNI-N is either queried or deleted by the UNI-C. New connections, which require the involvement of the UNI-C, cannot be established until resynchronization is completed.

11.7 TLV Encoding for UNI Parameters

The general TLV encoding described in Section 3.4 of [RFC3036] and in [CR-LDP] MUST be used for UNI signaling. This section describes the UNI-related TLV encoding.

11.7.1 Generalized Label Request, Generalized Label, Suggested Label, Upstream Label, Label Set and Admin Status TLVs

Defined in [GMPLS SIG], [GMPLS CR-LDP] and [GMPLS SONET/SDH].

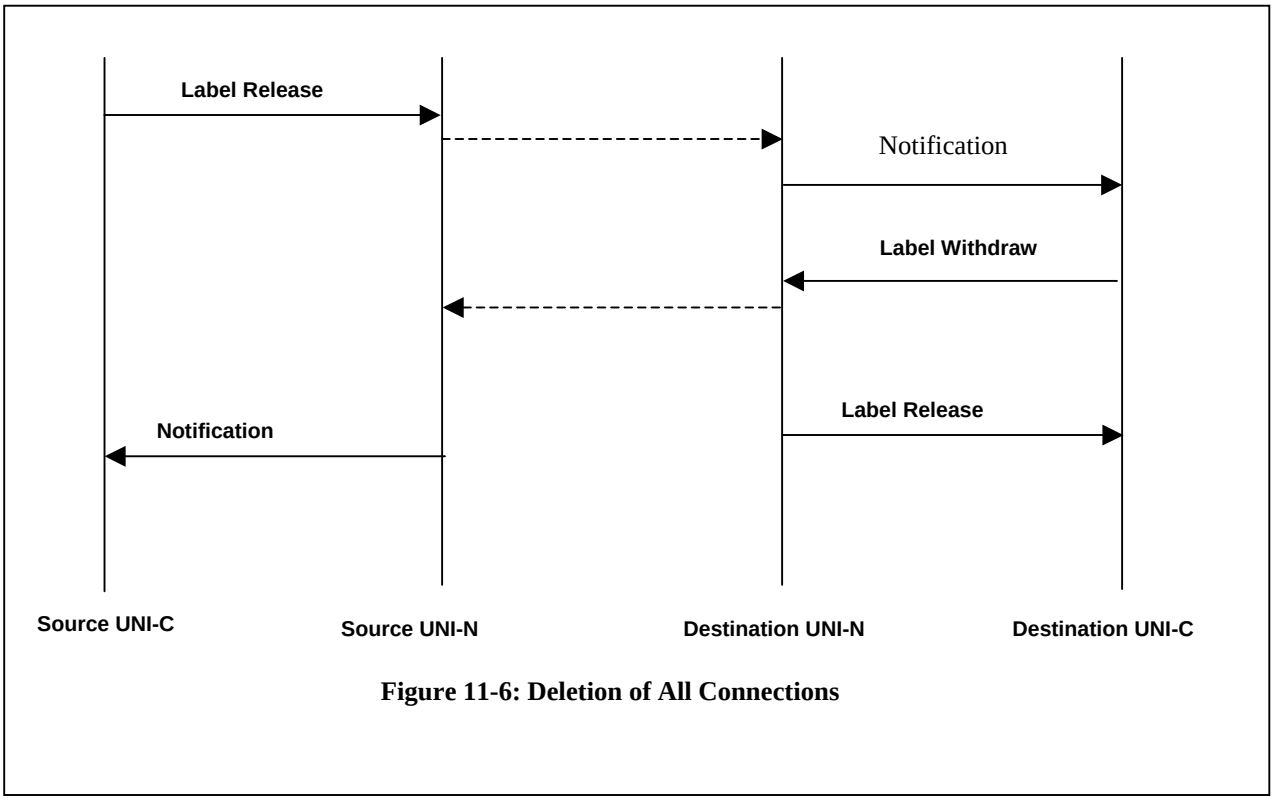


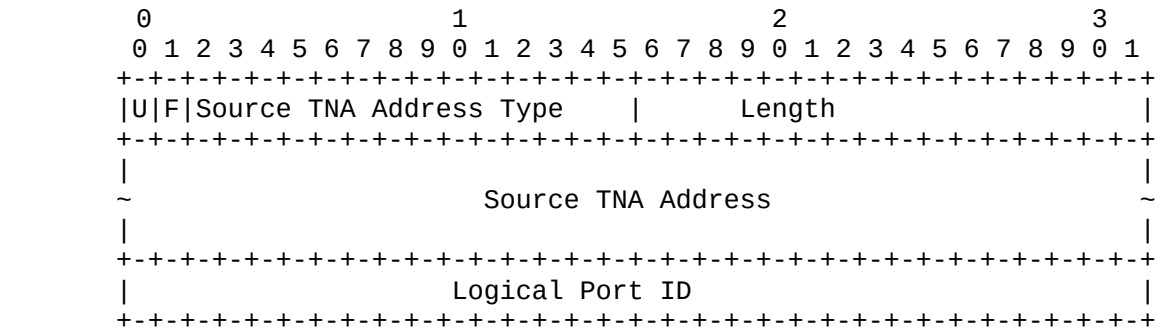
Figure 11-6: Deletion of All Connections

11.7.2 SONET/SDH Traffic Parameters TLV

Defined in [GMPLS SONET/SDH].

11.7.3 Source ID TLV

The Source ID TLV is used to identify the source client and logical port. The Source ID TLV MUST contain the source TNA address, and may OPTIONALLY contain a Logical port ID. The encoding for the Source ID TLV is:



Source TNA Address Type:

This can be in one of three formats, IPv4, IPv6, or NSAP. The codes (to be assigned) are:

Code (TBA)	Type
0x960 0x961 0x962	IPv4 IPv6 NSAP

Length

As per the contents of the Source TNA Address field, and the inclusion of the logical port ID field.

Source TNA Address

IPv4: 4-octet IPv4 address

IPv6: 16-octet IPv6 address

NSAP: Variable length. The NSAP format is structured according to ISO/IEC 8348, 1993 (identical to ITU X.213, 1992).

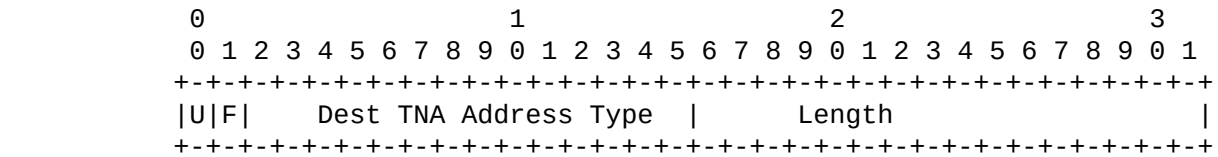
Only TNA addresses of the same type SHOULD be compared for equality.

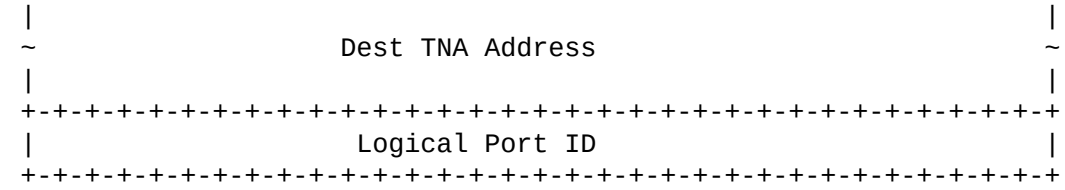
Logical Port ID:

A 32-bit unsigned number identifying a logical port at the source client. Logical Port ID is OPTIONAL and, as such, may not be present.

11.7.4 Dest ID TLV

Dest ID TLV is used to identify the destination end of the connection. The Dest ID TLV encoding is:





Dest TNA Address Type:

The codes are:

Code (TBA)	Type
0x963	IPv4
0x964	IPv6
0x965	NSAP

Length

As per the contents of the Dest TNA Address field, and the inclusion of the logical port ID field.

Dest TNA Address

IPv4: 4-octet IPv4 address

IPv6: 16-octet IPv6 address

NSAP: Variable length. The NSAP format is structured according to ISO/IEC 8348, 1993 (identical to ITU X.213, 1992).

The NSAP format is structured according to ISO/IEC 8348, 1993 (identical to ITU X.213, 1992)

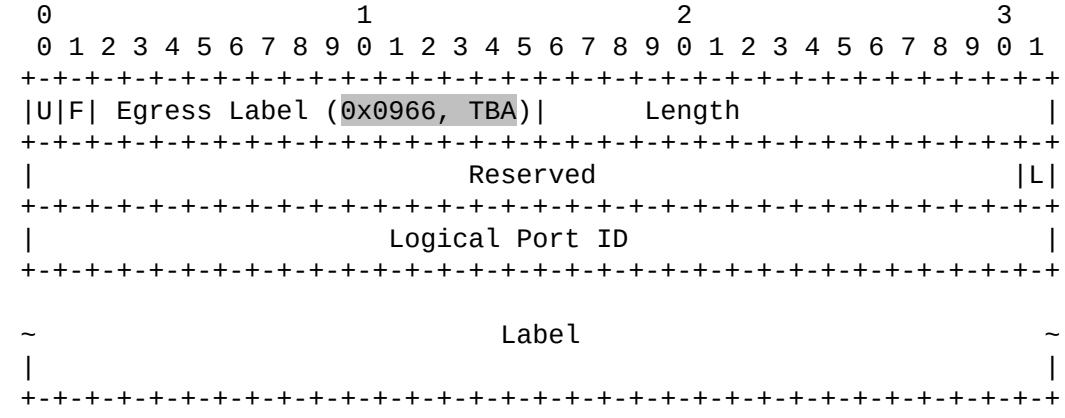
Only TNA addresses of the same type SHOULD be compared for equality.

Logical Port ID:

A 32-bit unsigned number indicating identifying a logical port at the destination client device. Logical Port ID is OPTIONAL and, as such, may not be present.

11.7.5 Egress Label TLV

Egress Label TLV is an OPTIONAL UNI-specific TLV. Egress Label TLV, when present, indicates the egress label to be used at the destination end. Its encoding is:



L-bit:

If L=0, then the Label value MUST be copied into Label Set TLV in the outgoing Label Request message from the destination UNI-N to destination UNI-C. If L=1, then the Label value MUST be

copied into an Upstream Label TLV in the outgoing Label Request message from the destination UNI-N to the destination UNI-C. Thus, the Label Request message originated by the source UNI-C may contain up to two Egress Label TLVs, one with the L-bit set, and the other with the L-bit not set.

Logical Port Identifier

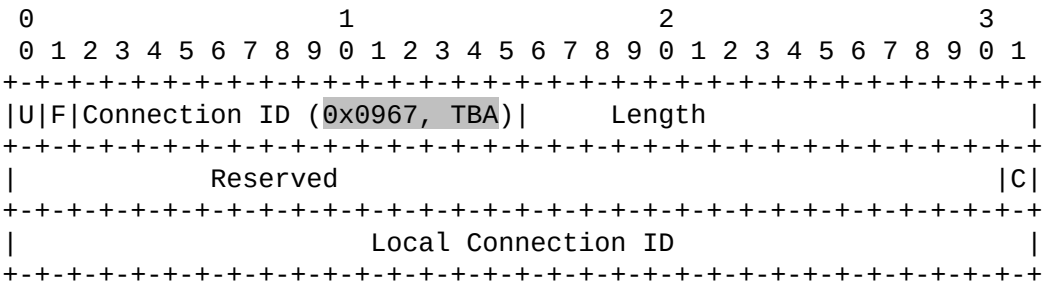
The Logical Port identifier is used to select a link at the destination UNI. The identified link MUST be used to allocate resources for the connection.

Label

This field identifies the label to be used. The format of this field is identical to the one used by the Label field in GMPLS SONET/SDH Label [GMPLS SONET/SDH].

11.7.6 Local Connection ID TLV

The Local Connection ID TLV is used to uniquely identify a connection at the local UNI. The format of the Local Connection ID TLV is as follows:



C-bit:

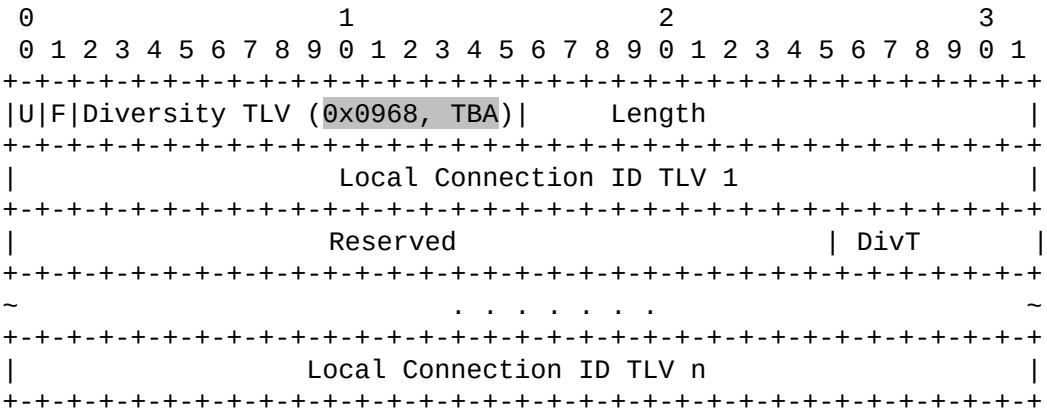
The value of the C bit is set by the destination UNI-C whenever a reservation confirmation indication is needed.

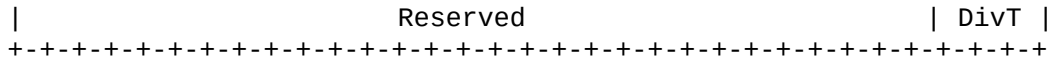
Local Connection ID:

This is a 32-bit number.

11.7.7 Diversity TLV

Diversity TLV lists all the other connections from which the requested connection MUST be diverse, or shares the physical path (see Section 10). It also specifies the type of diversity. The encoding of the Diversity TLV is as follows:





Local Connection ID TLV *i*:

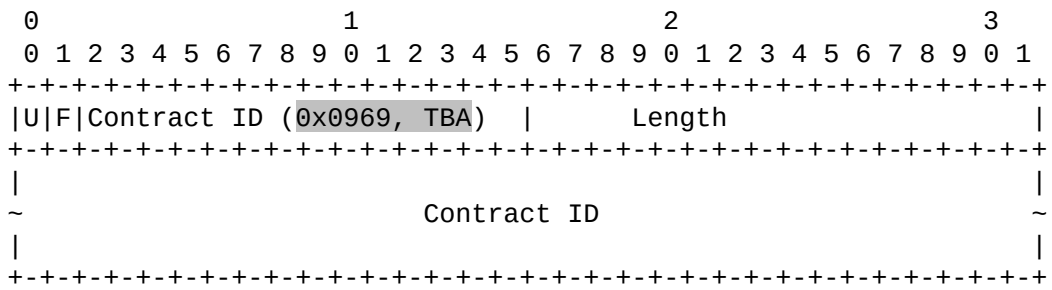
This is the Connection ID of an existing connection from which the requested connection must be diverse.

DivT (Diversity Type):

DivT specifies the manner by which the requested connection should be diverse (see Section 10). The allowed values are:

Value	Type
0x00	Link diverse
0x01	Node diverse
0x02	SRLG diverse
0x03	Shared path

11.7.8 Contract ID TLV

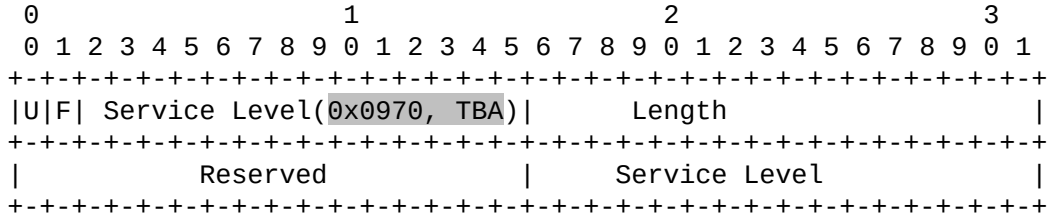


Contract ID:

This is a variable-length string of characters whose format will be determined by the service provider (see Section 10).

11.7.9 UNI Service Level TLV

The UNI Service Level is defined in Section 10. The encoding of the UNI Service TLV is:



Service Level:

A 16-bit number. The specific values are assigned by the network operator and correspond to various levels of service, as described in Section 10.

11.8 LDP Message Extensions for UNI

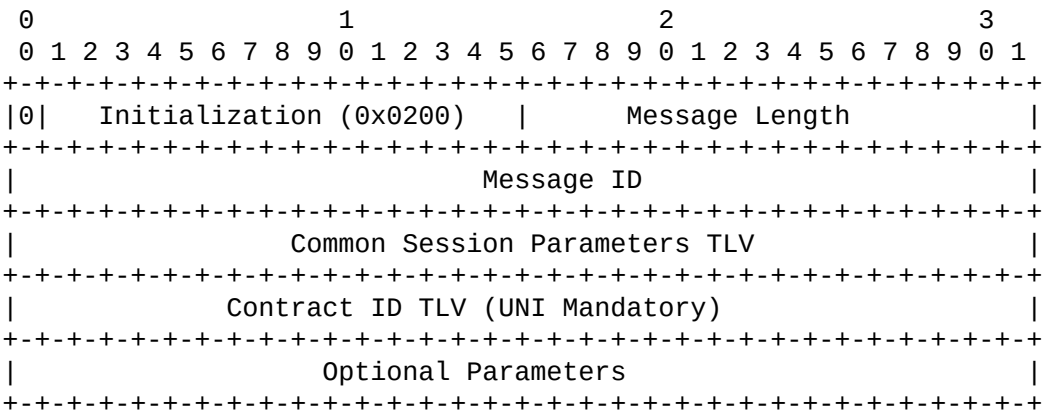
This section describes the necessary extensions for LDP messages for the support of UNI signaling. This section also includes the definition of two new messages, Status Enquiry and Status Response.

11.8.1 Hello Message

The format of the Hello message and the Hello procedure are as defined in section 3.5.2 in [RFC3036].

11.8.2 Initialization Message

The encoding for the Initialization message is:

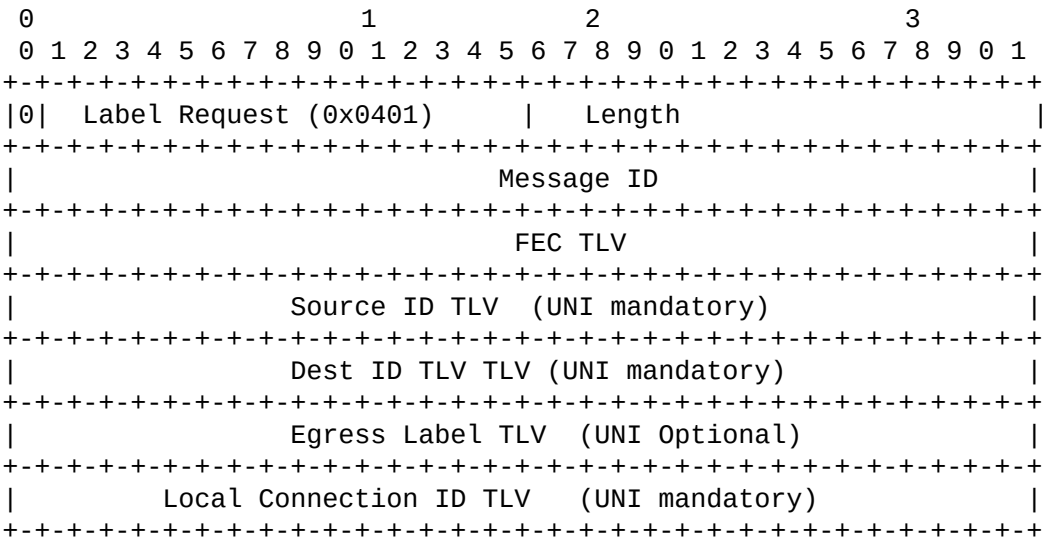


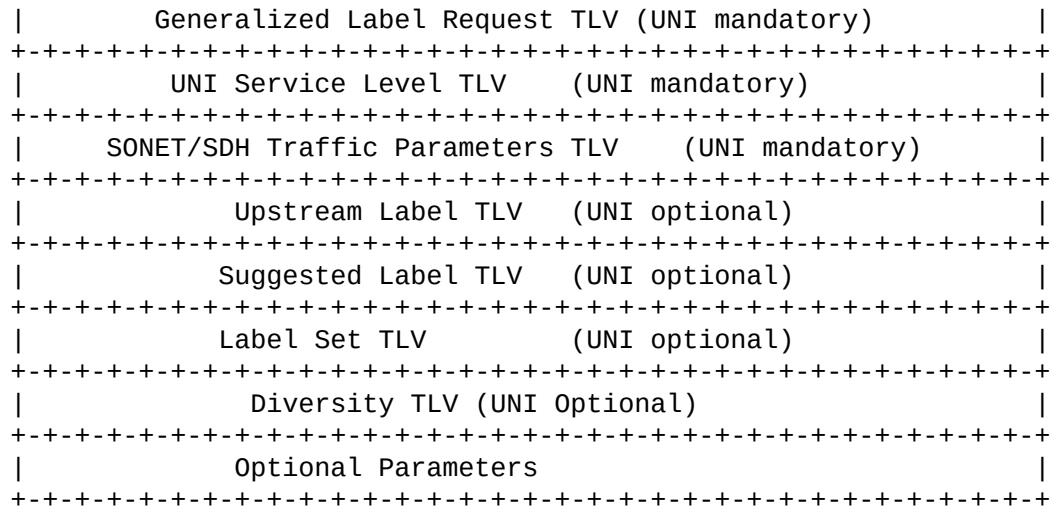
The initialization message procedure is as described in Section 2.5 in [RFC3036] and in Section 11.3 above.

Message ID is as defined in Section 3.5 in [RFC3036]. Common Session Parameter TLV is as defined in Section 3.5.3 in [RFC3036].

11.8.3 Label Request Message

The encoding of the Label Request Message for UNI is:





The Label Request Message is sent from:

- the source UNI-C to source UNI-N to indicate an outgoing connection request;
- the destination UNI-N to the destination UNI-C to indicate an incoming connection request.

How the Label Request message is propagated through the network from the source UNI-N to the destination UNI-N is outside the scope of this specification.

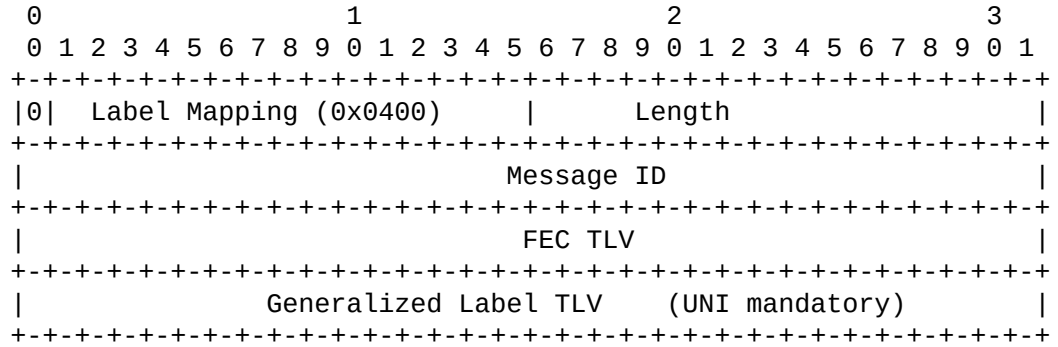
In the Label Request Message, the initiating UNI-C identifies the two connection termination points (Source and Dest ID TLVs). The Logical Port ID field MUST be present in the Source ID TLV. The source UNI-C also selects the local Connection ID. A (possibly different) local Connection ID is assigned at the destination end by the UNI-N.

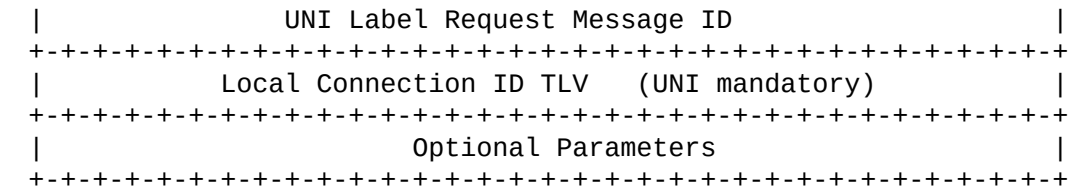
Upon the reception of the Label Request Message, the source UNI-N should verify that the signaled attributes (including the validity of the Source and the Destination IDs) can be supported. Failure to support one or more of the connection attributes triggers the generation of the Notification Message with the appropriate error code.

The Label Request message Message ID is used as transaction identifiers, as described in [RFC3036]. The UNI-C SHOULD NOT reuse the Message ID of a Label Request message until the corresponding transaction completes.

11.8.4 Label Mapping Message

The format of the Label Mapping message for the UNI is:





The Label Mapping message procedure for the UNI is limited to downstream on demand ordered control mode. The UNI Label Mapping Message flows between:

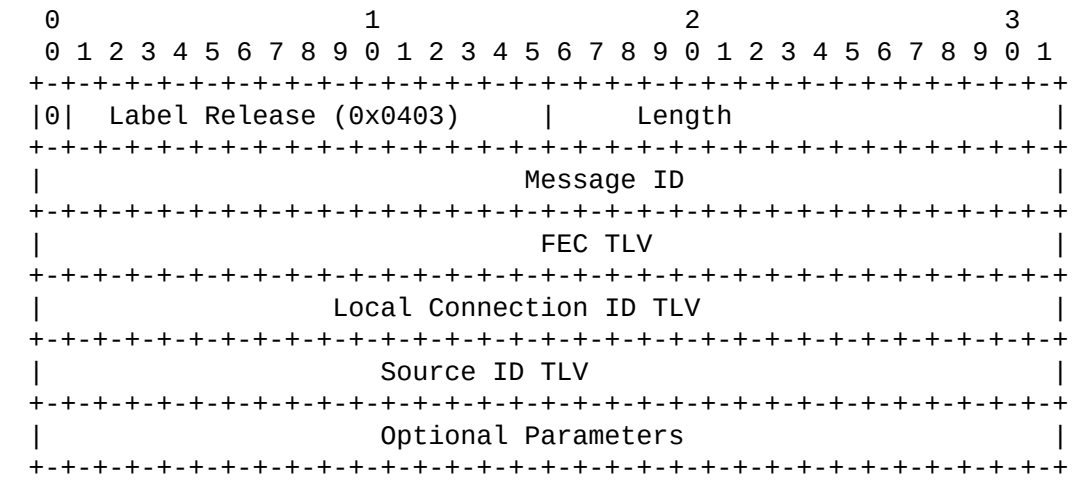
- The destination UNI-C to destination UNI-N in response to a Label Request message;
- The source UNI-N to the source UNI-C to indicate the successful establishment of a connection requested previously.

The Local Connection ID parameter is copied from the corresponding Label Request message.

A terminating UNI-C that desires to receive a reservation confirmation from the initiating UNI-C MUST set the C-bit in the Local Connection ID TLV.

11.8.5 Label Release Message

The encoding for the Label Release message is:



The LDP Label Release Message is used for connection deletion in the downstream direction, e.g. when the connection termination is initiated by

- the destination UNI-C (during graceful connection deletion)
- the source UNI-C (non-graceful connection deletion).

The UNI Label Release Message is sent by:

- The source UNI-C to source UNI-N to request the deletion a connection;

- The destination UNI-N to a destination UNI-C to indicate the deletion of a connection by the network.

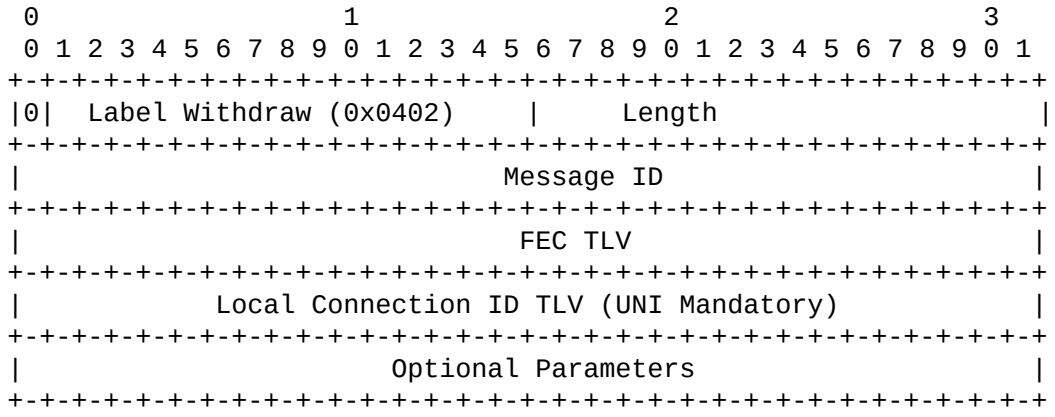
The receiver of the Label Release Message must respond with a Notification Message with the appropriate status code indicating the success of the delete request.

Either the Local Connection ID TLV or the Source ID TLV MUST be included in the Label Release Message. When the Local Connection ID TLV is included, it is an indication that a specific connection must be deleted. The Source ID TLV SHOULD only be present if a forced (i.e. non graceful) connection deletion is to be performed.

When the Source ID TLV is included, it is an indication to the network that the source UNI-C requires deletion of all the connections associated with the specified link. This feature is primarily used for forced deletion of connections in failure recovery.

11.8.6 Label Withdraw Message

The encoding for the Label Withdraw message at the UNI is:



The LDP Label Withdraw message is used for connection deletion in the upstream direction, e.g. when the connection termination is initiated by

- the destination UNI-C (non-graceful connection deletion), or
- the source UNI-C (graceful connection deletion).

The UNI Label Withdraw Message is sent by:

- The destination UNI-C to destination UNI-N to request the deletion of a connection
- The source UNI-N to the source UNI-C to indicate the deletion of the connection by the network.

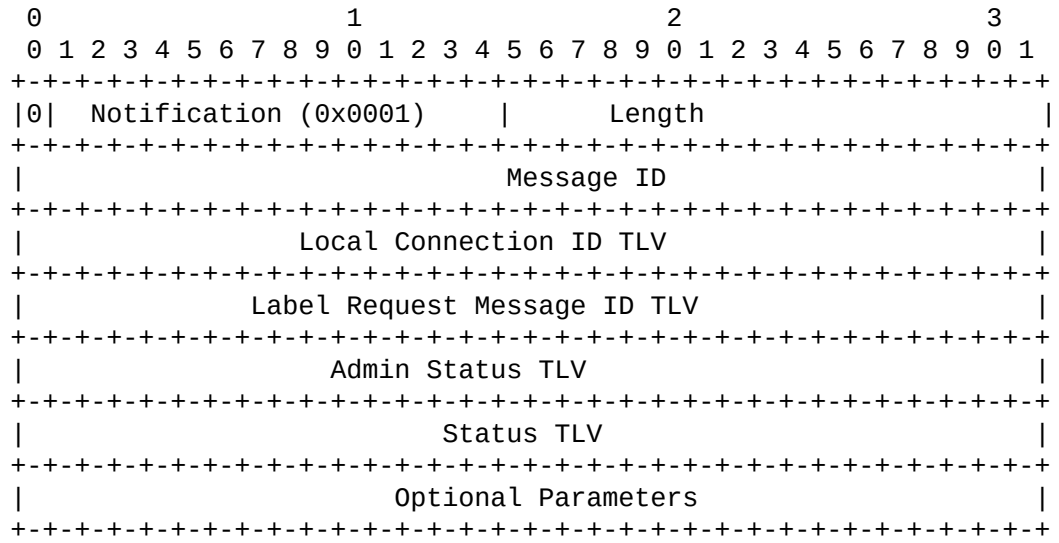
The procedure for the Label Withdraw Message is defined in section 3.5.10 of [RFC3036]. The recipient of the Label Withdraw Message MUST respond with a Label Release message as in [RFC3036]. The Label Withdraw message for UNI carries a mandatory Connection ID.

11.8.7 Label Abort Message

The format and the procedure of the Label Abort message are as given in section 3.5.9 of [RFC3036].

11.8.8 Notification Message

The format of the Notification message is:



The UNI Notification Message must be forwarded towards the entity originating the Label Request, Label Release, Status Enquiry, or Abort.

The Notification message plays a number of roles in UNI signaling:

- A failed connection setup event is indicated to the source UNI-C using a Notification message. In this case, the Notification message is generated in response to a Label Request message. The Status TLV MUST include the status code for the cause of the failed setup, e.g. “connection parameters not supported”.

- A Notification message is needed to initiate graceful deletion of a specific connection. In this case the Notification message MUST include the Local Connection ID of the connection to be deleted and the Admin Status TLV..

- A Notification message is sent in response to a connection deletion in the downstream direction, i.e. initiated by a Label Release message. In this case the Status TLV MUST include the status code for “delete_success”. The Notification message can be sent in response to a single connection deletion or the deletion of all the connections associated with a source identification point (LDP session, logical link, or TNA address). For single connection deletion, the Notification message MUST include the Local Connection ID of the deleted connection. When used to acknowledge the deletion of a group of connections, the Notification message MUST not contain the Local Connection IDs of any of the connections that were deleted.

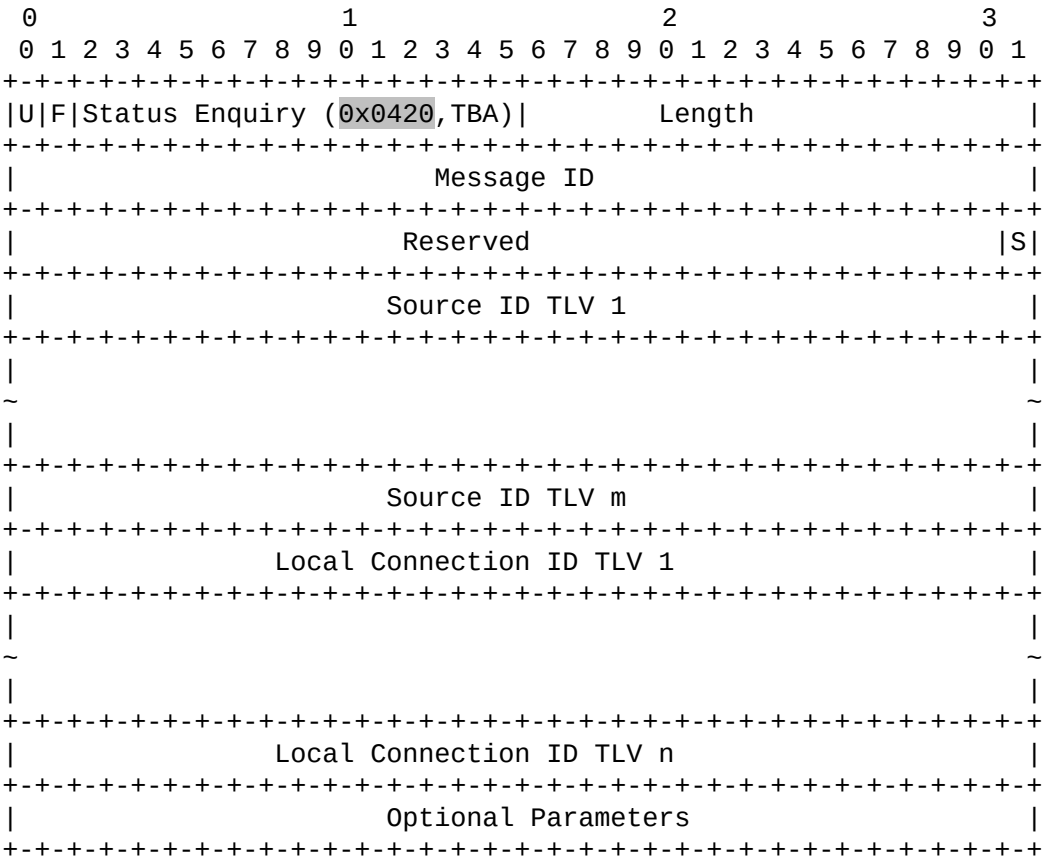
- A Notification message is sent for those cases where the destination UNI-C requests a reservation confirmation indication from the source UNI-C. In this case the status TLV MUST include the “reserv_confirm” status code.

- A Notification message is sent for those cases where the source UNI-C attempts to abort a connection request after sending the Label Request Message. The Notification message is sent in response to an Abort message. In this case the Notification message MUST include the Label Request Message ID TLV.

The use of the Notification message for the UNI includes those procedures that are specified in [RFC3036].

11.8.9 Status Enquiry Message

The Status Enquiry is a new LDP message that is defined specifically for LDP applications for UNI signaling. The encoding for the Status Enquiry Message is:



The Status Enquiry message allows a client to enquire about the status of a specific connection, or group of connections for which the client is an end point.

When a Source ID TLV is present in the Status Enquiry message, the client is in effect requesting the status of all the connections associated with the specified link.

When a Local Connection ID TLV is present in the Status Enquiry message, the client is requesting the status of that specific connection.

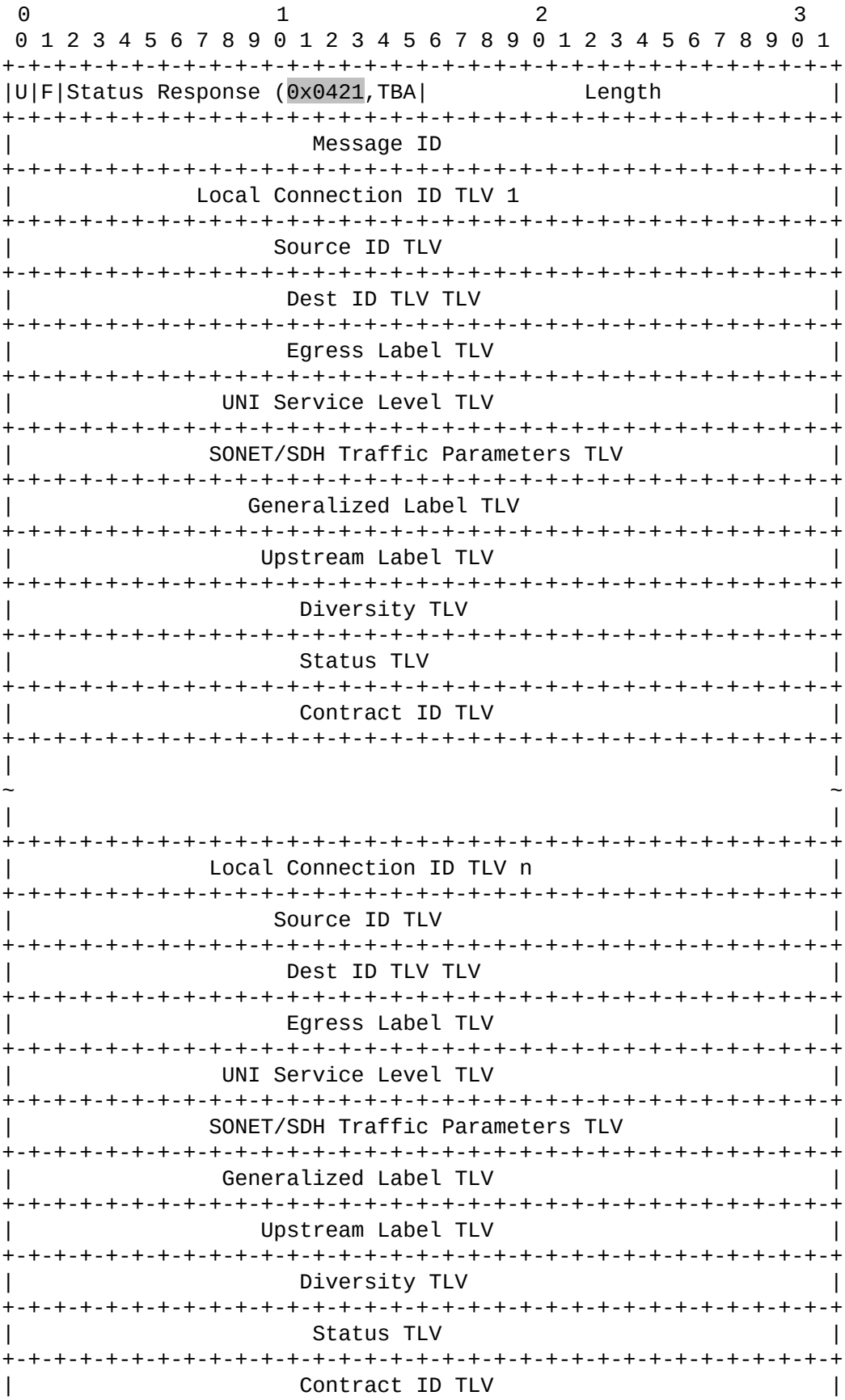
Status Enquiry can be generated by the client at any time. The Status Enquiry message can also be used to enquire about status of existing connections.

S-bit:

When S=1, it indicates that the Status Enquiry message is for summary information. In that case the Status Response message contains summary information (Connection ID and connection Status) of the specified connections. When S=0, detailed information, including connection attributes, is requested.

11.8.10 Status Response Message

The Status Response message is a new LDP message that is defined specifically for UNI signaling. The encoding of the Status Response message is:


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
|U|F|Status Response (0x0421,TBA|
Length
Message ID
Local Connection ID TLV 1
Source ID TLV
Dest ID TLV TLV
Egress Label TLV
UNI Service Level TLV
SONET/SDH Traffic Parameters TLV
Generalized Label TLV
Upstream Label TLV
Diversity TLV
Status TLV
Contract ID TLV
Local Connection ID TLV n
Source ID TLV
Dest ID TLV TLV
Egress Label TLV
UNI Service Level TLV
SONET/SDH Traffic Parameters TLV
Generalized Label TLV
Upstream Label TLV
Diversity TLV
Status TLV
Contract ID TLV

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Optional Parameters                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Status Response message is generated in response to a Status Enquiry message. The Status Response message contains information regarding the status of those connections (implicitly or explicitly) specified in the corresponding Status Enquiry message. The type of information included in the Status Response message depends on the whether the Status Enquiry is for summary or for detailed information.

11.9 LDP Code Points

UNI signaling using LDP defined in this section utilizes a number of new TLV-encoded objects. Many of these objects have been defined as part of the GMPLS CR-LDP signaling specification [GMPLS CR-LDP] and some new objects have been defined specifically for UNI signaling. All these objects must have unique type values assigned. Furthermore, status codes defined in this section must have unique values assigned. These assignments are performed by the Internet Assigned Numbers Authority (IANA) based on policies that can be found in [RFC2434]. For the new GMPLS CR-LDP objects, the class numbers and class types have been selected by the protocol designers but not officially assigned by the IANA. For other objects defined in this section, the type values have been suggested. *These values have not yet been assigned by the IANA*. These are all highlighted and marked To be Assigned (TBA). Until these code points are assigned, implementations that use these values must be prepared to comply with any changes in values finally assigned.

The following table summarizes the status codes specific to UNI signaling, as defined in this section. A number of other status codes generally applicable to LDP and CR-LDP signaling have been defined in [RFC3036] and [CR-LDP].

Description	Status Code	E/F Bits	Reference
Destination unreachable	0x3F100100	1/0	UNI Private use [CR-LDP]
Diversity not available	0x3F100101	1/0	UNI Private use [CR-LDP]
Service level not available	0x3F100102	1/0	UNI Private use [CR-LDP]
Invalid/Unknown connection ID	0x3F100103	1/0	UNI Private use [CR-LDP]
Unauthorized sender	0x3F100104	1/0	UNI Private use [CR-LDP]
Unauthorized receiver	0x3F100105	1/0	UNI Private use [CR-LDP]
G-PID undefined	0x3F100106	1/0	UNI Private use [CR-LDP]
Delete success	0x3F100107	0/0	UNI Private use [CR-LDP]
Reserve confirm	0x3F100108	0/0	UNI Private use [CR-LDP]
Connection active	0x3F100109	0/0	UNI Private use [CR-LDP]
Connection unavailable	0x3F10010A	0/0	UNI Private use [CR-LDP]
Connection Pending	0x3F10010B	0/0	UNI Private use [CR-LDP]

12 RSVP Extensions for UNI Signaling

12.1 Overview

RSVP (Resource reSerVation Protocol) is a protocol for establishing network resources for IP sessions (or "flows") [RFC2205]. The RSVP definition consists of basic procedures, messages and object formats for signaling in an IP network. RSVP with Traffic Engineering extensions (RSVP-TE) has been defined for establishing connections subject to routing constraints in an MPLS network [RSVP-TE]. The RSVP-TE definition includes additional procedures, messages and object formats as extensions to the base RSVP definition. Generalized MPLS (GMPLS) extensions for RSVP-TE signaling [GMPLS SIG, GMPLS RSVP-TE] extends RSVP-TE signaling procedures and objects to cover different types of switching applications such as circuit switching, wavelength switching, etc.

In this section, UNI signaling based on adapting RSVP, RSVP-TE, and GMPLS RSVP-TE specifications is defined. This definition leverages the above specifications to the maximum extent possible. A few new objects are defined for supporting connection attributes that are unique to UNI 1.0. In addition to defining these new objects, this section also specifies the applicable values for certain objects and the execution of specific procedures where the above RSVP-related specifications allow variants.

In this section, the directional terms "source" vs. "destination", "originating" vs. "terminating", "upstream" vs. "downstream", "previous hop" vs. "next hop", and "incoming interface" vs. "outgoing interface" are defined with respect to the direction of control message flow as in [RFC2205, RSVP-TE, GMPLS RSVP-TE].

12.2 Basic RSVP Protocol Operation

There are two fundamental RSVP message types: Path and Resv [RFC2205]. A node originates a connection establishment request by transmitting an RSVP Path message addressed to the connection destination. In response to receiving a Path message, the destination node sends a reservation request (Resv) message upstream towards the connection source. Path and Resv messages create "path" and "reservation" state in nodes along the path of the connection. A connection is established when the initiating node receives a Resv message for the connection.

The Path and Resv state can be explicitly removed using PathTear and ResvTear messages. The PathTear message is sent from the source to the destination and removes both the path and reservation state for the associated connection. The ResvTear message is sent from the destination to the source and only removes the associated reservation state.

For each RSVP message type, there is a set of rules for the permissible choice of object types. These rules are specified using Backus-Naur Form (BNF). The BNF specification implies an order for the objects in a message. However, in many (but not all) cases, object order makes no logical difference. An implementation should create messages with the objects in the order shown for each message, and MUST accept the objects in any order.

12.3 UNI 1.0 Signaling Messages and RSVP Objects

The UNI 1.0 abstract messages were described in Section 10. Most of these are directly supported by re-using existing procedures, messages, and objects defined under RSVP-TE [RSVP-TE] and GMPLS extensions for RSVP-TE [GMPLS SIG, GMPLS RSVP-TE]. This is summarized in Table 12-1.

Section 10 also defines the set of attributes to be signaled. Table 12-2 summarizes those attributes and the corresponding RSVP objects. Specific UNI-related object formats and usage are described in Section 12.5.

Message No.	Abstract Message Description	RSVP Message
1	Connection Create Request	Path
2	Connection Create Response	Resv, PathErr
3	Connection Create Confirmation	ResvConf
4	Connection Delete Request	Path or Resv with ADMIN_STATUS "Deletion in Progress" bit
5	Connection Delete Response	PathErr with Path_State_Removed flag, PathTear
6	Connection Status Enquiry	implicit
7	Connection Status Response	implicit
8	Notification	PathErr, ResvErr

Table 12-1: Mapping between UNI Abstract Messages and RSVP Messages

UNI Attributes	RSVP object	Reference
Source TNA Address	GENERALIZED_UNI/Source TNA Address	Section 12.5.2.3.1
Source Port ID	IPv4_IF_ID_RSVP_HOP	Section 12.5.2.9
Source Generalized Label	GENERALIZED_LABEL	[GMPLS RSVP-TE]
Destination TNA Address	GENERALIZED_UNI/Destination TNA	Section 12.5.2.3.5
Destination Port ID	GENERALIZED_UNI/Egress Label	Section 12.5.2.3.10
Destination Generalized Label	GENERALIZED_UNI/Egress Label	Section 12.5.2.3.10
Local Connection ID	(UNI_IPv4_SESSION, LSP_TUNNEL_IPv4_SENDER_TEMPLATE) or (UNI_IPv4_SESSION, LSP_TUNNEL_IPv4_FILTER_SPEC)	Section 12.5.2.2/12.5.2.1
Contract ID	POLICY_DATA	Section 13
SONET/SDH Traffic Parameters	SONET/SDH_SENDER_TSPEC, SONET/SDH_FLOWSPEC	[GMPLS SONET]
Directionality	UPSTREAM_LABEL	[GMPLS RSVP-TE]
Payload	GENERALIZED_LABEL_REQUEST/G-PID	[GMPLS RSVP-TE]
Service Level	GENERALIZED_UNI, Service Level	Section 12.5.2.3.11
Diversity	GENERALIZED_UNI, Diversity	Section 12.5.2.3.9
Error Code	IPv4_ERROR_SPEC	Section 12.5.2.6
Connection Status		

Table 12-2: Mapping between UNI Attributes and RSVP Objects

12.4 UNI RSVP Signaling procedures

The RSVP protocol definitions in this section apply only for UNI signaling. There is no implied requirement that RSVP-based signaling be supported within the network. In fact, the UNI RSVP messages contain values as if they are used to setup two separate single-hop connections, one between the initiating UNI-C and UNI-N, and the other between the UNI-N and the terminating UNI-C. The network is assumed to provide coordination of signaling information between the initiating and the terminating side of the connection.

12.4.1 UNI Interfaces, Signaling Channel, Control Channels, Logical Port Identifier and Addressing

RSVP messages are exchanged over the UNI signaling channel. The signaling channel may be realized over one or more underlying IP control channels, as described in Section 6. The determination of the UNI control interfaces and the maintenance of the IP control channel are described in Section 8. The identification of connection endpoints is described in Section 7.

12.4.2 Sending UNI RSVP Messages

When a UNI-C (UNI-N) is sending a RSVP message, it MUST address the message directly to its UNI-N (UNI-C) peer. The peer's Node ID is used for this purpose (see Section 7). A node SHOULD use the simple IP encapsulation and the router-alert option MUST NOT be included in any RSVP messages. This is shown in Table 12.3.

Either GRE or IP-in-IP encapsulation MAY be used (by configuration) if simple IP encapsulation poses operational problems.

IP Header Values for UNI RSVP messages	
Version	4
Header Length	5
TOS	As defined in RFC 2205
Total Length	Message length
Flags	As defined in RFC 791
Fragment Offset	As defined in RFC 791
TTL	≥ 1
Protocol	46
Header Checksum	As defined in RFC 791
Source Address	UNI-C/UNI-N Node ID
Destination Address	UNI-C/UNI-N Node ID

Table 12-3: IP header for UNI RSVP messages

12.4.3 Receiving UNI RSVP Messages

A UNI-C or a UNI-N node SHOULD process a received RSVP message as specified in [RFC2205, RFC2209] only if all security validation procedures have been successfully performed. Specifically, a received RSVP PDU that fails security validation MUST be dropped and an ACK message MUST NOT be generated, even if an ACK was requested.

12.4.4 Reliable Messaging

To support reliable messaging across the UNI, UNI-C and UNI-N implementations MUST support the RSVP Refresh Reduction Extensions [RFC 2961]. In particular, the MESSAGE_ID object MUST be included in Path, PathTear, PathErr, Resv, ResvTear, ResvErr, ResvConf, and Srefresh messages. The Ack_Desired flag in a MESSAGE_ID object MUST be set in all RSVP trigger messages, PathErr, ResvErr, ResvConf, and Srefresh message, and MAY be set in RSVP refresh messages.

Message identification and acknowledgment are done on a per-hop basis. Each MESSAGE_ID object contains a message identifier. This identifier MUST uniquely identify a message with respect to a node's Node Identifier.

Failure to receive a MESSAGE_ID_ACK for a refresh message MUST NOT result in the deletion of the corresponding connection.

Note that ACK messages or MESSAGE_ID_ACK object MAY not appear in the exact manner as shown in the timing diagrams in this section.

12.4.5 Connection State Maintenance

RSVP takes a "soft state" approach to manage the connection state. RSVP soft state is created and periodically refreshed by Path and Resv messages. The state is deleted if no matching refresh messages arrive before the expiration of a "cleanup timeout" interval. State may also be deleted by an explicit PathTear, PathErr with Path_State_Removed flag, or ResvTear message.

UNI signaling maintains the soft state approach but requires explicit tear-down messages from the user. That is, connection deletion should normally be in response to an explicit tear-down request rather than soft-state timeout. Therefore, a state timeout occurring at a UNI-C or a UNI-N indicates a problem. In response to a state timeout, an explicit tear-down message MUST be sent, and an alarm may be reported.

The use of reliable messaging, via the MESSAGE_ID and MESSAGE_ID_ACK objects, does not remove the need to refresh connection states. To reduce the number of refresh messages, a node SHOULD use the summary refresh (Srefresh) message [RFC 2961] to refresh the connection states.

12.4.6 Reservation Style

RSVP reservation "styles" are defined in [RFC2205]. For UNI signaling, the Fixed Filter (FF) style MUST be used.

12.4.7 Local Connection Identification

A Local Connection Identifier is used to uniquely identify a connection at a UNI. Under RSVP signaling, this is realized using the combination of the UNI_IPv4_SESSION object and the LSP_TUNNEL_IPv4_SENDER_TEMPLATE object in Path, PathTear, and PathErr messages, and the combination of the UNI_IPv4_SESSION object and the LSP_TUNNEL_IPv4_FILTER_SPEC object in Resv, ResvTear, ResvErr, and ResvConf messages. The local connection identifier remains unmodified during the lifetime of a connection.

12.4.8 Connection Traffic Parameters

The traffic parameters of a connection are technology dependent and are encoded in the SENDER_TSPEC and the FLOWSPEC object classes. A UNI-N and a UNI-C node MUST support standard SONET/SDH traffic parameters defined in GMPLS SONET-SDH specification [GMPS SONET] and covered in Section 10. When the requested connection traffic parameters are not supported by the network, a PathErr message with the following error code SHOULD be generated by the network: "Routing Problem: Connection parameters not supported".

12.4.9 Connection Creation

To create a connection, a UNI-C node sends a Path message to its adjacent UNI-N node. The Path message SHALL include a GENERALIZED_LABEL_REQUEST object, which indicates that a label binding is requested for this connection. The traffic parameters (characteristics) of the connection are encoded in a SONET/SDH SENDER_TSPEC object in the Path message and a SONET/SDH FLOWSPEC object in the corresponding Resv Message. Figure 12-1 shows the timing diagram and message flow during successful connection creation.

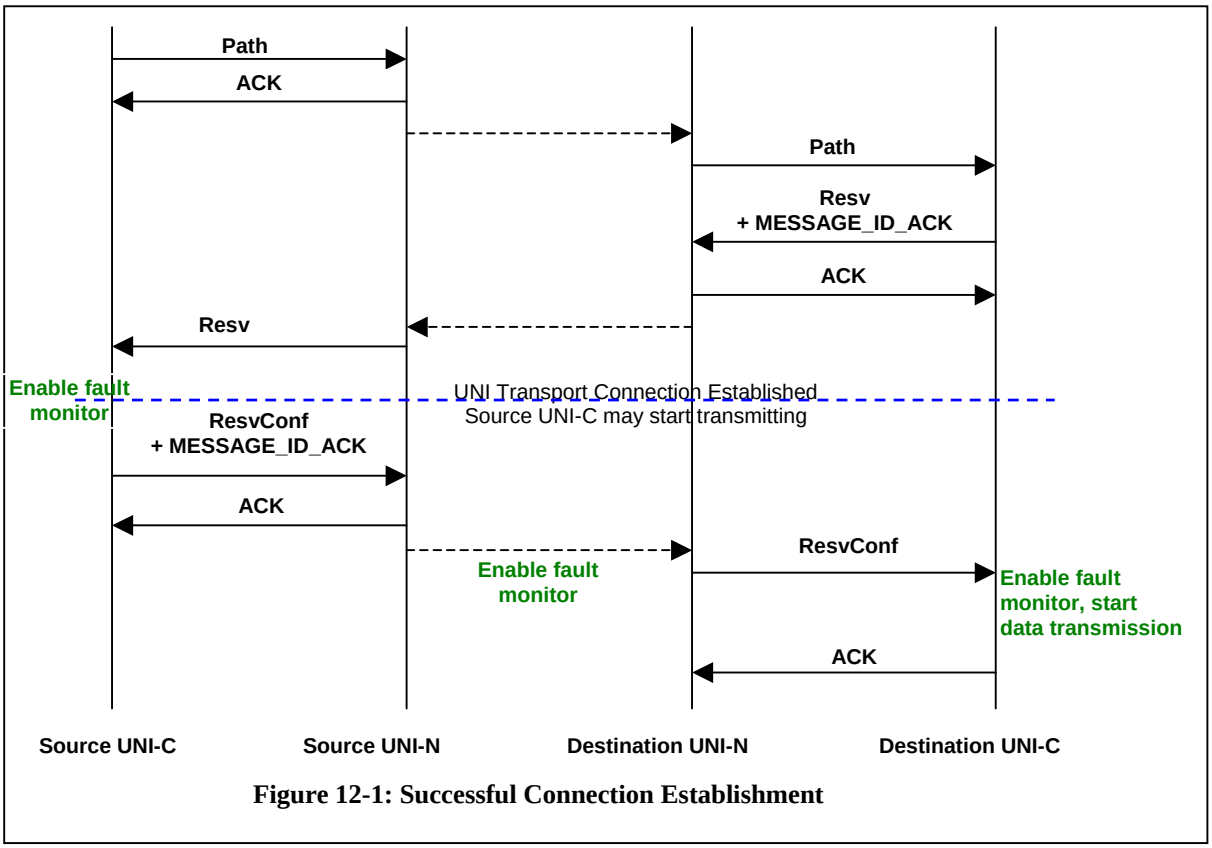


Figure 12-1: Successful Connection Establishment

The IPv4_IF_ID_RSVP_HOP object in a Path message specifies the interface over which the connection is to be established. The GENERALIZED_LABEL object in the Resv message specifies the allocated label(s) (on the interface) to be used by the connection.

To request a bi-directional connection, a UNI-C MUST insert an UPSTREAM_LABEL Object in the Path message to select the upstream label(s) for the connection.

If a node requires the GENERALIZED_LABEL and UPSTREAM_LABEL to be of the same value, it SHOULD include a LABEL_SET object such that the GENEALIZED_LABEL is limited to the same value as the UPSTREAM_LABEL. This is the typical configuration for SONET/SDH connections.

It can be assumed that the connection segment within the transport network has been established only when the UNI-N sends the Resv message to the source UNI-C. Therefore, to avoid loss of data, the source client should not start data transmission before the Resv message is received by the corresponding UNI-C and the destination client should not start data transmission before the ResvConf message is received by the corresponding UNI-C (if it had inserted a RESV_CONFIRM object in Resv message). The destination client should be ready to receive data before the corresponding UNI-C sends the Resv message, and source client should be ready to receive data before the corresponding UNI-C initiates the Path message. If a node monitors a connection, it should not raise a service affecting alarm until it has verified that the connection has been established end-to-end. This is shown by the dashed line in Figure 12-1.

Contention for labels may occur between two connection creation requests. The contention resolution procedure in this case is described in [GMPLS RSVP-TE].

A connection may not be successfully created due to resource unavailability, policy or reachability constraints. Figures 12-2, 12-3 and 12-4 illustrate timing diagrams and message flows for certain unsuccessful connection creation scenarios.

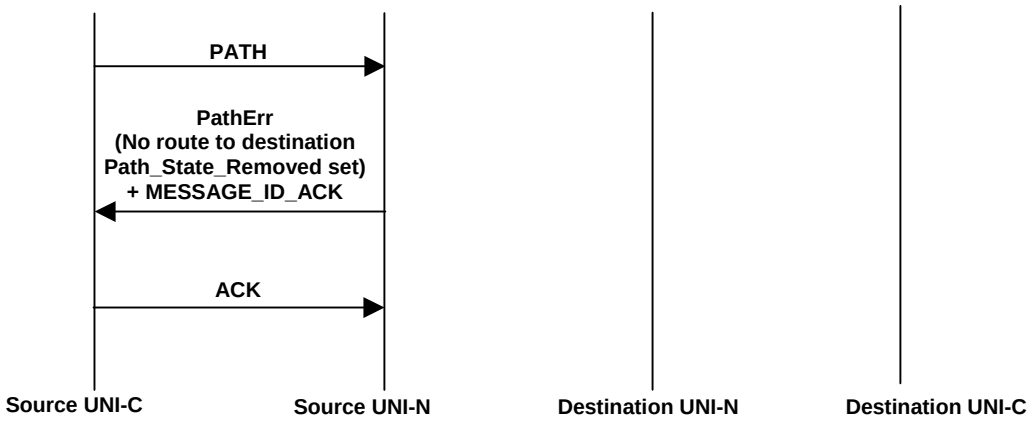


Figure 12-2: Connection Rejection by the Network using the Path_State_Removed Flag

A node rejecting a connection request SHOULD delete its own path state and set the Path_State_Removed flag in the PathErr message. A node receiving a PathErr with Path_State_Removed flags set, SHOULD NOT send a PathTear downstream. This is shown in Figures 12-2 and 12-4.

If the Path_State_Removed flag is not set in a PathErr message, the source UNI-C MUST decide whether to delete the connection explicitly or allow it to time out. To delete the connection explicitly, the UNI-C MUST send a PathTear message. A node that receives a PathTear, which 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. This is shown in Figure 12-3.

12.4.10 Connection Modification

UNI 1.0 does not support destructive connection modification. Modification of RSVP objects that do not change traffic parameters, e.g. ADMIN_STATUS object, MESSAGE_ID object, etc., is supported.

12.4.11 Connection Deletion

RSVP currently deletes 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 MUST be sent in 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 [GMPLS RSVP-TE] and shown in Figures 12-5 through 12-8.

If a UNI-C does not respond to a network initiated graceful deletion, the network SHOULD continue the message flow illustrated below the (red) dashed line in Figures 12-7 and 12-8.

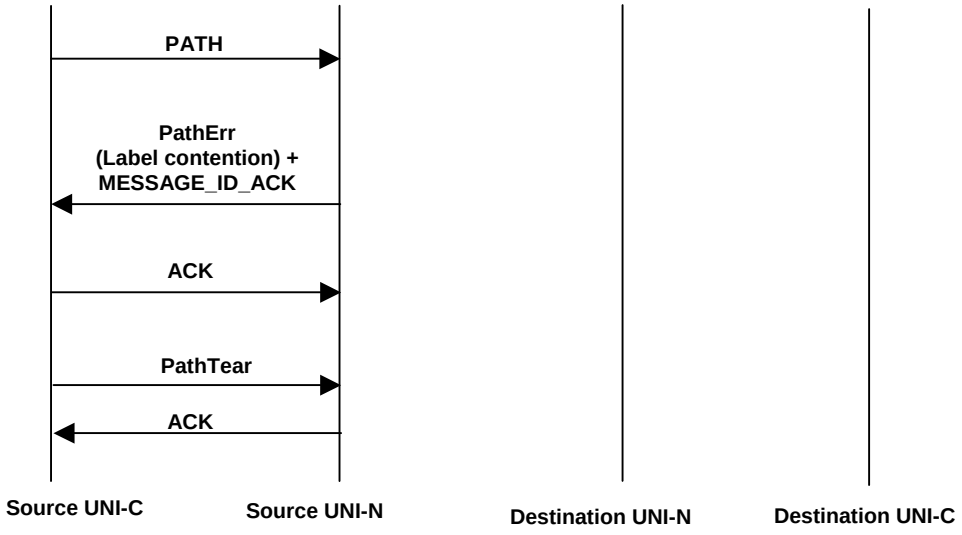


Figure12-3: Connection Rejection by the Network, without the Use of Path_State_Removed Flag

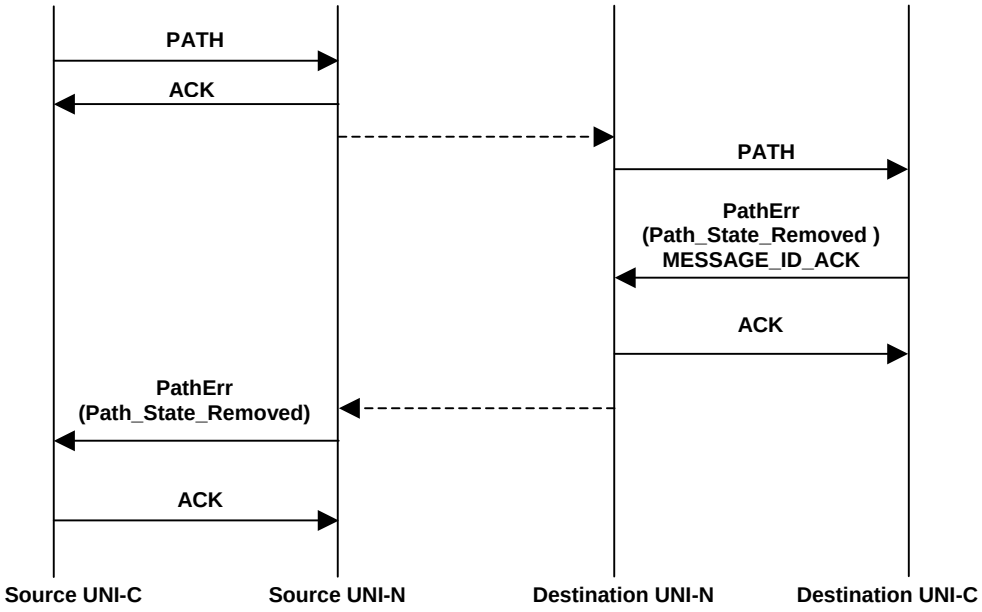


Figure12-4: Connection Set-Up Rejection by the Destination UNI-C

12.4.11.1 Forced Deletion

The forced deletion procedure should be used to handle scenarios that require unilateral deletion. In general this occurs due to a network-generated event that requires the connection to be deleted. For example,

- Internal network failures, which force the network to terminate connections.
- When the “Deletion In Progress” timer of an ADMIN_STATUS object expires.

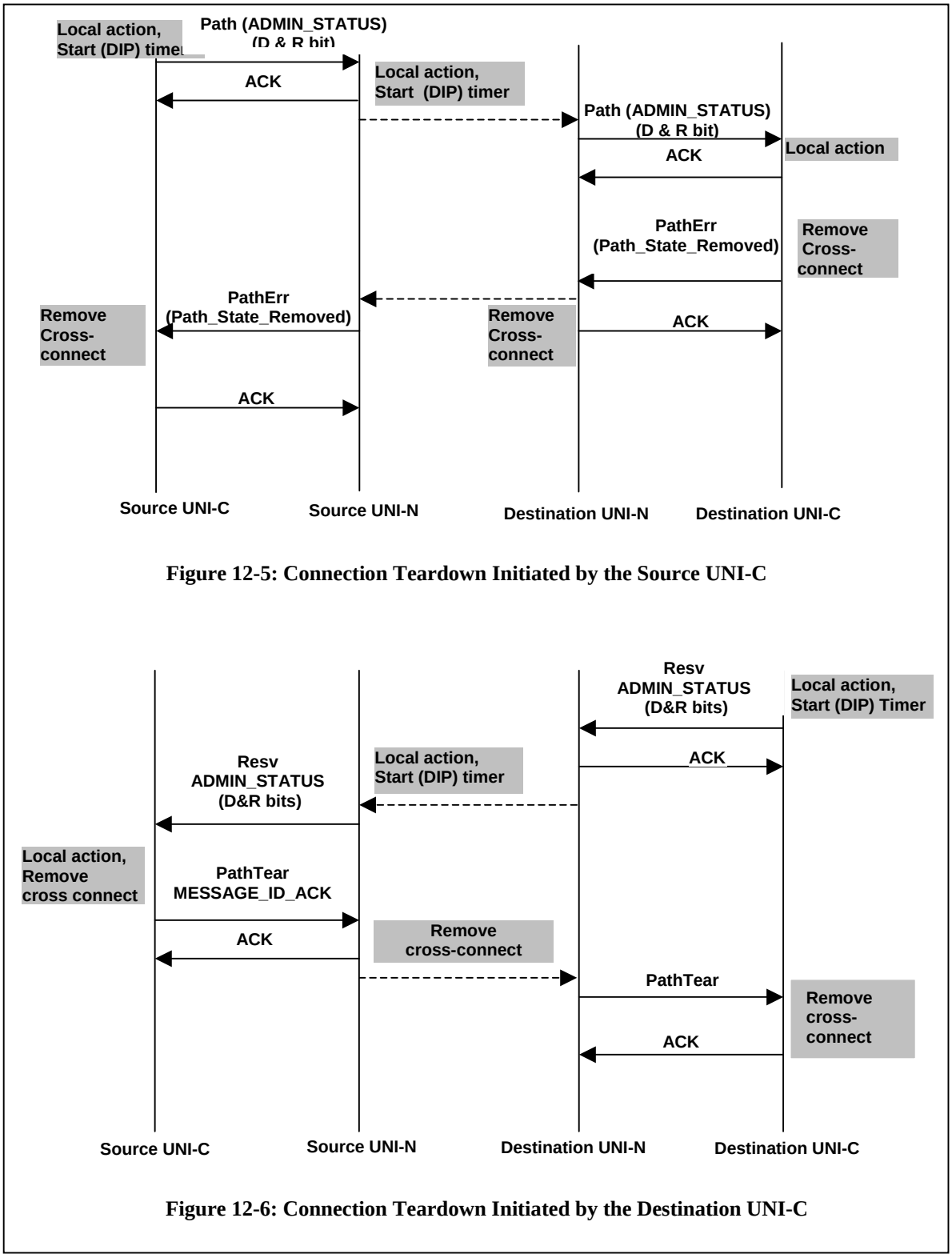


Figure 12-5: Connection Teardown Initiated by the Source UNI-C

Figure 12-6: Connection Teardown Initiated by the Destination UNI-C

A UNI-N initiates a forced deletion by sending a PathErr toward the source UNI-C and, simultaneously, a PathTear toward the destination UNI-C. The PathErr message sent to the source UNI-C has the

"Path_State_Removed" flag to indicate that the path state is deleted. In this case, a PathTear from the source UNI-C is not required to terminate the connection; in fact, such a PathTear would be discarded (but acknowledged) since Path state will have already been removed. The message flow for the forced deletion procedure is illustrated in Figure 12-9. This flow reflects the fact that connection tear-down can be initiated by a UNI-N unilaterally, without the consent of the other party.

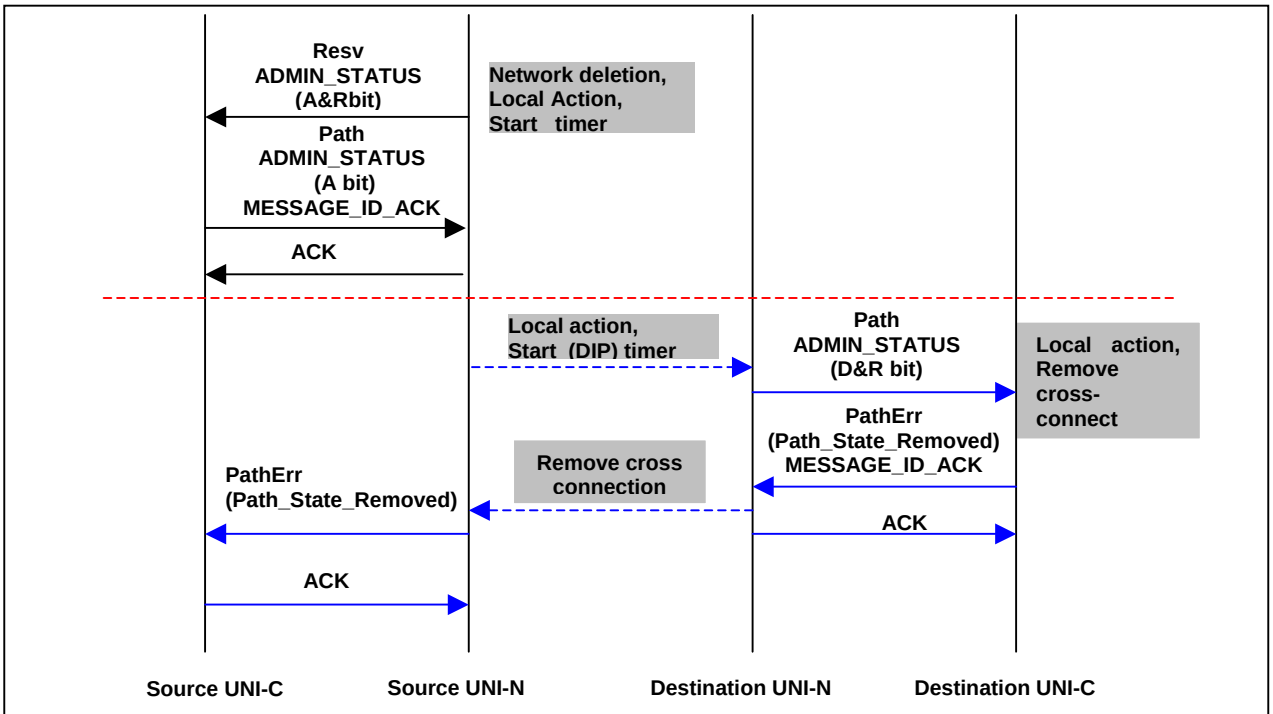


Figure 12-7: Connection Teardown Initiated by the Source UNI-N

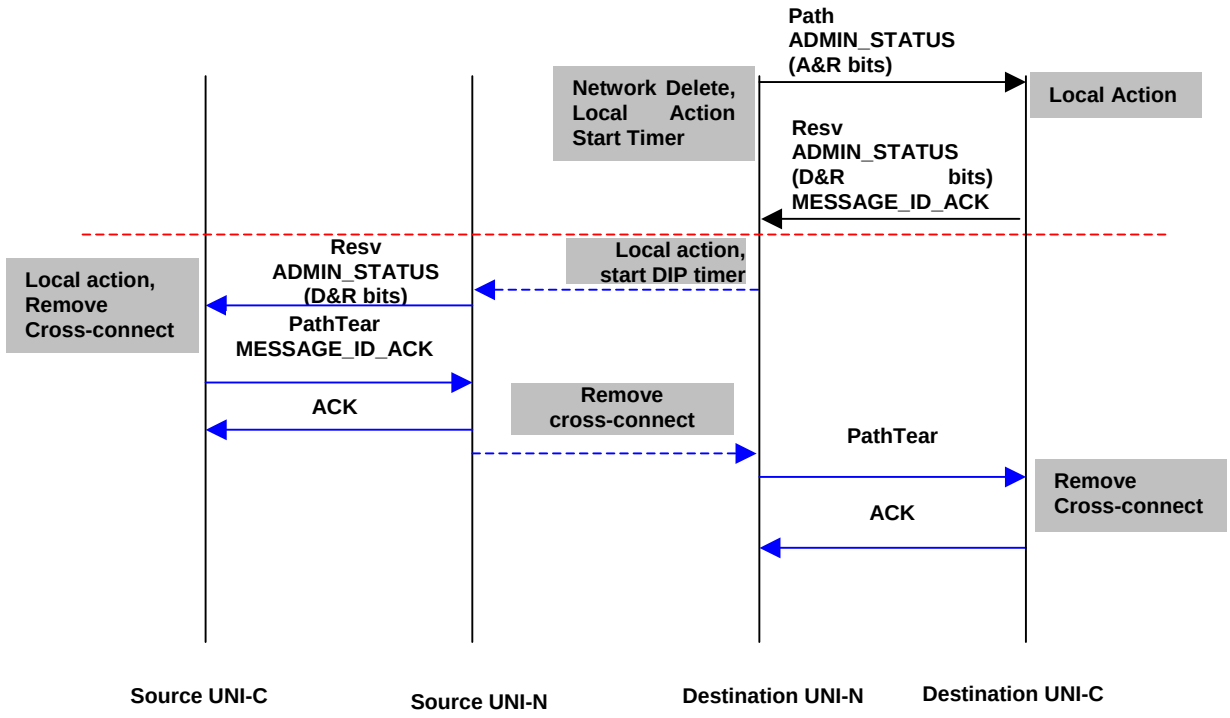


Figure 12-8: Connection Teardown Initiated by Destination UNI-N

To initiate a forced deletion, a source UNI-C sends a PathTear message to the source UNI-N and a destination UNI-C sends a PathErr with Path_State_Removed flag to the destination UNI-N. A forced deletion initiated by a PathErr with Path_State_Removed flag may be replaced by using the combination of a ResvTear and a PathTear message. A UNI-C node should initiate a forced deletion only when the "Deletion In Progress" timer expires or when the soft state times out.

12.4.12 Connection Status Enquiry And Response

The RSVP protocol periodically exchanges refresh messages to synchronize connection states between adjacent UNI-N and UNI-C nodes. This can be viewed as a continuous query and response process that keeps the states on both nodes synchronized. There is, therefore, no need for explicit query and response messages in order to find out a neighbor's connection state.

12.4.13 Signaling Channel Failure Detection and Recovery

Under the RSVP protocol, signaling messages relating to a data link are sent on the same link and the failure of a link or the control plane results in the eventual deletion of reservations made on the link. This, however, should not be the case under UNI signaling where the control plane is separate from the data plane and where the failure of the control plane does not imply data plane failure. Here, in particular, the failure of a signaling channel or control protocol entities must not result in the deletion of previously established connections.

To address this requirement, a node MUST support the fault handling procedure described in Section 9 of [GMPLS RSVP-TE], illustrated in Figures 12-10 and 12-11. Figure 12-10 illustrates the message flow in the case of recovery from a node failure (i.e., complete failure of the control plane). Figure 12-11 illustrates the message flow, based on Srefresh messages, between a pair of adjacent nodes in the case of recovery from signaling channel failures (i.e., where the signaling protocol entities did not fail).

The "self-refresh" procedure in Figures 12-10 and 12-11 refers to the behavior of a UNI-C or a UNI-N node which acts as if it is receiving periodic RSVP refresh messages from the neighbor during the failure situation. A UNI-C or a UNI-N stops the self-refresh behavior when RSVP adjacency is re-established or when the restart timer expires.

12.4.14 Data Plane Failure and Recovery

SONET/SDH provides rich features for failure detection, in particular the SONET/SDH overhead bytes allows a SONET/SDH node to detect transport layer failure. Recovery may be attempted at the node that detects the failure.

To allow different restoration and protection schemes within the network, a node should not delete a connection upon detecting a fault, but should continue to accept and send RSVP refresh messages until it receives explicit tear-down messages or the connection state times out.

12.5 RSVP Messages And Objects For UNI Signaling

This section describes the specific usage and procedures of RSVP/GMPLS RSVP-TE objects and messages that apply to UNI 1.0. Only objects, messages and behavior that are not already captured in standard IETF specifications, or for which specific details are necessary are described in this document. The standard behavior is captured in relevant IETF documents as referenced.

Table 12-4 and 12-5 show the complete list of messages and objects supported by UNI 1.0. RSVP trigger messages and ResvConf message have end-to-end significance and the network must relay these messages from the source UNI to the destination UNI and vice versa. Also, some objects must maintain the same value at the source and destination UNI-C. These are marked as end-to-end significant in Table 12-5.

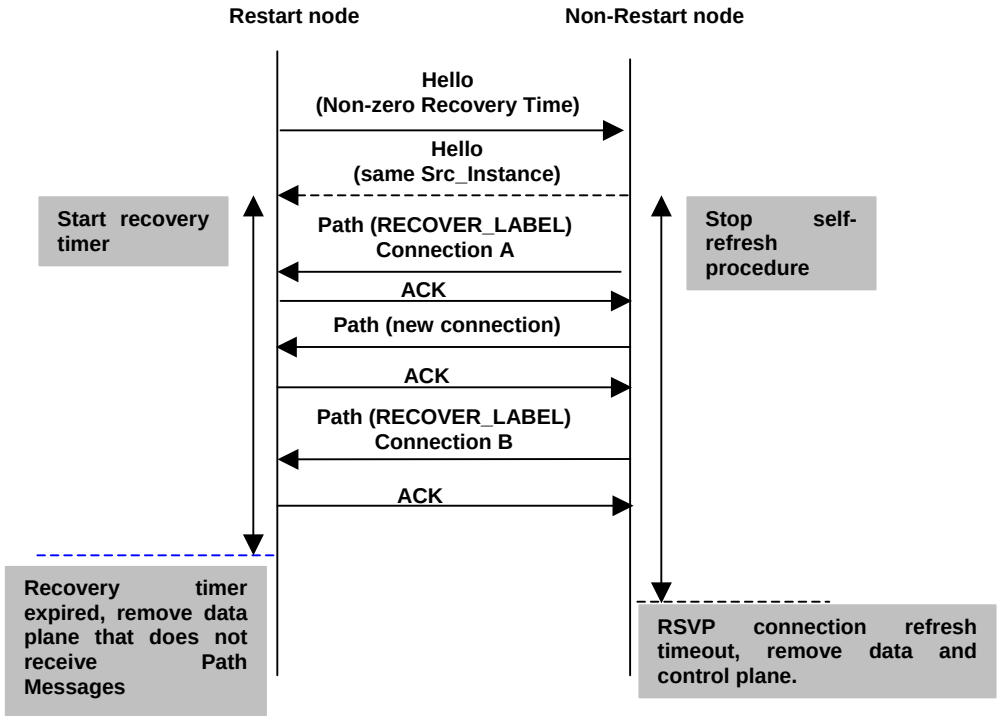


Figure 12-10: Recovery from Complete Failure

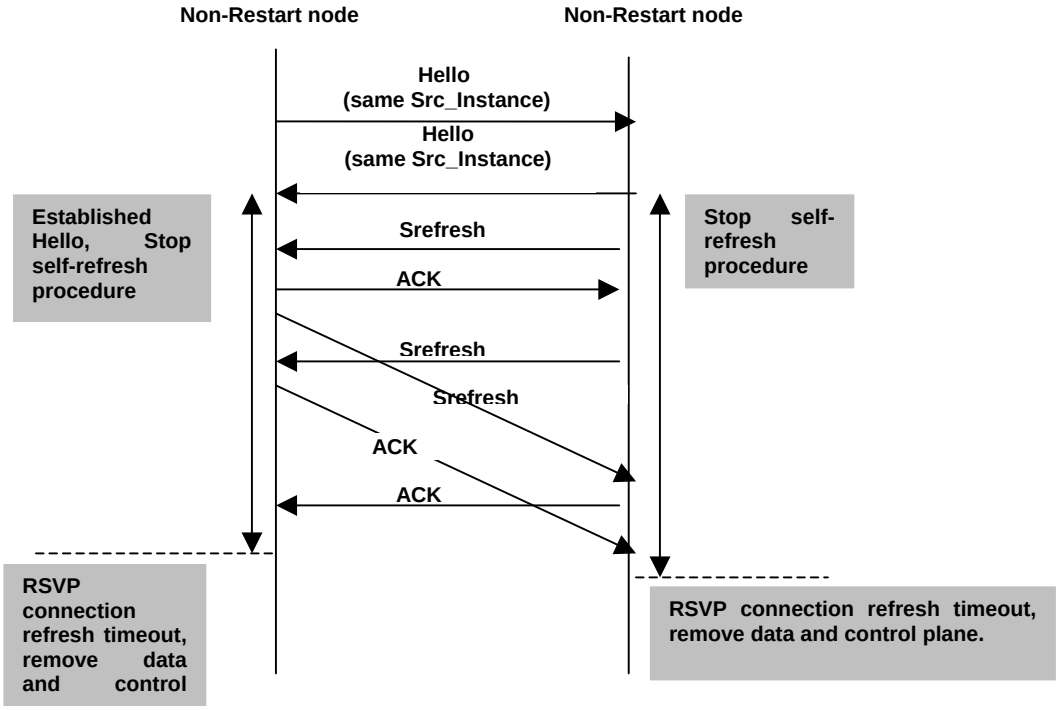


Figure 12-11: Recovery from Signaling Channel Failure

RSVP Messages	Direction	Message Type	Reference
Path	(Source) UNI-C→ UNI-N & (Dest) UNI-N → UNI-C	1	Section 12.5.1.3
PathTear	(Source) UNI-C→ UNI-N & (Dest) UNI-N → UNI-C	5	Section 12.5.1.5
PathErr	(Source) UNI-C→ UNI-N & (Dest) UNI-N → UNI-C	3	Section 12.5.1.4
Resv	(Source) UNI-C→ UNI-N & (Dest) UNI-N → UNI-C	2	Section 12.5.1.6
ResvErr	(Source) UNI-C→ UNI-N & (Dest) UNI-N → UNI-C	4	Section 12.5.1.8
ResvTear	(Source) UNI-C→ UNI-N & (Dest) UNI-N → UNI-C	6	Section 12.5.1.9
ResvConf	(Source) UNI-C→ UNI-N & (Dest) UNI-N → UNI-C	7	Section 12.5.1.7
Hello	UNI-N ↔UNI-C	20	[GMPLS RSVP-TE]
Ack	UNI-N ↔ UNI-C	13	[RFC 2961]
Srefresh	UNI-N ↔ UNI-C	15	[RFC 2961]
Bundle	UNI-N ↔ UNI-C	12	[RFC 2961]

Table 12-4: RSVP Messages Supported Under UNI 1.0

RSVP Object or Sub-Objects		End-to-End Significant?	C-Num/ C-Type	Reference
ACCEPTABLE _LABEL_SET		NO	10bbbbbb [TBA]/1	[GMPLS RSVP-TE]
ADMIN_STATUS		YES	11bbbbbb [TBA]/1	[GMPLS RSVP-TE]
SONET/SDH_FLOW_SPEC		YES	12/4	[GMPLS SONET]
LSP_TUNNEL_IPv4_FILTER_SPEC		NO	10/7	Section 12.5.2.7
GENERALIZED_LABEL		NO	16/2	[GMPLS RSVP-TE]
GENERALIZED_LABEL_REQUEST		YES	19/4 [TBA]	Section 12.5.2.4
GENERALIZED_UNI_ ATTRIBUTES	SOURCE _TNA	YES	11bbbbbb/1/1 (TBA)	Section 12.5.2.3.1
	DESTINATION_TNA	YES	11bbbbbb/1/2 (TBA)	Section 12.5.2.3.5
	DIVERSITY	NO	11bbbbbb/1/3 (TBA)	Section 12.5.2.3.9
	EGRESS_LABEL	NO	11bbbbbb/1/4 (TBA)	Section 12.5.2.3.10
	SERVICE_LEVEL	YES	11bbbbbb/1/5 (TBA)	Section 12.5.2.3.11
HELLO_REQUEST		NO	22/1	[GMPLS RSVP-TE]
HELLO ACK		NO	22/2	[GMPLS RSVP-TE]
INTEGRITY		NO	4/1	Section 13
IPv4_ERROR_SPEC, error code and value		YES	6/1	Section 12.5.2.6
IPv4_IF_ID_RSVP_HOP		NO	3/3	Section 12.5.2.9
IPv4_RESV_CONFIRM		NO	15/1	Section 12.5.2.5
LABEL_SET		NO	0bbbbbbb/1 (TBA)	[GMPLS-RSVP-TE]
MESSAGE ID		NO	23/1	[RFC 2961]
MESSAGE ID_ACK		NO	24/1	[RFC 2961]
MESSAGE ID_NACK		NO	24/2	[RFC 2961]
MESSAGE ID_LIST		NO	25/1	[RFC 2961]
POLICY_DATA		YES	14/1	Section 13
RECOVER_LABEL		NO	10bbbbbb/1 [TBA]	[GMPLS RSVP-TE]

RESTART_CAP	NO	10bbbbbb/1 (TBA)	[GMPLS RSVP-TE]
LSP_TUNNEL_IPv4_SENDER_TEMPLATE	NO	11/7	Section 12.5.2.1
SONET/SDH_SENDER_TSPEC	YES	12/4[TBA]	[GMPLS SONET]
STYLE	YES	8/1	[RFC 2205]
TIME_VALUE	NO	5/1	[RFC 2205]
UNI_IPv4_SESSION	NO	1/11 [TBA]	Section 12.5.2.2
UPSTREAM_LABEL	NO	0bbbbbbb/2 (TBA)	[GMPLS RSVP-TE]

Table 12-5: Summary of UNI RSVP Objects

12.5.1 RSVP Messages for UNI Signaling

12.5.1.1 RSVP Common Message Hearer

The flag field of RSVP common header MUST be set to 1, to indicate support of the Bundle and Srefresh messages.

12.5.1.2 Hello Message (Msg Type = 20 [RSVP-TE])

The Hello message is for node failure detection. It has the following format:

<Hello message> ::= <Common Header> [<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<HELLO>
<RESTART_CAP>

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

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

The Path message is used for connection creation. The format of the Path message is as follows:

<Path Message> ::=
<Common Header>
[<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<UNI_IPv4_SESSION> <IPv4_IF_ID_RSVP_HOP>
<TIME_VALUES>
<GENERALIZED_LABEL_REQUEST> [<LABEL_SET> ...]
[< ADMIN_STATUS>]
<Generalized UNI>
[<POLICY_DATA> ...]
<sender descriptor>

<Generalized UNI> ::=
<Common Object Header>
<DESTINATION_TNA>
<SOURCE_TNA>
[<DIVERSITY> …]
[<SERVICE_LEVEL>]
[<EGRESS_LABEL>]

The format of the sender descriptor for a unidirectional connection is:

<sender descriptor> ::=
<LSP_TUNNEL_IPv4_SENDER_TEMPLATE> <SONET/SDH_SENDER_TSPEC>
[< RECOVER_LABEL >]

The format of the sender descriptor for a bi-directional connection (default under UNI 1.0) is:

<sender descriptor> ::=
<LSP_TUNNEL_IPv4_SENDER_TEMPLATE> <SONET/SDH_SENDER_TSPEC>
<UPSTREAM_LABEL> [< RECOVER_LABEL >]

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

The PathErr message is used to report errors and for connection deletion. The format of the UNI PathErr message is shown as follows:

<PathErr message> ::= <Common Header> [<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<UNI_IPv4_SESSION>
<IPv4_ERROR_SPEC>
[<ACCEPTABLE_LABEL_SET>]
[<POLICY_DATA> …]
<sender descriptor>

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

The PathTear message is used when a connection is deleted by the source UNI-C. The format of the UNI PathTear message is as follows:

<PathTear Message> ::= <Common Header> [<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<UNI_IPv4_SESSION> <IPv4_IF_ID_RSVP_HOP>
<sender descriptor>

<sender descriptor> ::= (see earlier definition)

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

The Resv message is used for connection creation. A destination UNI-C SHOULD stop inserting the ResvConfirm Object in the Resv message if it receives a matching ResvConf message. The format of the UNI Resv message is as shown below:

<Resv Message> ::= <Common Header> [<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<UNI_IPv4_SESSION> < IPv4_IF_ID_RSVP_HOP >
<TIME_VALUES>
[<IPv4_RESV_CONFIRM>]
[<ADMIN_STATUS>]
[<POLICY_DATA> ...]
<STYLE>
<FF flow descriptor>

<FF flow descriptor> ::=
<SONET/SDH_FLOWSPEC> <LSP_TUNNEL_IPv4_FILTER_SPEC>
<GENERALIZED_LABEL>

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

The ResvConf message SHOULD be sent downstream from the source UNI-C to acknowledge the receipt of a Resv message that includes a RESV_CONFIRM Object. Specifically, under UNI 1.0, 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. The network MUST relay the ResvConf message from source UNI-N to destination UNI-N. The format of the UNI ResvConf message is shown below:

<ResvConf message> ::= <Common Header> [<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<UNI_IPv4_SESSION> <IPv4_ERROR_SPEC>
<IPv4_RESV_CONFIRM>
<STYLE> <FF flow descriptor >

12.5.1.8 ResvErr Message (Msg Type = 4 [RFC2205])

The ResvErr message is used for reporting reservation errors. The format of the UNI ResvErr message is as follows:

<ResvErr message> ::= <Common Header> [<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<UNI_IPv4_SESSION> <IPv4IF_ID_RSVP_HOP>
<IPv4_ERROR_SPEC>
[<ACCEPTABLE_LABEL_SET>]
[<POLICY_DATA> ...]
<STYLE> <FF flow description>

12.5.1.9 ResvTear Message (Msg Type = 6 [RFC2205])

The ResvTear message is supported by UNI 1.0 to be compatible with RSVP. The format of the UNI ResvTear message is shown below:

<ResvTear Message> ::=
<Common Header> [<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<UNI_IPv4_SESSION> <IPv4_IF_ID_RSVP_HOP>
<STYLE>
<FF flow description >

<FF flow description > ::= (see earlier list)

12.5.1.10 Srefresh Message (Msg Type = 15 [RFC2961])

The format of the UNI Srefresh Message is shown below:

<Srefresh message> ::= <Common Header> [<INTEGRITY>]
[[<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
<MESSAGE_ID>
<srefresh list>

<srefresh list> ::= <MESSAGE_ID_LIST>
[<srefresh list>]

12.5.2 UNI RSVP Objects Format

This section describes the RSVP objects that require specific behavior.

12.5.2.1 LSP_TUNNEL_IPv4_SENDER_TEMPLATE Object (Class-Num = 11 [RSVP-TE])

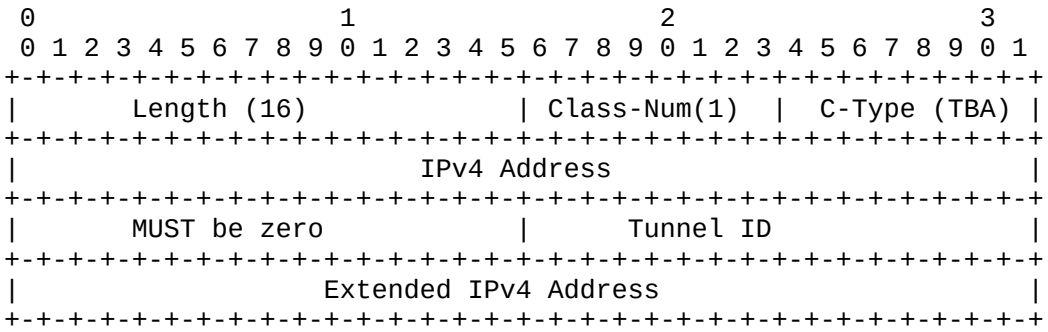
The format of the LSP_TUNNEL_IPv4_SENDER_TEMPLATE object is defined in [RSVP-TE]. For UNI 1.0, the IPv4 Tunnel sender address SHOULD be set to the source UNI-C's Node ID at the source UNI and SHOULD be set to the destination UNI-N's Node ID at the destination UNI. The LSP ID is assigned by the sender of the Path message and it remains constant during the life of a connection.

The combination of the LSP_TUNNEL_IPv4_SENDER_TEMPLATE object and UNI_IPv4_SESSION object MUST uniquely identify a connection at a local UNI.

12.5.2.2 UNI_IPv4_SESSION Object (Class-Num = 1 [RSVP-TE])

UNI_ IPv4_SESSION Object [RSVP-TE] has the following format:

- UNI_ IPv4_SESSION object: Class = 1, C-Type = TBA



IPv4 Address: This MUST be set to the source UNI-N's Node ID at the source UNI and it MUST be set to the destination UNI-C's Node ID at destination UNI.

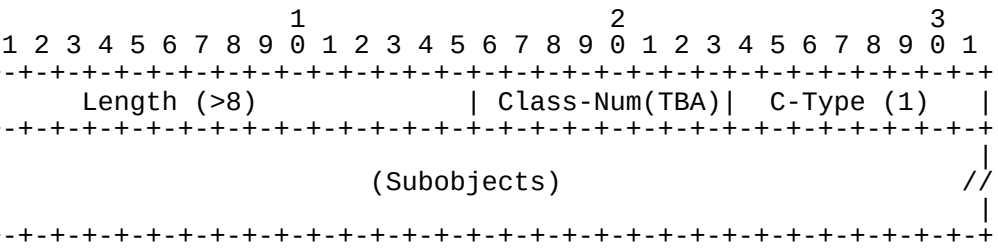
Tunnel ID: A 16-bit identifier, assigned by the sender of the Path message. This ID remains constant during the life of a connection.

Extended IPv4 address: This MUST be set to the Node ID of source UNI-C at the source UNI and it MUST be set to the Node ID of the destination UNI-N at the destination UNI.

The combination of the LSP_TUNNEL_IPv4_SENDER_TEMPLATE object and UNI_IPv4_SESSION object MUST uniquely identify a connection at a local UNI and 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".

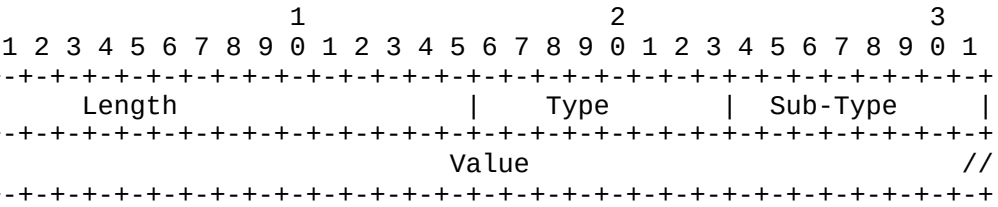
12.5.2.3 GENERALIZED_UNI Object (Class-Num=11bbbbbb (TBA))

UNI specific attributes are specified via the GENERALIZED_UNI object. The GENERALIZED_UNI object has the following format:



Subobjects:

The contents of a GENERALIZED_UNI object are a series of variable-length data items. The common format of the sub-objects is shown below:



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

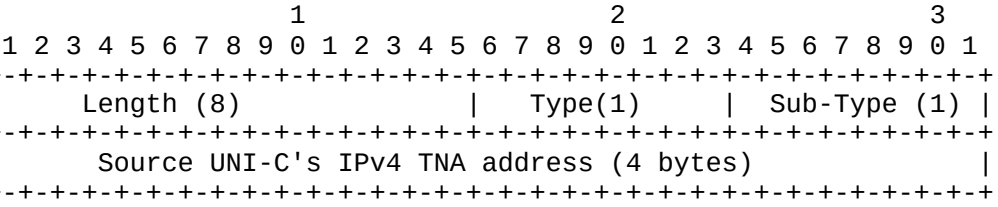
When the source or destination address specified cannot be serviced due to policy reasons, an error message with the error code “Policy control failure: Unauthorized sender,” or “Policy control failure: Unauthorized receiver”, respectively.

12.5.2.3.1 Source TNA Address Sub-Object (Type =1)

The source TNA address sub-object contains the Source UNI-C's TNA address. It may be in one of the three formats, i.e. IPv4, IPv6, or NSAP. If a GENERALIZED_UNI object contains more than one source TNA address sub-object, only the first is meaningful and all others MUST be ignored.

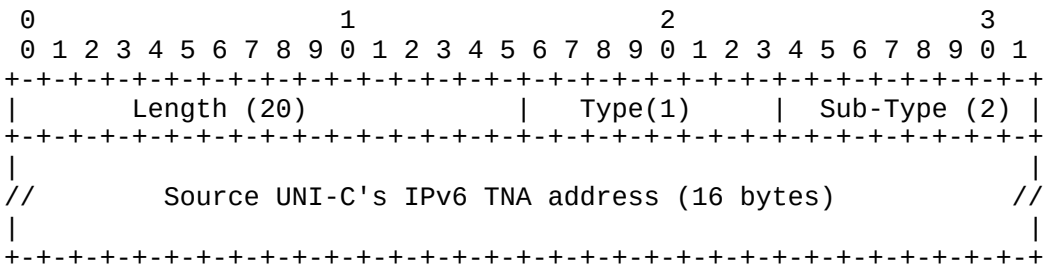
12.5.2.3.2 Source IPv4 TNA Address (Type=1, Sub-Type = 1)

The source IPv4 TNA address sub-object has the following format:



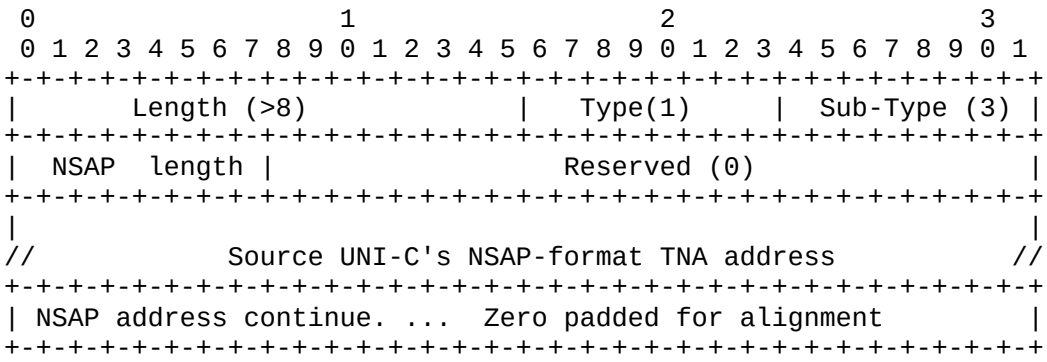
12.5.2.3.3 Source IPv6 TNA Address Sub-Object (Type=1, Sub-Type=2)

The source IPv6 TNA address sub-object has the following format:



12.5.2.3.4 Source NSAP TNA Address Sub-Object (Type=1, SubType = 3)

The source NSAP TNA address sub-object has the following format.



NSAP Length (8 bits): The length of source NSAP in bytes.

Source NSAP-format TNA address: Variable length field. Structured according to ISO/IEC 8348, 1993. Identical to ITU X.213, 1992.

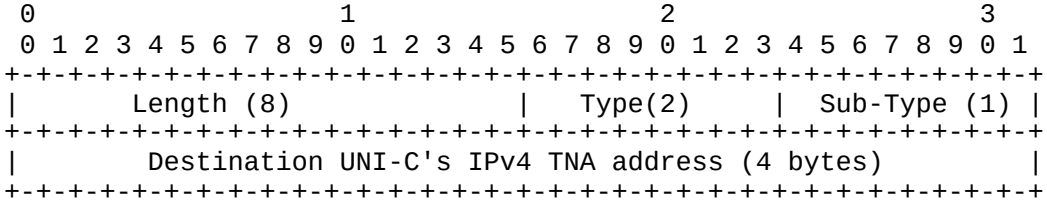
12.5.2.3.5 Destination TNA Address Sub-Object (Type =2)

The destination TNA address sub-object contains the destination UNI-C's TNA. It may be in one of the three formats, i.e. IPv4, IPv6, or NSAP. If a GENERALIZED_UNI object contains more than one destination TNA address sub-object, only the first is meaningful and all others MUST be ignored.

If a destination TNA address is unknown or not reachable, a signaling node SHOULD generate a PathErr message with error code “Routing problem: no route available toward destination”.

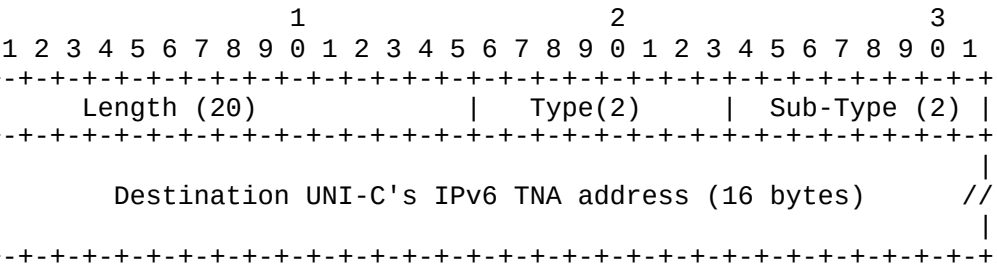
12.5.2.3.6 Destination IPv4 TNA Address Sub-Object (Type =2, Sub-Type = 1)

The destination IPv4 TNA address sub-object has the following format:



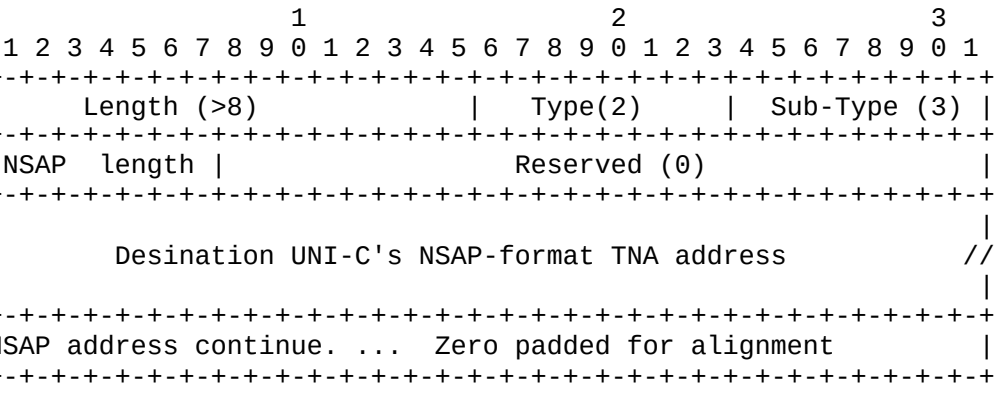
12.5.2.3.7 Destination IPv6 TNA Sub-object (Type =2, Sub-Type =2)

The destination IPv6 TNA address sub-object has the following format:



12.5.2.3.8 Destination NSAP-format TNA Address Sub-Object (Type =2, Sub-Type=3)

The destination NSAP TNA address sub-object has the following format:

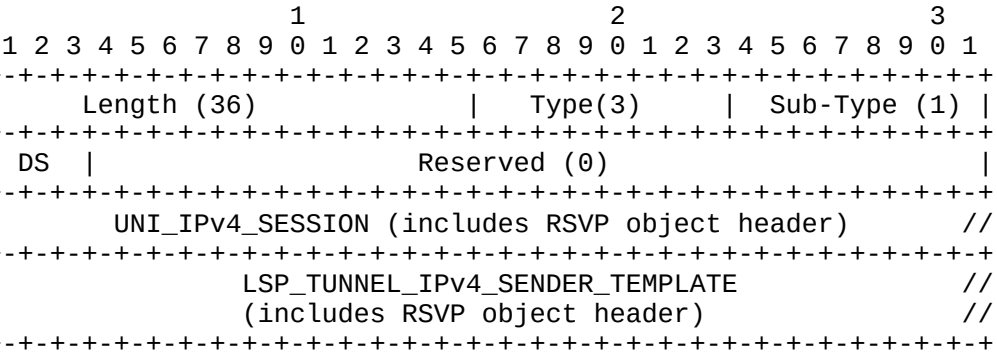


NSAP length (8 bits): The NSAP in bytes.

Destination NSAP-format TNA: Variable length field. Structured according to ISO/IEC 8348, 1993. Identical to ITU X.213, 1992.

12.5.2.3.9 Diversity Sub-Object (Type= 3, Sub-Type = 1)

The Diversity sub-object is a UNI defined object to specify the physical diversity required for a new connection. It carries the local connection identifier of an existing connection, and it may be carried in the Path message. Multiple instances of the diversity sub-object may be used. The sub-object has the following format:



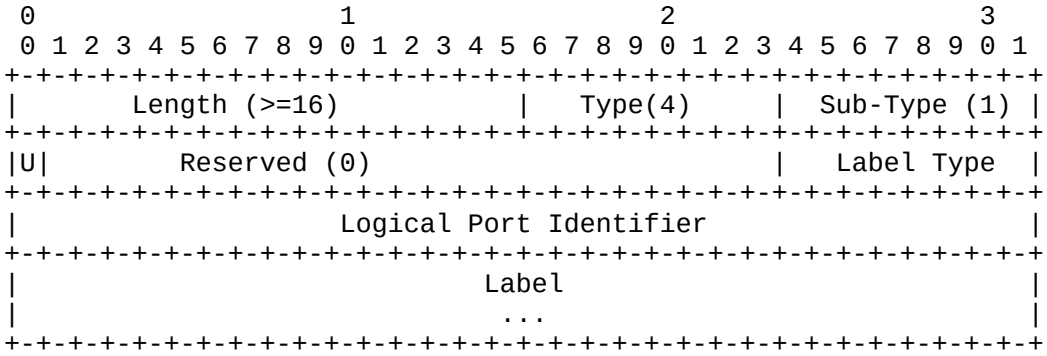
DS indicates the diversity requirement as following:

1 - Node diverse. The new connection SHALL NOT use any nodes that are on the path of the listed connection.
2 - Link Diverse. The new connection SHALL NOT use any links that are on the path of the listed connection.
3 - Shared Risk Link Group diverse. The new connection SHALL NOT use any link that have the same SRLG of the listed connection.
4 - Shared Path. The new connection SHALL use the same links as the listed connection.
0, 5-15 Reserved.

A node MUST report error “Routing Problem: Diversity not available” if it cannot provide the requested diversity or if it receives a diversity object specifying an unsupported value

12.5.2.3.10 Egress Label Sub-Object (Type=4, Sub-Type=1)

The Egress label Sub-Object is a UNI defined object and is used to specify the port, and label(s), to be used at the destination UNI. Up to two Egress label sub-objects MAY be included. It has the following format:



Logical Port Identifier:

This field indicates a logical port identifier assigned at the destination UNI-C. It is used to select a link at the destination UNI. The identified link MUST be used to allocate resources for the connection.

U-bit:

This bit indicates the direction of the label and the logical port ID. It is 0 for the downstream label and port ID and 1 for the upstream label and port ID. It is only used on bi-directional connections.

Label:

This field identifies the label to be used. The format of this field is identical to the one used by the Label field in GMPLS SONET/SDH Label [GMPLS SONET].

The following SHOULD result in a “Bad EXPLICIT_ROUTE object” error:

- If the Logical Port Identifier does not identify a valid link at the destination.
- If the U-bit is set when requesting a unidirectional connection set-up
- If there are two label sub-objects with the same U-bit values

A node examines one sub-object for unidirectional connections and two sub-objects for bi-directional connections. If the U-bit of the sub-object being examined is clear (0) then value of the label is copied into a new Label_Set object. This Label_Set object MUST be included in the corresponding outgoing Path message.

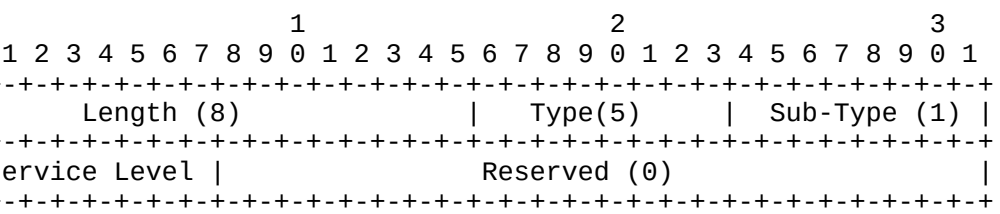
If the U-bit of the sub-object being examined is set (1) then the value of the label is to be used for upstream traffic associated with the bi-directional LSP. If this label is not acceptable then a “Bad

EXPLICIT_ROUTE object" error SHOULD be generated. If the label is acceptable then the label is copied into a new Upstream Label object. This Upstream Label object MUST be included on the corresponding outgoing Path message.

After processing is complete, the label sub-objects are removed from the GENERALIZED_UNI object.

12.5.2.3.11 Service Level Sub-Object (Type=5, Sub-Type=1)

The Service Level sub-object specifies an integer within the range 0-255 (called the "service level" (see Section 10)). Each service level corresponds to carrier predefined characteristics, such as type of restoration (e.g. unprotected, 1+1 protection), connection setup and hold priorities, reversion strategies for the connection after failures have been repaired, and retention strategies. The sub-object has the following format.



A node MUST report error "Routing Problem: Service level not available" if it receives a service level TLV specifying an unsupported value.

12.5.2.4 GENERALIZED_LABEL_REQUEST Object (Class-Num = 19 [GMPLS RSVP-TE])

The format of the GENERALIZED_LABEL_REQUEST object is defined in GMPLS RSVP-TE. For UNI 1.0, a node MUST support SONET and SDH LSP encoding types and TDM Switching Type.

12.5.2.5 IPv4_RESV_CONFIRM Object (Class-Num = 15, [RFC2205])

The format of the IPv4_RESV_CONFIRM object is defined in [RFC 2205]. For UNI 1.0, the IPv4 receiver address is set to the destination UNI-C's Node ID at the destination UNI, and it is set to the source UNI-N's Node ID at the source UNI.

12.5.2.6 IPv4_ERROR_SPEC Object (Class-Num = 6 [RFC2205])

The format of the IPv4_ERROR_SPEC object is defined in [RFC2205]. For UNI 1.0, the IPv4 Error Node Address is set to the Node ID of the UNI-C or the UNI-N which reported the error. UNI-N and UNI-C entities MUST support InPlace, NotGuilty, and Path_State_Removed flags.

12.5.2.7 LSP_TUNNEL_IPv4_FILTER_SPEC Object (Class-Num = 10 [RSVP-TE])

The LSP_TUNNEL_IPv4_FILTER_SPEC, combined with the UNI_IPv4_SESSION object, is used to uniquely identify a connection The format of this object is identical to the LSP_TUNNEL_IPv4_SENDER_TEMPLATE object.

12.5.2.8 MESSAGE_ID Object (Class-Num = 23 [RFC2961])

The format of MESSAGE_ID object is defined in [RFC296]. For UNI 1.0, the Ack_Desired flag in a MESSAGE_ID object MUST be set in trigger messages, PathErr, ResvErr and ResvConf messages, and in Srefresh message in order to support reliable messaging. The Ack_Desired flag MAY be set in a refresh message. Lack of acknowledgement of a refresh message at a node MUST NOT result in deletion of a connection.

12.5.2.9 IPv4_IF_ID_RSVP_HOP Object (Class-Num = 3, [GMPLS RSVP-TE])

The format of the IPv4_IF_ID_RSVP_HOP object is defined in GMPLS RSVP-TE. For UNI 1.0, the IPv4_IF_ID_RSVP_HOP object MUST be used to select the data link where a connection's resource should be allocated. UNI 1.0 SONET/SDH data links are modeled as multiplex capable bundled link. Therefore, a node MUST specify both the COMPONENT_IF_DOWNSTREAM and COMPONENT_IF_UPSTREAM TLVs for a bi-directional connection.

A node's Node identifier is used to fill an IP address field in the IPv4 IF_ID_RSVP_HOP object.

The logical port identifier, described in Section 7, should be used to set the value of an interface ID in the TLV.

12.6 RSVP Code Points

UNI signaling using RSVP defined in this section utilizes a number of new objects. Many of these objects have been defined as part of the GMPLS RSVP-TE signaling specification [GMPLS RSVP-TE] and some new objects have been defined specifically for UNI signaling. All these objects must have unique RSVP class number and class type values assigned. Furthermore, notification messages defined in this section have error codes for which unique values must be assigned. These assignments are performed by the Internet Assigned Numbers Authority (IANA) based on policies that can be found in [RFC2434]. *For many of the new GMPLS RSVP-TE objects, the class numbers and class types have been selected by the protocol designers but not officially assigned by the IANA. For some, the class numbers and class types have not been selected yet.* These are all highlighted and marked To be Assigned (TBA) in Table 12-2. Until these code points are assigned, interoperable implementations must use commonly agreed upon temporary code points and they should be prepared to comply with the code points finally assigned.

The following table summarizes the error codes specific to UNI signaling, as defined in this section. A number of other error codes generally applicable to RSVP and RSVP-TE signaling have been defined in [RFC2205] and [RSVP-TE].

Error Description	Error code/Value	Reference
"Routing problem: no route available toward destination". (Section 12.5.2.3.5)	24/5	[RSVP-TE]
"Routing Problem: Diversity not available" (Section 12.5.2.3.9)	24/100 (TBA)	
"Bad EXPLICIT_ROUTE object," (Section 12.5.2.3.10)	24/1	[RSVP-TE]
"Routing Problem: Service level not available" (Section 12.5.2.3.11)	24/101 (TBA)	
"Routing Problem: Invalid/Unknown connection ID" (Section 12.5.2.2)	24/102 (TBA)	
"Routing Problem: Connection parameters not supported" (Section 12.4.8)	21/02	[RFC2205]
"Policy Control Failure: Unauthorized sender" (Section 12.5.2.3)	2/100 (TBA)	
"Policy Control Failure: Unauthorized receiver" (Section 12.5.2.3)	2/101 (TBA)	

13 UNI Policy and Security Considerations

13.1 UNI Policy Control

The transport network must provide appropriate mechanisms to ensure accurate and authorized usage of network resources and client accountability. Collectively, these mechanisms are often referred to as *policy control*. Policy-based criteria are applied in addition to resource availability considerations when deciding whether a connection request can be accommodated within the transport network. Policy control rules may define conditions on parameters such as source and destination addresses, priorities, bilateral agreements among service providers, time-of-day constraints, cost constraints, etc. Policy control is also commonly associated with the mechanisms required to perform accounting. Upon initial deployment, policy control might employ simple rules, like for example, *"approve all requests on behalf of a given user group received from a given UNI-C agent, if the identity of the requestor can be verified"*. As more experience is gathered from initial deployments, it is envisioned that policy rules will become increasingly more sophisticated.

In order to support policy control, two main architectural entities are needed: the Policy Decision Point (PDP), where policy decisions are made and the Policy Enforcement Point (PEP), where policy decisions are actually enforced. The PEP resides within a transport network node, such as an OXC. The location of the PDP, however, depends on multiple factors, such as:

- The complexity of policy rules, including the computational load, type of software support and data access required. For example a policy might require complex cryptographic operations not supported within an OXC or access to a credit database which is physically located on a remote server. In this case, the PDP should be located in a remote server.

- The frequency of events requiring policy decisions. For example, connection set-up requests might be received infrequently thus reducing the computational complexity on the PDP.

- The intelligence and flexibility of the transport networking device. A sophisticated and easily upgrade-able transport network node is a better candidate to host the PDP.

The combination of the above factors determines whether the PDP should be co-located with the UNI-N agent or implemented on a remote policy server. If an external policy server is employed, a standardized protocol should be used for the communication between the PEP and PDP. This allows management of multiple PEPs from a single PDP and facilitates the integration of (standard) policy servers with multiple transport network elements. The COPS protocol [RFC2748] has been standardized by the IETF for PEP-PDP communication.

Specification of the UNI protocol does not depend on the placement of PEP and PDP modules within the transport network. Whether an external PDP is needed depends on the above factors, namely the frequency and complexity of policy decisions and the processing capabilities of the transport network element. If an external PDP is required, it is recommended that the COPS protocol be employed.

As shown in Figure 13-1, the PDP may further access an external server or database, for example to retrieve policy rules, centrally stored accounting information, etc. These additional mechanisms are not being specified by the OIF.

13.2 Sample Policies Applicable to Connection Provisioning

This section discusses sample policies that could be employed for controlling connection provisioning across the UNI. It is included for informational purposes only and is not intended to suggest standardization

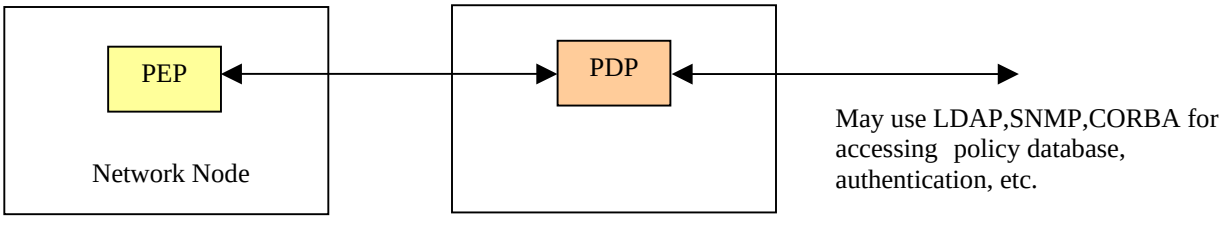


Figure 13-1: PEP and PDP

of any policies. In each of the cases presented, the information required to make the policy decision is also identified.

13.2.1 Time-Of-Day-Based Provisioning

A user's contract with a carrier allows placement of connection requests during a specific time of the day and/or day of the week. As an example, the connection could be used for data backup over a storage area network during night hours. In this scenario the UNI-N agent needs to verify whether connection requests are received during the contracted hours. All information required to make the policy decision (time of request receipt) is implicitly contained in the UNI signaling message.

13.2.2 Identity And Credit Verification For Connection Requestor

A carrier's carrier operates a *bandwidth exchange* which allows carriers to dynamically "trade" optical connections. The bandwidth exchange receives requests from a very large number of carriers. Multiple UNI-C agents, representing different carriers, might be contacting the same UNI-N agent in the carrier's carrier network. It is imperative for the carrier's carrier to be able to verify the identity of the originator of the request. Additionally, the ability of the requestor to pay for the service may also need to be verified. This might require information such as an account number or credit authorization. Policy based admission control at the UNI-N involves positive verification of the identity and creditworthiness of the requestor. In this scenario, the information required to make the policy decision might extend beyond that contained in mandatory attributes. Reference [RFC2752] describes identity representation when RSVP is used as the UNI 1.0 signaling protocol.

13.2.3 Usage-Based Accounting

Usage-based accounting can be supported using the contract identifier, which refers to the service contract of the connection owner (see Section 10). The contract identifier may be carrier specific, and it can be used for accounting, billing and SLA verification. Due to the sensitivity of the information contained, the contract identifier might be encrypted to protect the privacy of the originator.

13.3 Policy Control Mechanisms Associated with UNI Signaling Protocols

RSVP defines a policy data object that can be used to map optional objects that may be used for policy control across the UNI. Use of the policy data object to carry the contract ID is defined in Section 12. This object may be used to carry other policy-related data in the future.

Under LDP, a new Policy TLV is defined to carry policy-related data in signaling messages. This is described in Section 11.

13.4 UNI Security Considerations

Because optical connections carry high volumes of data and consume significant network resources, security mechanisms are required to safeguard a transport network against attacks on the control plane or unauthorized use of network resources.

Communication protocols usually require two main security mechanisms: *authentication* and *confidentiality*. Authentication mechanisms ensure data *origin verification* and *message integrity* of UNI messages so that unauthorized UNI operations can be detected and discarded. For example, UNI message authentication service can prevent a malicious UNI-C agent from mounting a denial of service attack against a service provider by inserting an excessive number of connection creation requests. Additionally, authentication mechanisms can provide (1) *replay protection*, which detects and rejects attempts to reorder, duplicate, truncate, or otherwise tamper with the proper sequence of messages, and (2) *non-repudiation*, which may be desirable for accounting and billing purposes. Authentication and confidentiality can be achieved using either *symmetric* or *public-key cryptographic algorithms*. Simple message integrity and confidentiality are normally achieved using symmetric cryptographic algorithms. These algorithms typically require pair-wise shared secret keys and do not provide non-repudiation, but are typically less computationally intensive. Replay protection is normally achieved by adding sequence numbers to the messages or by relying on other protocol (e.g., TCP) to guarantee the proper sequencing of the message stream above the integrity service. Public-key or *asymmetric* cryptographic algorithms are typically used, initially, to provide *two-way peer entity authentication* and *key agreement*, which facilitate use of the integrity and confidentiality services described above. Asymmetric algorithms also provide *digital signatures* used to implement a non-repudiation service. The use of asymmetric algorithms may be supported by a public-key infrastructure (PKI) or some other, community-defined, key assignment scheme. Asymmetric algorithms are typically more computationally intensive than symmetric algorithms. *It is expected that from the point of view of UNI 1.0 requirements, the most important security feature could be message authentication.* Confidentiality of UNI messages is also likely to be desirable, especially in cases where UNI message attributes include information private to the communicating parties (client and transport network operator). Examples of such attributes include account numbers, contract identification numbers, etc.

The case of non-co-located equipment presents increases security and policy control requirements. In this scenario, it is assumed that the UNI-C and UNI-N nodes are connected via networking devices such as layer 2 switches and IP routers. Since these devices could belong to different network operators and might be outside the control of the service provider, control communication between the UNI-C and UNI-N is subject to increased security threats, such as IP address spoofing, eavesdropping and unauthorized intrusion attempts. To counter these threats , appropriate security mechanisms have to be employed to protect the UNI signaling and control channel(s).

13.5 Security Mechanisms Relevant to UNI 1.0

13.5.1 RSVP Security Mechanisms

RSVP cryptographic authentication [RFC2747] defines an INTEGRITY object that provides hop-by-hop message integrity and authentication of RSVP messages. The INTEGRITY object contains a message authentication code (MAC) , computed using a secret key, shared by the two parties, and a MAC algorithm, such as keyed HMAC-MD-5. The shared secret key is never sent over the network, as in clear-text password schemes. This provides protection against eavesdropping attacks based on capturing information through access to the physical medium or packet data path. Thus, this scheme provides protection against forgery or message modification. In each RSVP message, the INTEGRITY object is also tagged with a one-time-use sequence number, which allows the message receiver to identify playbacks and hence avoid replay attacks. This, however, does not detect message deletion. This mechanism also *does not provide any confidentiality*, because messages stay in the clear. In addition, since it is only based on symmetric cryptography, it cannot provide non-repudiation. Standard two-way peer authentication and key management procedures have to be performed to complement this scheme. Security service negotiation is not provided, because negotiation would naturally be a part of the key management infrastructure. Manual key management can be used; in this case a privileged user manually configures the authentication keys.

IKE [RFC2409, RFC2407] or Kerberos could also be used for this purpose; Use of the latter is described in [RFC2747].

Keyed MD5 and HMAC-MD5 are already being used for cryptographic authentication of IP routing protocols, such as OSPF, IS-IS and BGP. An advantage of using this mechanism for RSVP is that the widespread prior use of the same or similar schemes will facilitate the deployment of this mechanism. Note, however, that the use of the 128-bit HMAC-MD5 in [RFC2747] differs from the 96-bit version in IPsec [RFC2403].

13.5.2 LDP Security Mechanisms

The LDP specification [RFC3036] defines a mechanism to protect the integrity of LDP messages. This mechanism is based on the use of the TCP MD5 signature option [RFC2385], which was defined primarily for BGP authentication and protects against the introduction of spoofed TCP segments into LDP connection streams. From a security services standpoint, the TCP MD5 signature option provides very similar properties to those provided by the RSVP authentication mechanism.

However, this mechanism has several shortcomings:

1. It provides no extensibility to new or user-defined algorithms, mechanisms, or services.
2. The use of MD5 in [RFC2385] does not conform to the commonly used HMAC construction (see Section 13.4.2).
3. As with RSVP, peer entity authentication and key management are unspecified.
4. There is no automated way to specify and enforce a policy as to whether the security option must be present.

13.5.3 Neighbor Discovery Security Mechanisms

Neighbor discovery is considered to be an elementary function that supports higher layer protocols. As such, security is not considered as part of neighbor discovery. Any security association is expected to be established *after* neighbor discovery is completed. If there is a concern about security in executing automatic neighbor discovery then it is suggested that this procedure be disabled and manual configuration of neighbor information be performed.

13.6 IP Security Protocols (IPSEC)

IPsec defines a suite of protocols for providing various security services at the IP layer, for both IPv4 and IPv6. These include two traffic protection protocols, the *Authentication Header* (*AH*) [RFC2402] and the *Encapsulating Security Payload (ESP)* [RFC2406] and one key management protocol, the *Internet Key Exchange (IKE)* [RFC2409, RFC2407]. The services offered by IPsec include access control, connectionless integrity, data origin authentication, replay protection and confidentiality (encryption). An important characteristic is that these services are provided at the IP layer, *offering protection for IP and upper layer protocols*. As such, IPSEC can be applied to both RSVP-TE and LDP signaling realizations of the UNI, as well as to protect other protocols referenced in this document such as LMP.

AH provides connectionless integrity, data origin authentication, and optionally, a replay protection service, whereas ESP provides confidentiality in addition to all the properties offered by AH. The mechanisms are designed to be algorithm independent and IPsec accommodates an extensible range of cryptographic algorithms. IKE is an elaborate two-way peer entity authentication, key management and security services negotiation protocol, which includes multiple algorithms and modes of authentication.

It is recognized in this specification that using the IPsec suite of protocols to protect UNI control messages offers a number of advantages. In particular, IPsec allows a single security solution for both RSVP-TE and LDP UNI signaling realizations, as well as neighbor discovery based on LMP, provides confidentiality, which is offered by neither of the signaling transport mechanisms and, additionally includes a key management protocol.

However, it is also recognized that IPsec is a relatively complex and heavyweight suite of protocols and, since AH and ESP are implemented at the IP layer, IPsec typically requires kernel modifications, potentially making implementations harder. Furthermore, it is not clear how many potential UNI clients will support IPsec in the immediate future.

13.7 UNI 1.0 Security Roadmap

UNI 1.0 uses the cryptographic authentication options of the underlying signaling transport mechanisms (RSVP-TE, LDP). This specification is made in order to accelerate deployment and interoperability for UNI 1.0, given the widespread deployment of and experience with similar cryptographic schemes. These mechanisms provide data origin authentication and message integrity and hence offer protection against denial of service attacks.

Both RSVP and LDP security mechanisms should use the HMAC-MD5 [RFC2104] *algorithm with 128 bits instead of the original MD5 algorithm specified in* [RFC2385]. This specification is made in order to achieve commonality across the different mechanisms as well as strengthen security properties. It should be noted, for example, that HMAC-MD5 offers improved security properties when compared to the original MD5 algorithm while requiring minimal additional computational complexity. The HMAC construction also allows the use of other similar hash functions in the future (e.g., SHA-1, RIPEMD-160, and SHA-256).
The standardization of more flexible and inclusive security options, such as IPsec , as well as the profiling of such mechanisms to make their implementation simpler, more efficient and more straightforward, may be considered in future versions of the optical UNI.

14 References

[CR-LDP] Jamoussi, B. Ed, "Constraint-Based LSP Setup Using LDP," (IETF Work in Progress), OIF2001.460, September, 2001.

[GMPLS SIG] P. Ashwood-Smith, et. al, "Generalized MPLS - Signaling Functional Description," (IETF Work in Progress), OIF2001.464, September, 2001.

[GMPLS CRLDP] P. Ashwood-Smith, et. al, "Generalized MPLS - CR-LDP Signaling Functional Description," (IETF Work in Progress), OIF2001.465, September, 2001.

[GMPLS RSVP-TE] P. Ashwood-Smith, et. al, "Generalized MPLS - RSVP-TE Signaling Functional Description," (IETF Work in Progress), OIF2001.466, September, 2001.

[GMPLS SONET] E. Mannie, et. al, "GMPLS Extensions for SONET and SDH Control," (IETF Work in Progress), OIF2001.438, August, 2001.

[LMP] J. P. Lang, et al., "Link Management Protocol," (IETF Work in Progress), OIF2001.461, September, 2001.

[OIF2000.155] J. Strand, et. al., "Carrier Optical Services Framework and Associated Requirements for UNI," OIF2000.155, September, 2000.

[RFC1662] W. Simpson, Ed., "PPP in HDLC-Like Framing", IETF RFC 1662.

[RFC1701] S. Henks, et. al, " Generic Routing Encapsulation (GRE)", IETF RFC 1701.

[RFC1853] W. Simpson, et. al, "IP in IP tunnel", IETF RFC 1853.

[RFC2104] H. Krawczyk, M. Bellare and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," IETF RFC 2104.

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

[RFC2205] R. Braden, Ed., "Resource Reservation Protocol (RSVP) - Version 1 Functional Specification," IETF RFC 2205, September, 1997.

[RFC2209] RSVP - Message Processing Rules", IETF RFC 2209.

[RFC2385] A. Heffernan, "Protection of BGP Sessions via the TCP MD5 Signature Option," IETF RFC 2385.

[RFC2402] S. Kent and R. Atkinson, "IP Authentication Header," IETF RFC 2402.

[RFC2403] C. Madson and R. Glenn, "The Use of HMAC-MD5-96 within ESP and AH," IETF RFC 2403.

[RFC2406] S. Kent and R. Atkinson, "IP Encapsulating Security Payload (ESP)," IETF RFC 2406.

[RFC2407] D. Piper, "The Internet IP Security Domain of Interpretation for ISAKMP," IETF RFC 2407.

[RFC2409] D. Harkins and D. Carrel, "The Internet Key Exchange (IKE)", IETF RFC 2409.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs," IETF RFC 2434.

[RFC2615] A. Malis and W. Simpson, "PPP over SONET/SDH," IETF RFC 2615.

[RFC2747] F. Baker et al. "RSVP Cryptographic Authentication," IETF RFC 2747.

[RFC2748] D. Durham, et al., "The COPS (Common Open Policy Service) Protocol," IETF RFC2748.

[RFC2750] S. Herzog, et al., "RSVP Extensions for Policy Control", IETF RFC 2750.

[RFC2752] S. Yadav, "Identity Representation for RSVP," IETF RFC 2752, January 2000.

[RFC2961] L. Berger, et al., "RSVP Refresh Overhead Reduction Extensions," IETF RFC 2961.

[RFC3036] L. Andersson, et. al., "LDP Specifications," IETF RFC 3036.

[RSVP-TE] D. Awduche, et. al, "Extensions to RSVP for LSP Tunnels," (IETF Work in Progress), OIF2001.462, September, 2001.

Appendix A: ITU Terminology and Functional Models

A.1 Terminology

Client/Server Generic terms used to differentiate the initiator in a relationship (the client) from the responder and service provider (the server). The term "client" and "server" are multi-used and so should not be used in isolation, without qualification or context. In earlier sections, the transport network is assumed to be the server and user networks (e.g., IP) are assumed to be the clients.

Client-Layer A layer acting as a client with regard to transport services provided by a server layer.

Client-Layer Address An address used in client-layer protocols.

Connection Termination Point The client side of an adaptation function. Refer to M.3100 for complete definition of connection termination point.

Layer Network For SONET Layers refer to ANSI T1.105
For SDH Layers refer to ITU-T G.803
For OTN Layers refer to ITU-T G.872

Layer-Network A set of potentially connectable Access-Points (AP) all of the same Characteristic-Information (CI). A layer network can be either or both a client-layer-network and/or a server-layer-network. (It can be isolated, or it can be at the top or bottom of a stack, or it can be in the middle of a stack of layer-networks.) Refer to G.805 for complete definition of layer-network.

Link Represents the available transport capacity between two sub-networks. Refer to ITU G.805 for complete definition of link.

Link Connection An entity that represents the smallest granularity capacity that can be allocated on a link. Refer to ITU G.805 for complete definition of link connection.

Network Connection Entity that transports information transparently across a particular network layer without any check on the quality of the transport. Refer to G.805 for complete definition of network connection. This has been referred to as "Optical Connection" or simply, "Connection", in earlier sections.

Server Layer The server layer provides transparent transport for the client layer. Refer to ITU G.805 for complete definition of server layer.

Sub-network In the context of the connection domain, an entity that represents flexible connectivity; there is no notion of distance being traversed. Refer to ITU G.805 for complete definition of sub-network.

Sub-network Connection A transport entity that transfers information across a sub-network. It is formed by the flexible association of ports on the boundary of the sub-network. Refer to ITU G.805 for complete definition of sub-network connection.

Sub-network Points A pair of addressable connection points in the client(s) network between which a link connection is established using UNI signaling.

Trail An end-to-end connection across a particular network layer, and the entity required to provide an automatic means to check the quality of transport Refer to G.805 for complete definition of trail.

Trail Termination Point End points of a network connection (which coincides with the ends of a trail in the information model). Refer to M.3100 for complete definition of trail termination point.

A.2 Functional Model Terminology

The functional model, illustrated in Figure A-1 represents all network functions by “atomic” or “elementary” functions: connection, termination and adaptation. Note that the link-connection, trail and network connection are defined above, and shown in bold in Figure A-1.

.

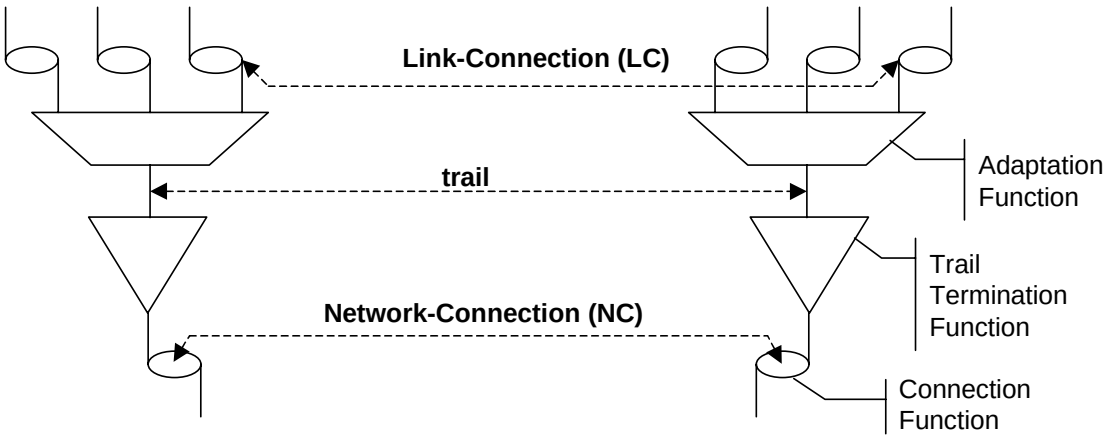


Figure A-1 - Functional Model

The adaptation function changes the information from the client layer to a form suitable for the server layer and vice-versa. The trail termination function provides the capability to monitor the integrity of the signal, typically by adding, deleting and using overhead. The connection function provides the means to communicate the information (or items of information) between groups of atomic functions within the same layer.

A.3 Informational Model Terminology

The informational model, illustrated in Figure A-2 presents data specified in atomic functions to a management system. Note that the link-connection and network connection are common with the functional model and are defined above. In addition, the sub-network, sub-network-connection, CTP and TTP are defined above and shown in bold in Figure A-2.

The information model follows from the functional model. They are, however, not identical because they have different emphasis. The functional model is concerned with the transformation of payload signals, while the informational model is concerned with the connectivity of signals.

The mapping of the functional model to the informational model is illustrated in Figure A-3.

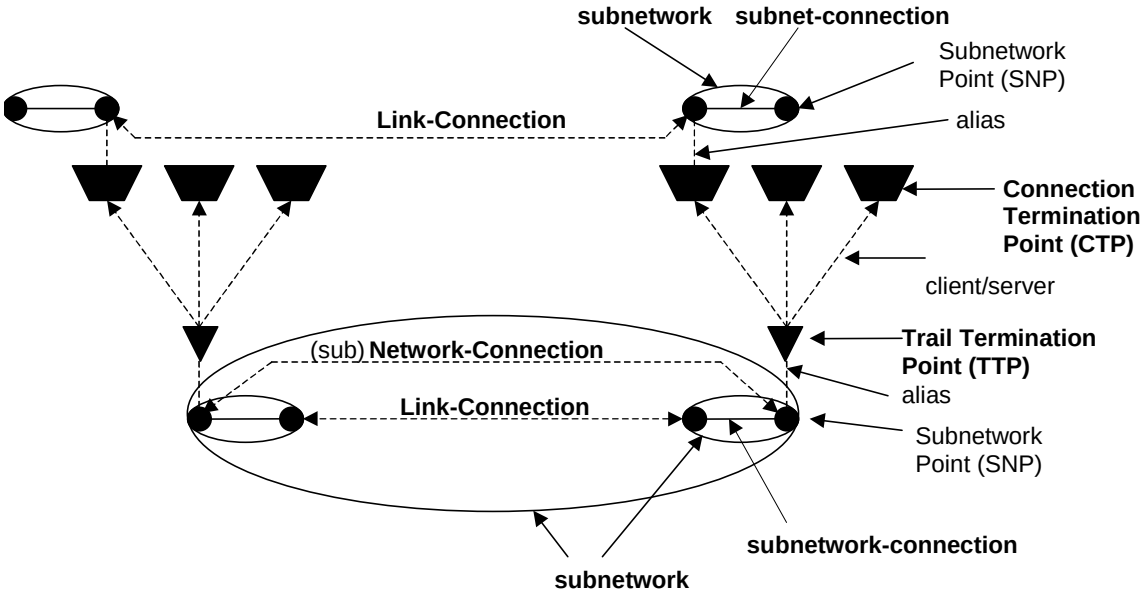


Figure A-2 – Informational Model

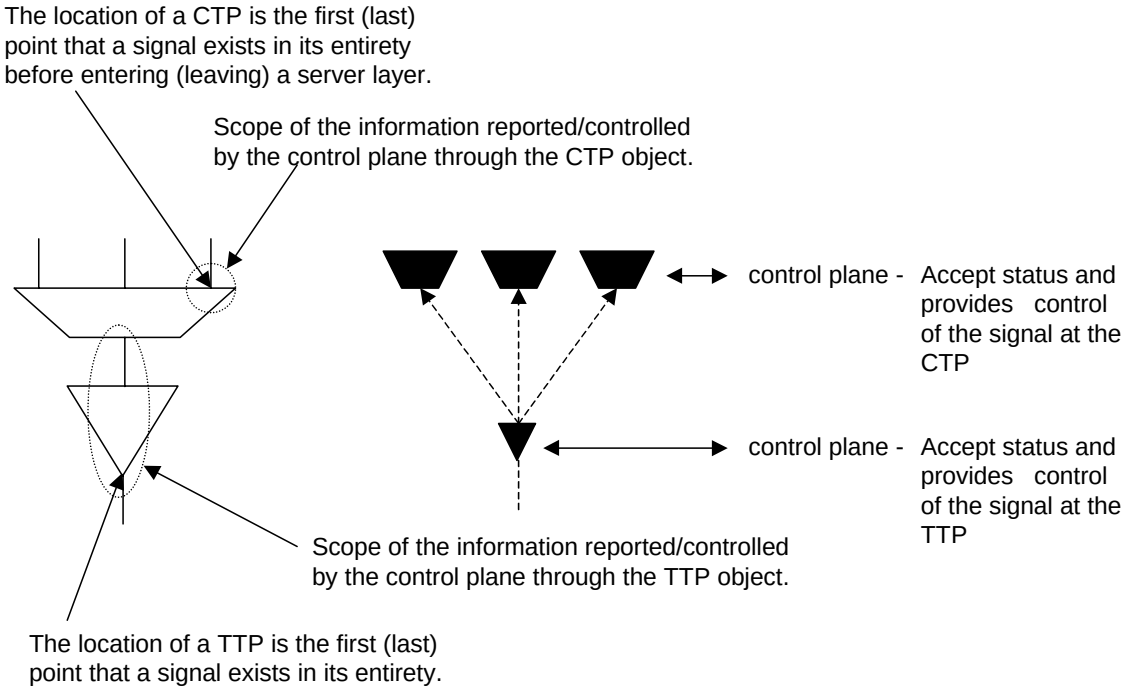


Figure A-3 - Mapping, Functional Model-to-Informational Model