Contribution Number: OIF2001.152

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

TITLE: Interim User Network Interface (UNI) Signaling Implementation Agreement for SuperComm 2001

DATE: April 18, 2001

Document Status: Draft
Project Name: Signaling
Project Number:

Notice: This draft implementation agreement document has been created by the Optical Networking Forum (OIF). This document is offered to the OIF Membership solely as a basis for agreement and is not a binding proposal on the companies listed as resources above. The OIF reserves the rights to at any time to add, amend, or withdraw statements contained herein. Nothing in this document is in any way binding on the OIF or any of its members.

The user's attention is called to the possibility that implementation of the OIF implementation agreement contained herein may require the use of inventions covered by the patent rights held by third parties. By publication of this OIF implementation agreement, the OIF makes no representation or warranty whatsoever, whether expressed or implied, that implementation of the specification will not infringe any third party rights, nor does the OIF make any representation or warranty whatsoever, whether expressed or implied, with respect to any claim that has been or may be asserted by any third party, the validity of any patent rights related to any such claim, or the extent to which a license to use any such rights may or may not be available or the terms hereof.

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

List of Contributors

Osama Abul-Magd, Nortel

Olga Aparicio, Cable and Wireless

K. Arvind, Tenor Networks

Greg Berstnein, Ciena

Yang Cao, Sycamore Networks

Amy Copley, Sycamore Networks

Hans-Martin Foisel, Deutsche Telekom

Raj Jain, Nayna Networks

Jim Jones, Alcatel

Jonathan Lang, Calient Networks

Fong Liaw, Zaffire

Zhi-Wei Lin, Lucent

Swee Loke, Avici

Eric Mannie, Ebone

Gerald Neufeld, RedBack Networks

Wilson Nheu, Agilent

Dimitri Papadimitriou, Alcatel

Dimitrios Pendarakis, Tellium (editor)

Bala Rajagopalan, Tellium

Robert Rennison, Laurel Networks

George Swallow, Cisco Systems

Russ Tuck, Pluris

Jim West, Ciena

Fritz-Joachim Westphal, T-Nova – Deutsche Telekom

Richard Whitebrook, Agilent

Cary Wright, Agilent

Yangguang Xu, Lucent

Jennifer Yates, AT&T

Lucy Yong, Williams Communications Group

John Z. Yu, Zaffire

1 Introduction

The UNI 1.0 interoperability agreement as defined by the OIF is quickly approaching a mature state. Interoperability testing is required to fine tune the UNI 1.0 document, agree on different interpretations, and accelerate implementation and industry acceptance.

This document defines a minimal subset of UNI functionality, named the "SuperComm Interim UNI", and a plan for interoperability testing the SuperComm Interim UNI. The test plan includes a mandatory minimum level of functionality and additional optional functionality to be implemented and tested. This document replaces OIF2000.125.3 for the purpose of demonstrating the interim UNI at Supercomm 2001.

This document is the result of collaboration between multiple vendor companies participating in the OIF interoperability event. Comments and additional material are solicited from all OIF members interested in this event. The initial version of this document was generated by Fong Liaw (Zaffire), Wilson Nheu (Agilent), Cary Wright (Agilent) and Dimitrios Pendarakis (Tellium).

1.1 Mandatory UNI Components

The SuperComm Interim UNI consists of the following functional components.

- IP Control Channel Configuration. For the SuperComm Interim UNI the *mandatory* realization of the IP Control Channel (IPCC) is out-of-band, out-of-fiber over an Ethernet. Other configurations (in-band, in-fiber or out-of-band, in-fiber [1]) are optional. The mandatory IPCC configuration is described in Section 2.1.
- Manually Configured Neighbor Discovery Information. Neighbor Discovery allows two connected network elements (such as a UNI-C and a UNI-N device) to discover port connectivity information. For the SuperComm Interim UNI this information will be manually configured. Automated neighbor discovery [1] is optional for the SuperComm demonstration.
- UNI Signaling. The mandatory signaling protocol for the SuperComm Interim UNI is based on the RSVP-TE protocol. In particular, this document specifies the implementation of only a subset of the RSVP messages and objects described in OIF2000.125.3. Section 6 of this document lists the messages and objects that are considered mandatory for the SuperComm Interim UNI. In general, the document covers the essential messages and objects that allow basic UNI functionality and includes their definitions and usage examples. *Support of the LDP-based UNI signaling protocol is optional*.

Due to timing constraints, initial versions of this document will focus on the mandatory components of the SuperComm Interim UNI. Optional components will be described in future versions, depending on timing consideration and/or vendor demand.

1.2 Testing Overview

Testing will be held in two stages. A closed door interoperability event for initial detailed testing and debugging, followed by a public demonstration to be held at SuperComm.

1.3 Closed Door Event Participation Pre-Requisites

The intention of the closed door event, is to allow developers to debug and test implementations without fear of negative publicity. Attendees will be bound by confidential disclosure, and individual test results will not be published.

Companies wishing to participate in the closed door event must fulfill the following criteria:

1. Be a Principal Member of the OIF.

2. A working implementation of SuperComm Interim UNI client or network side integrated into a product that will be commercially available to the market, or UNI software stack that will be commercially available to the market, or test equipment designed for testing UNI protocols.
3. Acceptance of the Confidential Disclosure Agreement terms and conditions.
4. Payment of a nominal fee.

It is expected that OIF carriers will be invited to participate during the last day(s) of the closed door event. The exact length and scope of carrier participation is currently being discussed.

1.4 Public Demonstration Event Participation Pre-Requisites

The public demonstration event is intended to demonstrate maturity of the SuperComm Interim UNI agreement by displaying working implementations at a major trade show. The format will attempt to replicate the closed door test environment.

To participate at the public demonstration, companies must fulfill the following criteria:

1. Successful participation at the closed door event.
2. Payment of a nominal fee.
3. Agree to the terms and conditions of trade show participation.

1.5 Timeline

Closed Door Event:	May 4	Equipment installation
	May 7 to May 11	Testing Days
Observer Day:	May 14	
Tear down:	May 15	
Public Demonstration:	June 3~7	

2 Test Reference Configuration

The actual network test topology will be defined before the private interoperability event and as soon as vendor participation details (including type of equipment, equipment role, size, etc.) are finalized. Participants are expected to support multiple interface configurations. The following is an example of a multiple interface topology:

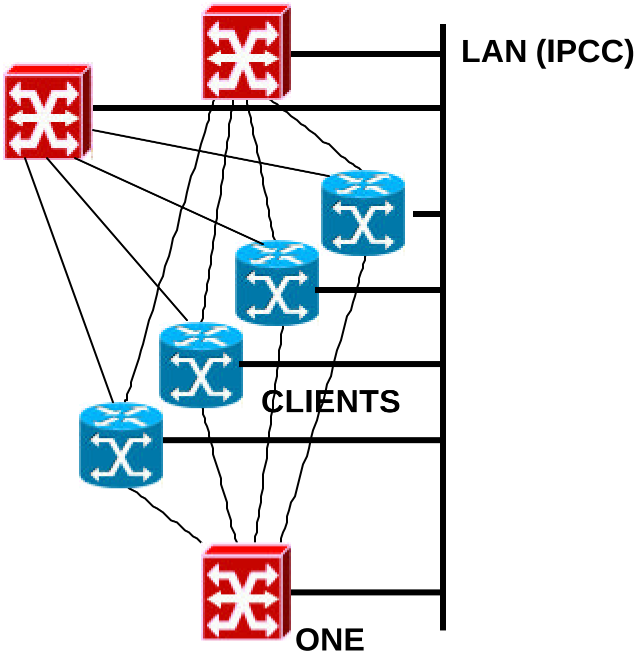


Figure 1: Example test topology for the SuperComm Demonstration

It is expected that UNI-C devices will be connected to multiple optical network elements. In addition, it is expected that several network elements may play the role of both UNI-C and UNI-N.

An interconnect schedule will be developed for use at the private event. While this schedule will not cover all combinations of ONE and Client devices, it is expected that the schedule will create significant diversity.
One of the combinations tested above will be selected for demonstration at the public event, this combination will be pre-staged at the end of the private interoperability event. Vendors will have the option of suggesting preferred partners for the public demonstration, however they must be willing to accept additional partners to ensure that all vendors can publicly demonstrate interoperability.

2.1 Mandatory IP Control Channel (IPCC)

The mandatory IP control channel (IPCC) transport mechanism for the demonstration is out-of-band, out-of-fiber. Specifically, both UNI-C agents and the UNI-N agent involved in a single round of experiments will be connected by a 10BaseT or 100BaseT Ethernet network.

The signaling messages exchanged over this network are RSVP messages encapsulated in IP over Ethernet frames. PPP framing should not be used for the IP control channel. Similarly IP-in-IP encapsulation should not be used for RSVP messages. This assumes that the whole demonstration network is a single bridged network. Clients and ONE's shall be directly connected to a single IP subnet via a switched Ethernet LAN, no routers shall be in the path of signaling messages.

2.2 Transport Interface Specifications

For interoperability testing, the transport interface for both Client and Network devices shall support either SONET OC48c, OC48, SDH STM-16c, or STM-16, using 1310nm Single Mode Short Reach optics with a transmit power of -3dBm to -9.5dBm. Higher output power transmitters may be used in conjunction with attenuators to ensure optical power level at the receiver does not exceed -3dBm.

The recommended default interface is concatenated OC48c or STM16c, it is expected that most vendors will supply concatenated interfaces. However, vendors may supply equipment supporting non-concatenated OC48/STM16 interfaces. Vendors supplying non-concatenated interfaces must take the responsibility to ensure that there are mutually agreed UNI-N and UNI-C partners to interoperate with.

It is expected that most of the client equipment (UNI-C) will be path terminating. The default level of transparency expected by client equipment should be path level transparency. This is the minimum amount of transparency that can be requested from optical network elements (UNI-N) and thus should satisfy the requirements of all participating UNI-N equipment (whether they are Line or Section terminating). UNI-C elements requiring higher degrees of transparency should ensure the their UNI-N partner vendors can support the required level of transparency.

3 Addressing

UNI operation involves multiple address spaces, related to both signaling and transport, as shown in Figure 2~~Figure 2~~. We describe these address spaces and their relation to the interoperability event, next.

1. Internal optical network addresses. These are addresses used by optical network elements within the optical network for internal routing and provisioning purposes and are shown in red color in Figure

2~~Figure 2~~. These addresses are not used in UNI signaling messages, are not exported to users and, thus, are **outside the scope of this specification**.

2. UNI-N and UNI-C IP Control Channel (IPCC) addresses. These are unique IPv4 addresses used as the source and destination of UNI signaling messages. They are shown inside green boxes in Figure 2~~Figure 2~~. For the SuperComm Interim UNI demo **IPCC addresses will be routable IPv4 addresses assigned at the interoperability event**. Given the expected topological complexity of the interoperability network, all IPCC addresses SHALL belong to the same IP sub-network.
3. UNI connection endpoints are identified by ONA addresses, shown in black in Figure 2~~Figure 2~~ [10]. Each ONA is an IPv4 address assigned to one or more physical transport links (such as OC-48 links) connecting a UNI-N and a UNI-C. For the SuperComm demonstration ONA addresses are *unique* IPv4 addresses assigned prior to the interoperability event.
4. **Client addresses**, denoted by C_A, C_B and C_C in Figure 2~~Figure 2~~ and shown in blue color. These follow the addressing scheme of different client types (IP routers, ATM switches, SONET ADM) and are not exchanged in signaling messages. They are therefore not considered in this specification. Client addresses may be used for connection verification purposes. For example they might be used as the destination addresses in ping messages between IP router UNI-Cs.

Figure 3 ~~**Figure 3**~~ shows an example reference configuration and the relevant addressing conventions.

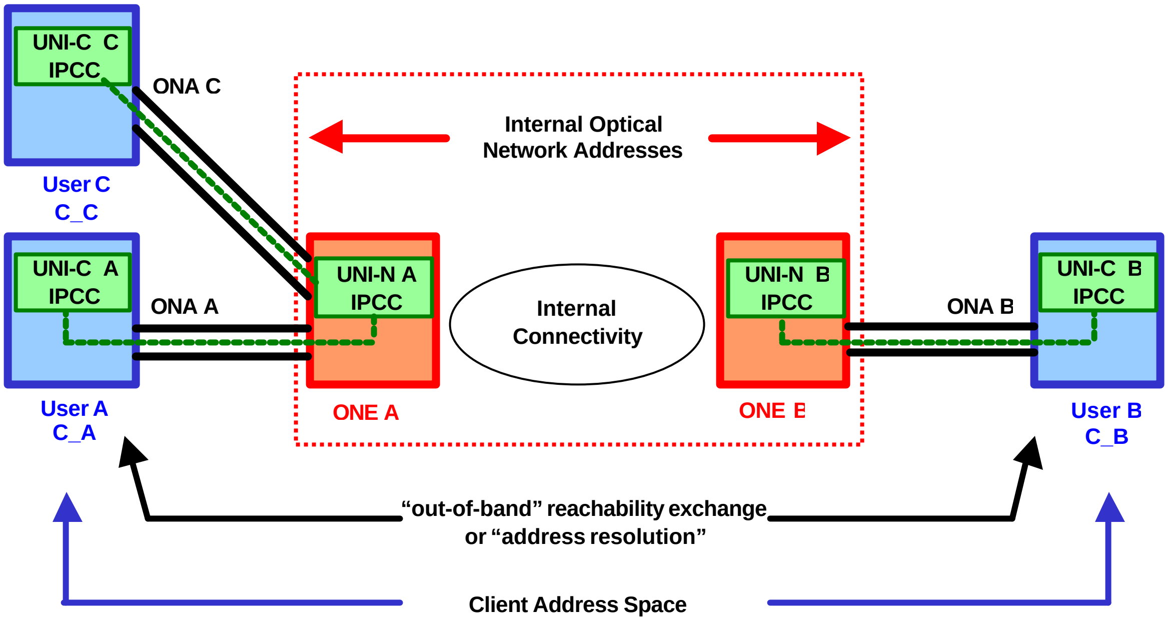


Figure 2: Address Spaces Relevant to UNI Operation

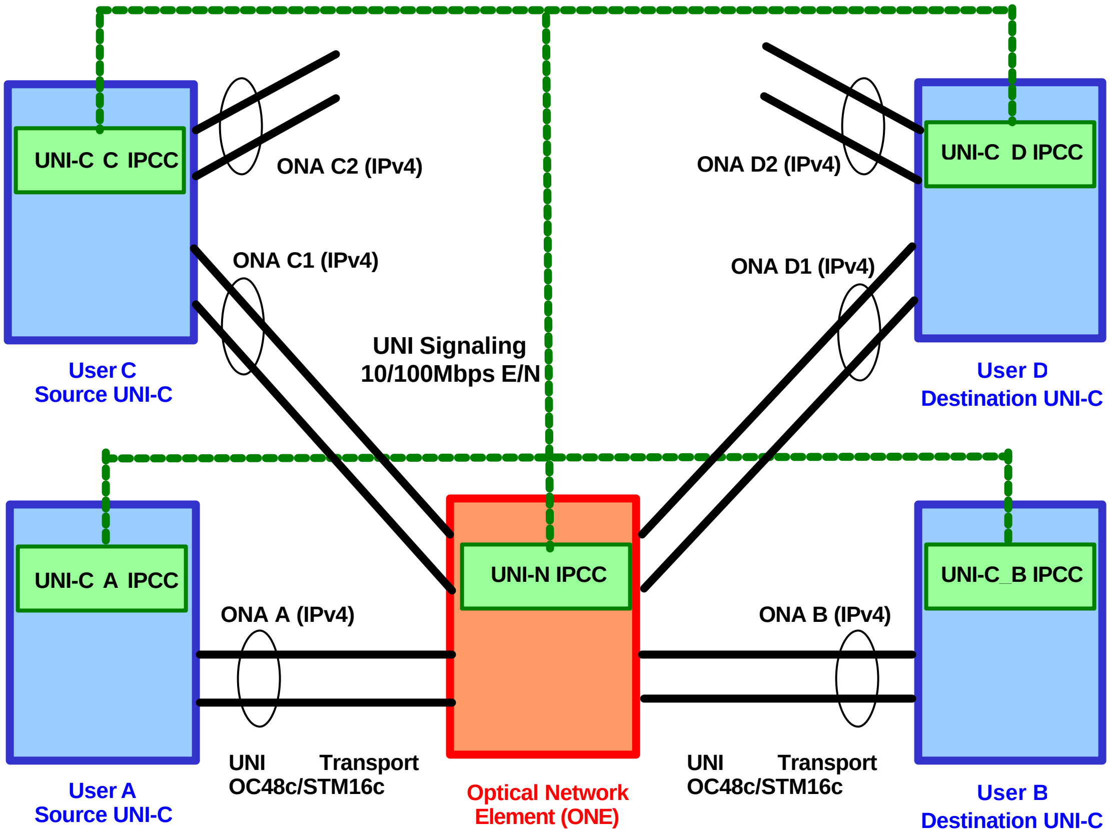


Figure 3: UNI Interoperability Event Addressing Conventions

The remaining of the document assumes the above reference configuration in its use of address assignment, i.e. UNI-C A IPCC, UNI-C B IPCC, ONA A and ONA B.

4 Configured Neighbor Discovery Table

As described in Section 3, a single ONA corresponds to one or more physical links. In the case where a single ONA corresponds to more than one physical links, individual links sharing the same ONA are identified using a **port identifier**. A port identifier is used to identify for signaling purposes a physical port within a network element (UNI-C or UNI-N). Assignment of port identifiers is a *local decision of each network element*; thus each link connecting a UNI-C and a UNI-N will be associated with two port identifiers, one on the UNI-C and one on the UNI-N. UNI signaling messages may carry a port identifier in order to identify individual physical links sharing the same ONA. **For the purpose of the SuperComm Interim UNI a port identifier is encoded as a 32-bit integer and must be unique a network element**. *It should be noted that port identifiers may be different from the internal port numbering scheme used within a network element , so that the internal optical network port numbering scheme is not exposed to network users*

Correct operation of UNI signaling relies on *unambiguous mapping between UNI-C and UNI-N port identifiers*, as these are carried in connection establishment and tear-down requests. This mapping can be established either automatically, using the automated neighbor discovery procedure [1], or manually, by configuration of both UNI-C and UNI-N. Since automated neighbor discovery is optional for the

SuperComm Interim UNI, by default port identifier values will be manually configured in a table, which will be termed *configured neighbor discovery table* in the rest of this document.

Figure 4~~Figure 4~~ shows two interconnected UNI-C and UNI-N devices and their ports identifiers, CP_* and NP_*, respectively. The configured neighbor discovery table in each device will contain the mapping between local and remote port identifiers.

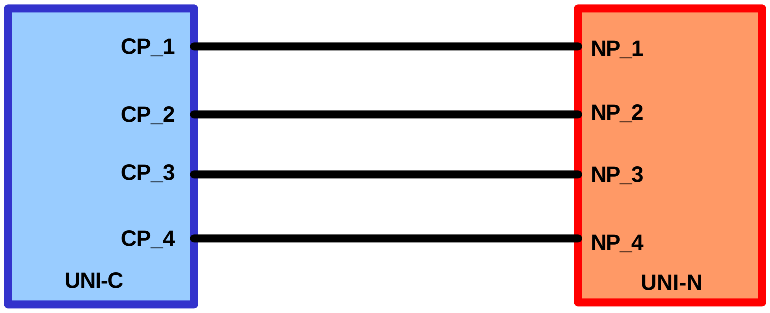


Figure 4: UNI-C and UNI-N port mapping

5 Signaling Test Scenarios

The following is a set of test scenarios to be shown at the interoperability event. The diagrams depict the source and destination UNI-C agents, as well as an ingress and egress UNI-N ports. Since the interoperability demonstration is based on a single optical network element (ONE), the ingress and egress UNI-N agent are be collapsed into a single agent. In the interest of generality, in the following diagrams we still show the ingress and egress UNI-N ports separately. All test scenarios are assumed to also support proxy signaling agents signaling on behalf of clients or the ONE.

As part of the mandatory tests, connections should be maintained for at least 3 (RSVP) refresh time intervals before a tear-down is initiated. The default refresh timer is 30 sec; use of infinite refresh timers is optional. *All connections are assumed bi-directional by default.*

For the interoperability event, *mandatory* test scenarios are:

- Successful connection establishment.
- Connection tear-down initiated by source UNI-C.
- Connection tear-down initiated by the destination UNI-C.

Other test scenarios may optionally be demonstrated. In particular, this document defines the following optional scenarios:

- Connection tear-down initiated by the Network
- Connections setup rejected by the Network due to unknown ONA address.
- Connection setup rejected by destination UNI-C. This could be due to the destination UNI-C not accepting connections from the UNI-C originating the request, for example source UNI-C may belong to different administrative domain.

A set of timing diagrams showing the message flow for these scenarios are shown below. Timing diagrams 5.1, 5.2, 5.3 represent mandatory test scenarios, all others are optional.

5.1 Successful connection establishment

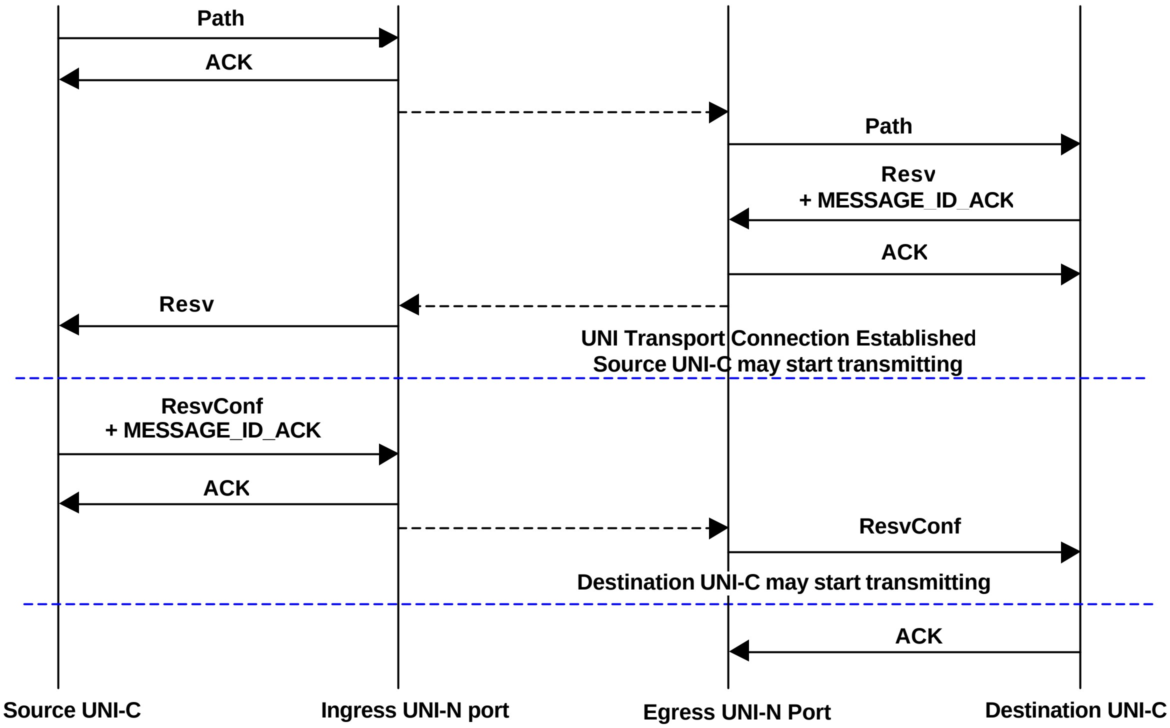


Figure 5: Successful Connection Establishment

The blue dotted lines in Figure 5~~Figure 5~~ are relevant to the establishment of the transport layer connection. It is assumed that the optical network element (UNI-N) has established the optical connection across the ONE before sending the Resv message to the source UNI-C. Therefore, the source UNI-C *must not* start data transmission before the Resv message is received. The destination *should not* start transmission before the ResvConf message is received. *The blue dotted lines are provided for informational purposes; it is not expected that adherence to this recommendation for data transmission purposes will be tested at the interoperability event.*

5.1.1 UNI-C and UNI-N Port Identifier Selection

UNI signaling messages carry a port identifier which is selected by upstream devices, consistent with the RSVP specification [RFC2205].

Figure 6~~Figure 6~~ shows an example of source UNI-C, UNI-N and destination UNI-C devices involved in a UNI connection set-up. The respective port identifiers used in signaling messages are denoted by SCP_*, NP_* and DCP_*, as shown in the figure. As mentioned before, these indexes may be different from the actual internal port numbering scheme used within the network elements.

In reference to Figure 6~~Figure 6~~, a UNI RSVP message from the source UNI-C to the UNI-N will carry a port identifier of the source UNI-C, for example **SCP_2**. Using the configured neighbor discovery table, the UNI-N translates this to its port identifier **NP_2**. Next, the UNI-N selects the outgoing port identifier, say **NP_7**, and includes it in the UNI RSVP signaling message sent to the destination UNI-C. Again, the downstream device (i.e. destination UNI-C) uses its configured neighbor discovery table to map the received port identifier to its own port identifier **DCP_3.**

This approach is consistent with the RSVP specification [RFC2205].

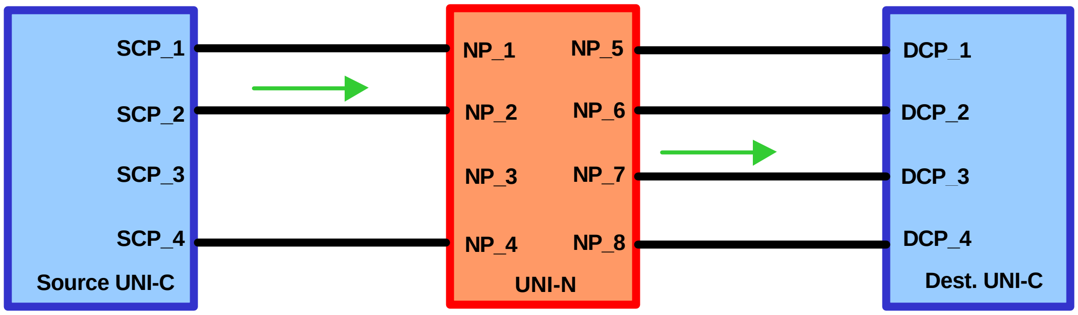


Figure 6: UNI-C and UNI-N Port Selection

5.2 Connection Tear-Down Initiated by Source UNI-C

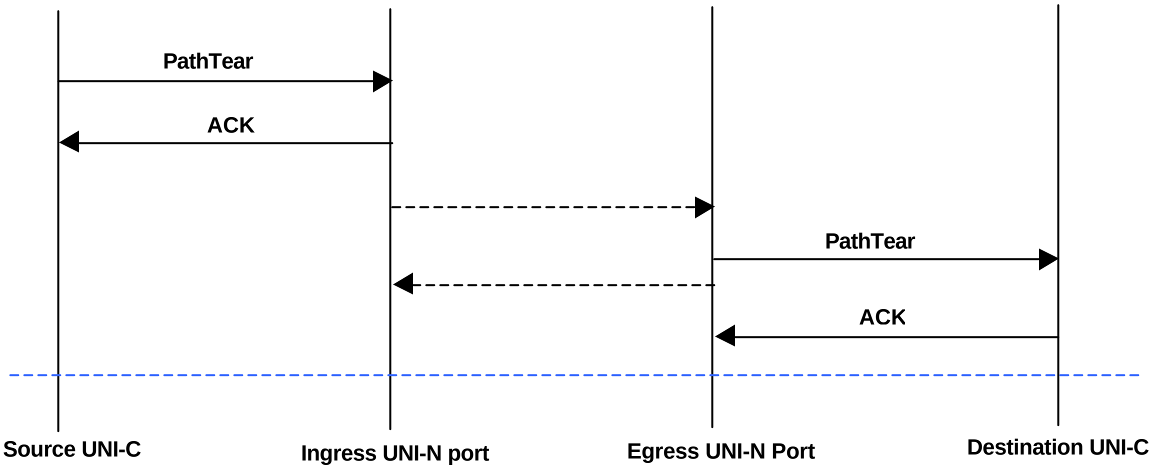


Figure 7: Connection Tear-Down Initiated by Source UNI-C

In this test scenario, care must be taken that the source UNI-C does not terminate the data channel before the PathTear message is received at the destination UNI-C, otherwise the destination UNI-C network element might generate various alarms. Solution to this problem might require modifications to the default RSVP/RSVP-TE operation which are currently being discussed.

The blue dotted line in Figure 7~~Figure 7~~ is meant to indicate when the source UNI-C can terminate the data channel transmission. The source UNI-C should allow for some reasonable time after the tear-down message is generated for the message to be transmitted by the network, received by the destination UNI-C and action be taken by the destination UNI-C. One potential solution to this problem, proposed in [11], introduces a "blockade state" which introduces a delay whenever path state is to be deleted. A high-level description of this operation is as follows:

```
if (path state is to be removed)
  {
      start blockade timer (configurable, default 30 sec?);
```

```
        enter blockade state;
    }

Upon blockade timer expiration
    {
        remove data plane connection;
        if (no unacknowledged messages)
              delete path state;
        else
           mark state as waiting to be deleted;

Upon receipt of Message_ID_ACK
        if (state is waiting-to-be-deleted)
              delete path state;
```

More details on this approach will be provided in a future contribution. It should be noted that this is not a standards based solution and it is expected that it will be discussed further in the OIF. It is provided here for informational purposes.

5.3 Connection Tear-Down Initiated by Destination UNI-C

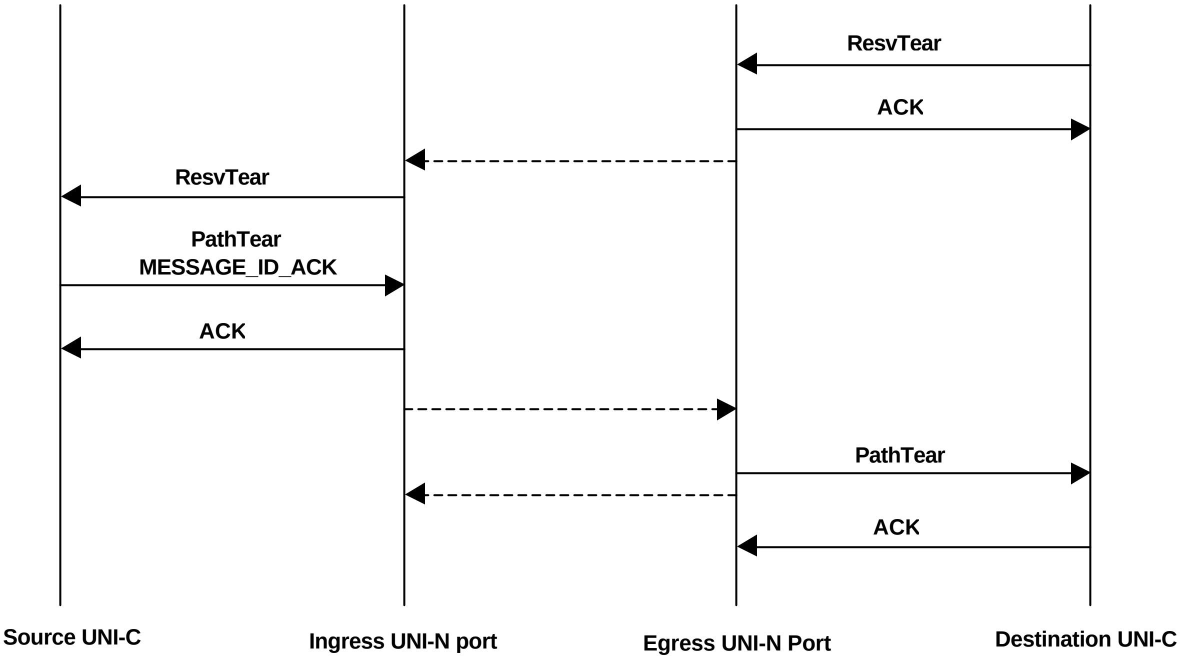


Figure 8: Connection Tear-Down Initiated by Destination UNI-C

Note that the upstream direction of the connection should not be removed when processing the ResvTear message. This is to avoid triggering unnecessary alarms at the source UNI-C. Ideally, the cross-connect should be removed when processing the PathTear message.

5.4 Connection Tear-Down Initiated by the Network

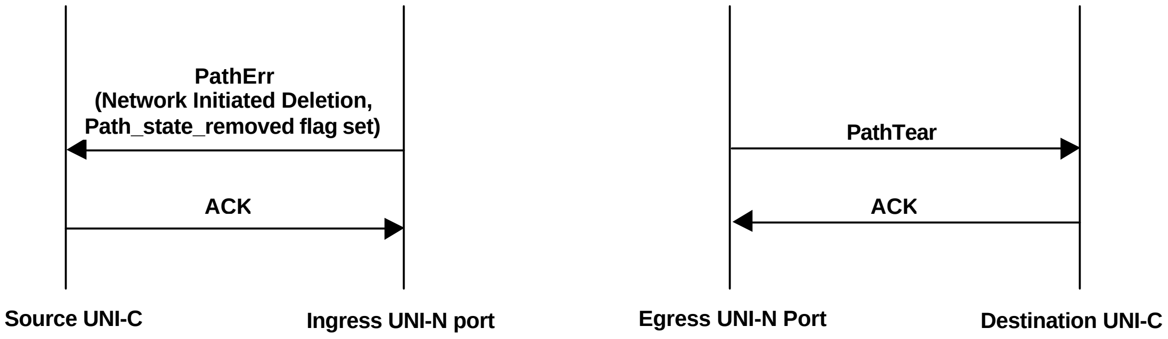


Figure 9: Connection Tear-Down Initiated by the network

In Figure 9~~Figure 9~~ the network initiates a connection deletion by sending a PathErr to the source UNI-C and, simultaneously, a PathTear to the destination UNI-C. The PathErr message send to the source UNI-C has the "Path_state_removed" flag set to indicate that since the network is initiating the tear-down the path state is deleted. Therefore, a PathTear from the source UNI-C is not required to terminate the connection; in fact, a PathTear would be discarded since Path state will have already been removed.

This reflects the fact that connection tear down can be initiated by either the UNI-N or UNI-C on a unilateral basis, without consent by the other party. While normally connection tear-down will be initiated by the UNI-C, UNI-N might also initiate tear-down in response to event such as:

- A user's (UNI-C) account is not in good standing.
- Internal optical network failures which force the network to terminate connections.

In either case, the network may act regardless of the client response. It only needs to inform UNI-Cs about the action taken.

5.5 Connection Rejection by the Network

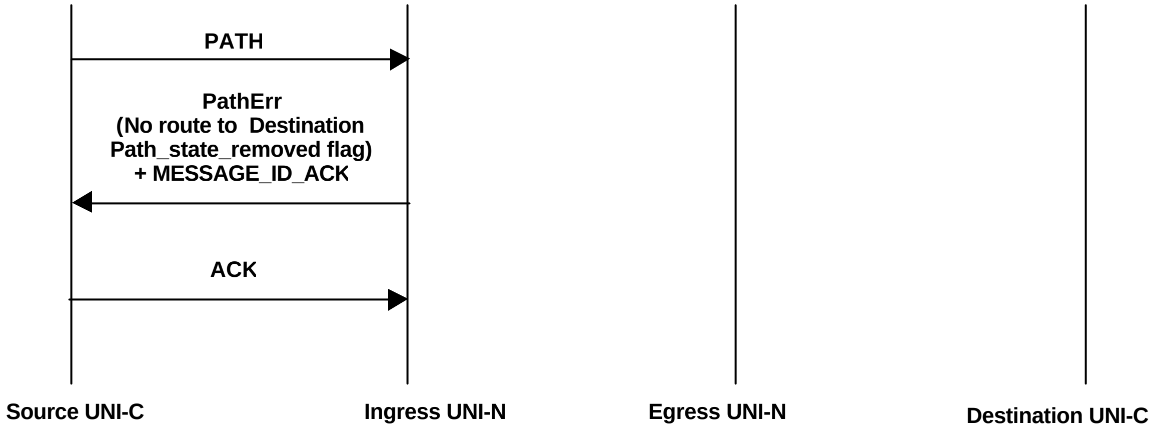


Figure 10: Connection rejection by the network due to bad destination ONA address

A UNI-N rejecting a connection request from a UNI-C SHOULD set the "Path_state_removed" flag.

5.6 Connection Set-Up Rejection by Destination UNI-C

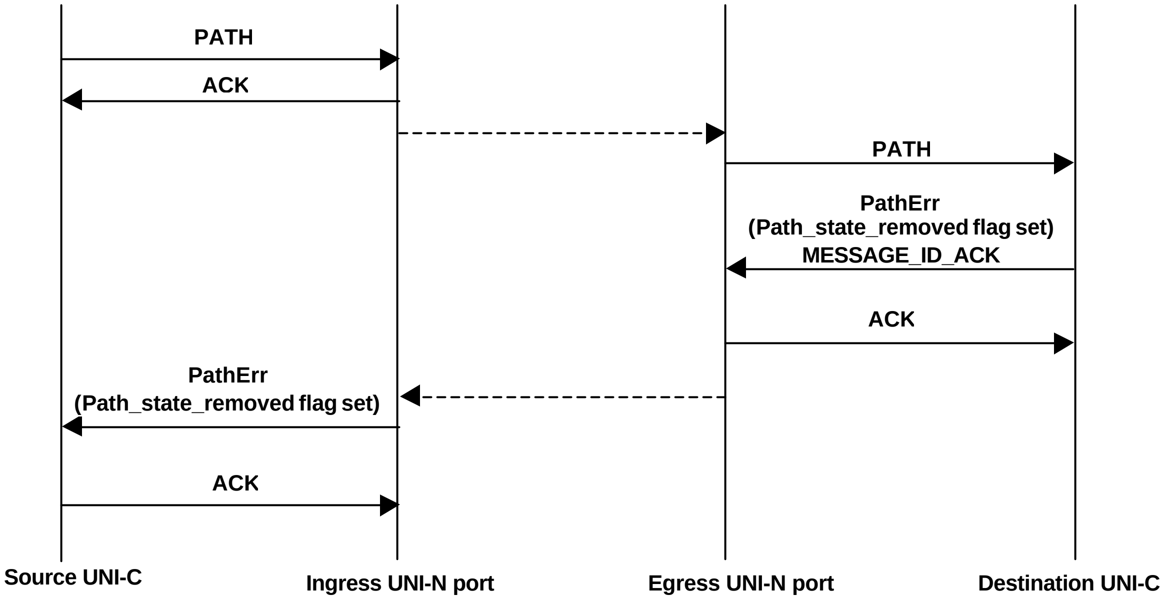


Figure 11: Connection set-up rejection by destination UNI-C

5.7 Error Handling

A number of error conditions can occur in the above timing diagrams. Detailed discussion of error handling will be included in a later version of this document, after all possible fault combinations are taken into account. In some cases, error handling can be considered an application layer policy issue for the respective network elements and is therefore left up to individual implementations to handle.

1. As an example, consider the following error conditions: A source UNI-C sends a Path message which is acknowledged by the UNI-N, but subsequently no Resv message is received. In this scenario it is possible that the optical network has reserved resources, but since no Resv is received the connection cannot be used.
2. A destination UNI-C has responded to a Path message with a Resv, but no ResvConf has been received. Again, while resources may be reserved by the optical network, the destination UNI-C should not start transmitting data.

In both cases, the error is due to the fact that the expected response to a message that was sent by a network element was not received. The way in which network elements deal with such errors can be considered a matter of local policy. As an example course of action for condition (1) above, the source UNI-C may start a configurable timer after the Path message is sent. When this timer expires, it would take corrective actions by sending a PathTear message. The same action could be taken by the UNI-N.

Such error handling conditions are not expected to be tested in the SuperComm demonstration.

6 RSVP Message Encoding for SuperComm Demonstration

The following message encoding is defined for the SuperComm Interim UNI demonstration. Vendors must encode messages using the values and encoding order defined below.

6.1 Message Header

An RSVP message transmitted out-of-fiber, out-of-band, over a directly connected 10/100 Ethernet MUST be encapsulated into an IP over Ethernet Frame as described below. It is expected that all RSVP messages will fit in a single Ethernet frame, so that no fragmentation is required. UDP encapsulation of RSVP messages MUST NOT be used.

```
+--------------------------------+
|       DA MAC                   | 6 bytes
+--------------------------------+
|       SA MAC                   | 6 bytes
+--------------------------------+
|       EtherType 0x08-00        | 2 bytes
+--------------------------------+
|              .                 |
|           IP PDU               |
|     (up to 1500 octets)        |
|              .                 |
+--------------------------------+
|             CRC                | 4 bytes
+--------------------------------+
```

6.1.1 IP Header

The format and field value of the IP header that carries an RSVP message for configuration 1 is shown below.

IP Header for UNI RSVP messages	
Version	4
IHL	5
Type of Service	0111b (Network control)
Total Length	message length
Identification	(as defined by RFC791)
Flags	(as defined by RFC791)
Fragment Offset	(as defined by RFC791)
Time to Live	>= 1
Protocol	46
Header Checksum	(as defined by RFC791)
Source Address	UNI-C/UNI-N IPCC (IPv4)
Destination Address	UNI-N/UNI-C IPCC (IPv4)

Note that, as shown in the above table router alert option SHOULD NOT be used for the interoperability event. The router alert is not needed in the Path, PathTear or ResvConf messages since these messages are addressed to the UNI-N IPCC address. Any value of the “Time to Live” (TTL) field larger or equal to 1 is acceptable. Note that a value of 1 would suffice for the interop event, since all network elements will be one-hope away from each other.

6.1.2 RSVP Header

The format and field of the RSVP header for the reference configuration is show below.

RSVP Header	
Version	1

```
            | Flags                      0 (Message_ID only)       |
            | Msg Type                   (as defined in RFC2205)   |
            | RSVP Checksum              (SHALL compute)           |
            | Send_TTL                   >= 1
    |
            | RSVP Length                (include RSVP header)     |
            +-------------------------------------------------------+
```

Note that the value of the flags field is 0 since a node that only supports reliable RSVP message delivery (and not Bundle and Srefresh messages) should not set this flag.

The RSVP checksum SHOULD be computed. An all zero value is used to indicate that no checksum was transmitted [RFC2205]. A transmitted value of 0 is acceptable for the SuperComm Interim UNI. Any value of the "Send TTL" field greater or equal than 1 is acceptable. The send TTL should match that of the IP header.

6.2 Message Types

For the interoperability event, the following message types are mandatory. Any messages not mentioned will be treated as optional. Objects MAY appear in the message in any permissible order.

Note: Any of the following messages may potentially include a MESSAGE_ID_ACK object. If that is the case, these objects should immediately follow the INTEGRITY object as per the RSVP Refresh Reductions Specification. Since INTEGRITY is optional for the demo these objects will follow the RSVP header.

6.2.1 Path Message (Msg Type = 1)

A Path message which establishes a bi-directional UNI connection MUST contain the following objects.

MESSAGE_ID
SESSION
RSVP_HOP
TIME_VALUE
GENERALIZED_LABEL_REQUEST
SENDER_TEMPLATE
SENDER_TSPEC
UPSTREAM_LABEL

6.2.2 Resv Message (Msg Type = 2)

A RESV message which reserves a bi-directional UNI connection MUST contain the following objects:

MESSAGE_ID
SESSION
RSVP_HOP
TIME_VALUE
RESV_CONFIRM
FF_STYLE
FLOWSPEC
FILTER_SPEC
GENERALIZED_LABEL

6.2.3 PathErr Message (Msg Type = 3)

A PathErr message which deletes a connection (by terminating UNI-C or UNI-N) MUST contain the following objects:

MESSAGE_ID
SESSION
ERROR_SPEC (with Path_state_removed flag set)
SENDER_TEMPLATE
SENDER _SPEC

6.2.4 ResvErr Message (Msg Type = 4)

A ResvErr Message which used to indicate error in response to a connection request MUST contain the following objects:

MESSAGE_ID
SESSION
RSVP_HOP
ERROR_SPEC
FF_STYLE
FLOWSPEC
FILTER_SPEC

6.2.5 PathTear Message (Msg Type = 5)

A PathTear Message which deletes a connection MUST contain the following objects:

MESSAGE_ID
SESSION
RSVP_HOP
SENDER_TEMPLATE
SENDER_TSPEC

6.2.6 ResvTear Message (Msg Type = 6)

A ResvTear Message which indicates a request for deletion MUST contain the following objects:

MESSAGE_ID
SESSION
RSVP_HOP
STYLE
FLOWSPEC
FILTER_SPEC

6.2.7 ResvConf (Msg Type = 7)

A ResvConf Message which confirms the receipt of a Resv message MUST contain the following objects:

MESSAGE_ID
SESSION
ERROR_SPEC
RESV_CONFIRM
STYLE
FLOWSPEC
FILTER_SPEC

6.2.8 Ack Message (Msg Type = 13)

An Ack message acknowledge receiving of a message MAY contain one or more of MESSAGE_ID_ACK object.

6.3 RSVP Objects

For the interoperability event, the following Objects are mandatory. Any objects not mentioned will be treated as optional.

The Object Format is defined in [RFC2205] as a (n + 1) x 32-bit field:

```
      0             1              2             3
+-------------+-------------+-------------+-------------+
|       Length (bytes)      |  Class-Num  |   C-Type    |
+-------------+-------------+-------------+-------------+
|                                                       |
//                  (Object contents)                  //
|                                                       |
+-------------+-------------+-------------+-------------+
```

where the length indicates the total object length in bytes.

6.3.1 GENERALIZED_LABEL

The format of the generalized label is given in [5]. Note that this draft, which has since been replaced by newer version(s), is also available as OIF contribution oif2001.274.

The format of the label for SDH and/or SONET TDM link 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               S               |   U   |   K   |   L   |   M   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Specific examples relevant to the SuperComm demonstration are given, below:

```
+----------------------------------------------------------+
| OC48c SONET LABEL                                        |
+----------------------------------------------------------+
| Class                        16                          |
| C-Type                       2                           |
| S                            1 (start with first STS-1)  |
| U                            0                           |
| K                            0                           |
| L                            0 (ignored)                 |
| M                            0 (ignored)                 |
+----------------------------------------------------------+
```

```
+----------------------------------------------------------+
| OC48 SONET LABEL                                         |
+----------------------------------------------------------+
| Class                        16                          |
| C-Type                       2                           |
| SUKLM                        1,0,0,0,0 (first STS-1)     |
| SUKLM                        2,0,0,0,0 (second STS-1)    |
//                      ...                               //
| SUKLM                        48,0,0,0,0 (48th STS-1)     |
+----------------------------------------------------------+
+-------------------------------------------------------+
| STM-16c SDH LABEL                                     |
+-------------------------------------------------------+
| Class                        16                       |
| C-Type                       2                        |
| S                            1 (start with first STM-1) |
| U                            1 (VC-4)                 |
| K                            1 (VC-4-4c)              |
| L                            0 (ignored)              |
| M                            0 (ignored)              |
+-------------------------------------------------------+

+-------------------------------------------------------+
| STM-16 SDH LABEL                                      |
+-------------------------------------------------------+
| Class                        16                       |
| C-Type                       2                        |
| SUKLM                        1,1,1,0,0 (first STM-1)  |
| SUKLM                        2,1,1,0,0 (second STM-1) |
//                      ...                            //
| SUKLM                        16,1,1,0,0 (16th STM-1)  |
+-------------------------------------------------------+
```

6.3.2 UPSTREAM_LABEL

The UPSTREAM_LABEL has exactly the same format and value as Generalized Label except that the class should be set to 26.

6.3.3 SESSION

The SESSION object is described in [3]. For the SuperComm Interim UNI, its format is as follows:

```
+-------------------------------------------------------+
| LSP_TUNNEL_IPv4 SESSION                               |
+-------------------------------------------------------+
| Class                         1                       |
| C-Type                        7 (LSP TUNNEL IPv4)     |
| IPv4 tunnel end point address   Destination  UNI-C ONA
|

| Tunnel ID                     1-(2^^16-1)             |
| Extended Tunnel ID            Source UNI-C ONA     |
+-------------------------------------------------------+
```

As can be seen the SESSION object carries the source and destination ONA. The Tunnel ID is a 16-bit identifier that remains constant over the life of a UNI connection. It takes values from the *range* indicated above.

6.3.4 RSVP_HOP

The RSVP_HOP object is used to carry the port identifier (see Section 4).

According to [RFC2205], the RSVP_HOP carries the IP address of the interface through which the last RSVP-capable node forwarded the RSVP message and a logical outgoing interface handle (LIH). An RSVP_HOP object is denoted as a PHOP ("previous hop") object for downstream messages or as a NHOP ("next hop") object for upstream messages.

Downstream messages are defined as those traveling in the direction from source UNI-C to destination UNI-C; upstream from destination UNI-C to source UNI-C. So, for example, for Path messages the

RSVP_HOP object is denoted as PHOP and for Resv messages as NHOP. Note that this terminology is derived from the original RSVP specification [7] which defines directionality with respect to the direction of (unidirectional) data flow, which is assumed to be from the source UNI-C to the destination UNI-C.

The Logical Interface Handle (LIH) is used to identify the logical outgoing interface on which the reservation is required. **A node receiving an LIH in a Path message saves its value and returns it in the HOP objects of subsequent Resv messages sent to the node that originated the LIH**.

For the SuperComm Interim UNI, the PHOP/NHOP will carry the UNI-C or UNI-N IPCC and the LIH will carry a **port identifier**. The port identifier is a 32-bit number unique within a network element. The semantics of the port identifier and its use are described in Section 4.

```
+---------------------------------------------------------+
| IPv4 RSVP_HOP                                           |
+---------------------------------------------------------+
| Class                         3                         |
| C-Type                        1 (LSP TUNNEL IPv4)       |
| PHOP/NHOP                     UNI-C/UNI-N IPCC          |
| Logical Interface Handle      Port Identifier           |
+---------------------------------------------------------+
```

The following figure explains the use of the RSVP_HOP according to the above specification.

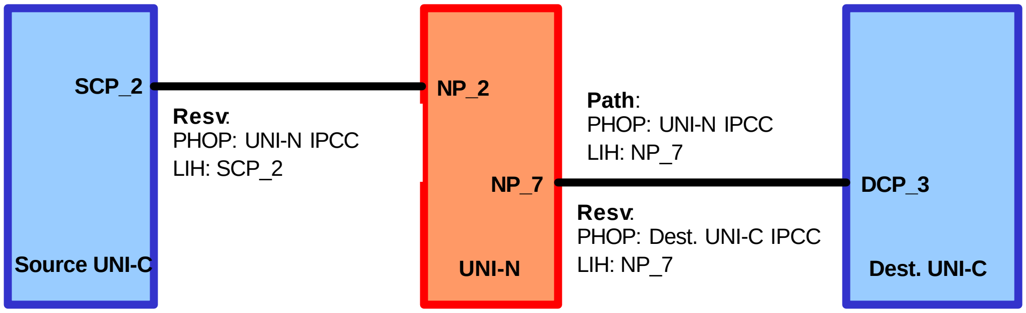


Figure 12: Use of RSVP_HOP object

6.3.5 TIME_VALUES

```
+---------------------------------------------------------+
| TIME_VALUES                                             |
+---------------------------------------------------------+
| Class                         5                         |
| C-Type                        1                         |
| Refresh Period                30000 (30 seconds)        |
+---------------------------------------------------------+
```

Note: At the last OIF meeting in Tampa it was agreed that infinite timers should be supported for UNI 1.0. This implies hard state operation for RSVP. The default operation for the SuperComm Interim UNI SHOULD use finite timers. Use of infinite timers is optional for the demo. This recommendation is made to allow time for a complete analysis and specification of modifications to RSVP for hard state operation.

6.3.6 GENERALIZED_LABEL_REQUEST WITH SONET LABEL RANGE

6.3.6.1 Generalized Label Request with SONET/SDH Label Range

The format of a Generalized Label Request with SONET/SDH Label Range ~~-~~is given in [5]. ~~:~~Note that this draft, which has since been replaced by newer version(s), is also available as OIF contribution oif2001.274.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class-Num (19)|C-Type (5)[TBA]|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| LSP Enc. Type |    Reserved   |             G-PID             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             RNC               |  Signal Type  |Rsrved.|  RGT  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

See [GMPLS-SIG.02], [4,5], for a description of parameters. For the SuperComm Interim UNI, **the object formats described in [5] are applicable, unless explicitly stated otherwise**.

The **LSP Encoding Type** is an 8 bit field which indicates the encoding of the connection being requested over the UNI. Permitted values and their meaning can be found in [5].

Note that the default LSP encoding type is SONET (6). Other values may be optionally supported by mutual agreement of UNI-C and UNI-N devices.

The **Generalized PID (G-PID)** is a 16 bit number which identifies the payload carried by the connection, i.e., an identifier of the client layer for that UNI connection. This is used by the UNI-C nodes at the endpoints of a connection. Standard Ethertype values are used for packet and Ethernet connections; other values are listed in the following table. **Note that this table modifies the one shown in [5] by adding more specific values for the Packet Over SONET (POS) option**.

Value	Type	Technology
0	Unknown	All
1	DS1 SF	ANSI-PDH
2	DS1 ESF	ANSI-PDH
3	DS3 M23	ANSI-PDH
4	DS3 C-Bit Parity	ANSI-PDH
5	Asynchronous mapping of E4	SDH
6	Asynchronous mapping of DS3/T3	SDH
7	Asynchronous mapping of E3	SDH
8	Bit synchronous mapping of E3	SDH
9	Byte synchronous mapping of E3	SDH
10	Asynchronous mapping of DS2/T2	SDH
11	Bit synchronous mapping of DS2/T2	SDH
12	Byte synchronous mapping of DS2/T2	SDH
13	Asynchronous mapping of E1	SDH
14	Byte synchronous mapping of E1	SDH
15	Byte synchronous mapping of 31 * DS0	SDH
16	Asynchronous mapping of DS1/T1	SDH
17	Bit synchronous mapping of DS1/T1	SDH
18	Byte synchronous mapping of DS1/T1	SDH
19	Same as 12 but in a VC-12	SDH
20	Same as 13 but in a VC-12	SDH
21	Same as 14 but in a VC-12	SDH
22	ATM mapping	SDH, SONET
22	DS1 SF Asynchronous	SONET
23	DS1 ESF Asynchronous	SONET
24	DS3 M23 Asynchronous	SONET

25	DS3 C-Bit Parity Asynchronous	SONET
26	VT	SONET
27	STS	SONET
28	**POS - No Scrambling, 16 bit CRC**	**SONET**
29	**POS - No Scrambling, 32 bit CRC**	**SONET**
30	**POS - Scrambling, 16 bit CRC**	**SONET**
31	**POS - Scrambling, 32 bit CRC**	**SONET**
32	Ethernet	Lambda, Fiber
33	SDH	Lambda, Fiber
34	SONET	Lambda, Fiber
35	Digital Wrapper	Lambda, Fiber
36	Lambda	Fiber

6.3.6.2 Encoding for OC48c Path Termination SONET Generalized Label Request

```
+--------------------------------------------------------+
| GLR WITH SONET LABEL RANGE                             |
+--------------------------------------------------------+
| Class                         19                       |
| C-Type                         5                       |
| LSP Enc. Type                  6 (SONET)               |
| GPID                          *                        |
| RNC                           48                       |
| Signal Type                   6 (STS-1)                |
| RGT                           2 (contiguous standard)  |
+--------------------------------------------------------+
```

The encoding example shown above denotes an "OC-48c Path Signal", i.e. an OC-48 signal with Path layer transparency (RNC=48, Signal Type=6 & RGT=2). This is consistent with the definition of the Generalized Label Request Object encoding in [5]. Note that the default LSP encoding type is SONET (6). Other values may be optionally supported by mutual agreement of UNI-C and UNI-N devices.

Generalized PID values should be programmable by the clients, in accordance with the values shown in the above table.

6.3.6.3 Encoding for STM-16c Path Termination SDH Generalized Label Request

```
+--------------------------------------------------------+
| GLR WITH SDH LABEL RANGE                               |
+--------------------------------------------------------+
| Class                         19                       |
| C-Type                         5                       |
| LSP Enc. Type                  5 (SDH)                 |
| GPID                          *                        |
| RNC                            16                      |
| Signal Type                   8 (STM-1)                |
| RGT                           2 (contiguous standard)  |
+--------------------------------------------------------+
```

The encoding example shown above denotes an "STM-16c Path Signal" (ie. RNC=16, Signal Type=8 & RGT=2). This is consistent with the definition of the Generalized Label Request Object encoding in [5]. Note that the default LSP encoding type is SDH (5).

Generalized PID values should be programmable by the clients.

6.3.6.4 Encoding for OC48 Path Termination SONET Generalized Label Request

```
+--------------------------------------------------------+
| GLR WITH SONET LABEL RANGE                             |
+--------------------------------------------------------+
| Class                         19                       |
| C-Type                         5                       |
| LSP Enc. Type                  6 (SONET)               |
| GPID                          *                        |
| RNC                            48                      |
| Signal Type                   6 (STS-1)                |
| RGT                           0 (no concatenation)     |
+--------------------------------------------------------+
```

The encoding example shown above denotes an "OC-48 Path Signal" (ie. RNC=48, Signal Type=6 & RGT=0). This is consistent with the definition of the Generalized Label Request Object encoding in [5].

6.3.6.5 Encoding for STM-16 Path Termination SDH Generalized Label Request

```
+--------------------------------------------------------+
| GLR WITH SDH LABEL RANGE                               |
+--------------------------------------------------------+
| Class                         19                       |
| C-Type                         5                       |
| LSP Enc. Type                  5 (SDH)                 |
| GPID                          *                        |
| RNC                            16                      |
| Signal Type                   8 (STM-1)                |
| RGT                           0 (no concatenation)     |
+--------------------------------------------------------+
```

The encoding example shown above denotes an "STM-16 Path Signal" (ie. RNC=16, Signal Type=8 & RGT=0). This is consistent with the definition of the Generalized Label Request Object encoding in [5]. Note that the default LSP encoding type is SDH (5).

Generalized PID values should be programmable by the clients.

6.3.7 SENDER_TEMPLATE

```
+--------------------------------------------------------+
| LSP_TUNNEL_IPv4 SENDER_TEMPLATE                        |
+--------------------------------------------------------+
| Class                         11                       |
| C-Type                        7                        |
| IPv4 Tunnel Sender address    Originating UNI-C ONA    |
| LSP ID                        *                        |
+--------------------------------------------------------+
```

The LSP_ID is a 16-bit identifier selected by the ingress node. Since the UNI does not currently support modification operation, the LSP_ID should not be modified for the duration of a connection.

6.3.8 SENDER_TSPEC

The IntServ object defined for guaranteed QoS Service [12] will be used for the OIF UNI. This object contains fields which are relevant to packet data streams described by a token bucket. The only relevant parameter for the SuperComm Interim UNI is that of connection bandwidth, hence only the peak data rate field will be examined; all other fields should be ignored.

6.3.8.1 SENDER_TSPEC Object for Guaranteed QoS Service

```
    ~~31          24 23          16 15           8 7            0~~
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
1  |             Length            | Class-Num (12)|   C-Type (2)  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
2  | 0 (a) |    reserved           |             7 (b)             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
3  |    1  (c)     |0| reserved    |             6 (d)             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
4  |   127 (e)     |    0 (f)      |             5 (g)             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
5  |  Token Bucket Rate [r] (32-bit IEEE floating point number)    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
6  |  Token Bucket Size [b] (32-bit IEEE floating point number)    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
7  |  Peak Data Rate [p] (32-bit IEEE floating point number)       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
8  |  Minimum Policed Unit [m] (32-bit integer)                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
9  |  Maximum Packet Size [M]  (32-bit integer)                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(a) - Message format version number (0)

(b) - Overall length (7 words not including header). The value of this field is 7, indicating the length of the object *in words*.

(c) - Service header, service number 1 (default/global information)

(d) - Length of service 1 data, 6 words not including header. The value of this field is set to 6.

(e) - Parameter ID, parameter 127 (Token_Bucket_TSpec). The value of this field is 127.

(f) - Parameter 127 flags (none set). The value of this field is 0.

(g) - Parameter 127 length, 5 words not including header. The value of this field is 5.

6.3.8.2 Encoding for SENDER_TSPEC requesting OC48c Path Termination as Guaranteed QoS Service

```
+---------------------------------------------------------+
| SENDER_TSPEC                                            |
+---------------------------------------------------------+
| Class                         12                        |
| C-Type                         2 (int-serv)             |
| Token Bucket Rate [r]         0 (ignored)               |
| Token Bucket Size [b]         0 (ignored)               |
| Peak Data Rate [p]            0x4D9450C0 (OC48)         |
| Minimum Policed Unit [m]      0 (ignored)               |
| Maximum Packet Size [M]       0 (ignored)               |
+---------------------------------------------------------+
```

6.3.9 RESV_CONFIRM

```
+---------------------------------------------------------+
| RESV_CONFIRM                                            |
+---------------------------------------------------------+
| Class                         15                        |
| C-Type                         1                        |
| Receiver Address              Terminating UNI-C's ONA   |
+---------------------------------------------------------+
```

When replying to a bi-directional connection request indicated by a Path message, a destination UNI-C MUST insert a RESV_CONFIRM object in the Resv Message and SHOULD wait for the ResvConf message before initiating data transmission. **A destination UNI-C SHOULD stop inserting the ResvConfirm object in refresh messages after it receives a matching ResvConf Message**. UNI-C and UNI-N nodes MUST wait for the ResvConf message before enabling any SONET/SDH alarm monitoring and reporting.

6.3.10 FF_STYLE

```
+---------------------------------------------------------+
| FIXED FORMAT_STYLE                                      |
+---------------------------------------------------------+
| Class                           8                       |
| C-Type                          1                       |
| Flags                           0 (reserved)            |
| Option Vector                   01010b (Fixed Format)   |
+---------------------------------------------------------+
```

6.3.11 FLOWSPEC

6.3.11.1 FLOWSPEC Object Format for Guaranteed Service

```
     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
    ~~31           24 23             16 15             8 7           0~~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
1   |              Length             |Class-Num (9) |   C-Type (2)  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
2   | 0 (a) |    Unused               |            10 (b)            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
3   |    2  (c)     |0| reserved      |             9 (d)            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
4   |   127 (e)     |    0 (f)        |             5 (g)            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
5   |  Token Bucket Rate [r] (32-bit IEEE floating point number)     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
6   |  Token Bucket Size [b] (32-bit IEEE floating point number)     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
7   |  Peak Data Rate [p] (32-bit IEEE floating point number)        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
8   |  Minimum Policed Unit [m] (32-bit integer)                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
9   |  Maximum Packet Size [M]  (32-bit integer)                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
10  |     130 (h)   |    0 (i)        |            2 (j)             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
11  |  Rate [R]  (32-bit IEEE floating point number)                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
12  |  Slack Term [S]  (32-bit integer)                              |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

         (a) - Message format version number (0)
         (b) - Overall length (9 words not including header)
```

```
        (c) - Service header, service number 2 (Guaranteed)
        (d) - Length of per-service data, 9 words not including per-
   service
              header
        (e) - Parameter ID, parameter 127 (Token Bucket TSpec)
        (f) - Parameter 127 flags (none set)
        (g) - Parameter 127 length, 5 words not including parameter
   header
        (h) - Parameter ID, parameter 130 (Guaranteed Service RSpec)
        (i) - Parameter 130 flags (none set)
        (j) - Parameter 130 length, 2 words not including parameter
   header
```

6.3.11.2 Encoding for FLOWSPEC Requesting OC48c Path Termination as Guaranteed QoS Service

```
+---------------------------------------------------------+
| FLOWSPEC                                                |
+---------------------------------------------------------+
| Class                          9                        |
| C-Type                         2 (Int-serv)             |
| Token Bucket Rate [r]          0 (ignored)              |
| Token Bucket Size [b]          0 (ignored)              |
| Peak Data Rate [p]            0x4D9450C0 (OC48)         |
| Minimum Policed Unit [m]       0 (ignored)              |
| Maximum Packet Size [M]        0 (ignored)              |
| Rate [R]                       0 (ignored)              |
| Slack Term [S]                 0 (zero)                 |
+---------------------------------------------------------+
```

6.3.12 FILTER_SPEC

```
+---------------------------------------------------------+
| LSP_TUNNEL_IPv4 FILTER_SPEC                             |
+---------------------------------------------------------+
| Class                          10                       |
| C-Type                          7                       |
| IPv4 Tunnel Sender address     Originating UNI-C ONA    |
| LSP ID                         *                        |
+---------------------------------------------------------+
```

The LSP_ID should not be modified for the duration of a connection. Its value should match that in the SENDER_TEMPLATE object.

6.3.13 ERROR_SPEC

All the Error values carries the globally-defined sub-code, i.e. the ssur [RFC2205] SHALL be set to 0010b. This specification also extends these error code and values to be used in messages other then ResvErr messages.

```
              +---------------------------------------------------------+
              | IPv4 ERROR_SPEC                                         |
              +---------------------------------------------------------+
              | Class                              6                    |
              | C-Type                             1 (IPv4)             |
              | IPv4 Error node address              IPCC address of node
   detecting the error                         |
              | Flags                              0 or 0x4             |
              |                                (Path_state_removed)     |
              | Error Code                         See below            |
              | Error Value                        See below            |
              +---------------------------------------------------------+
              +--------------------------------------------------------+
              | Error Code      00 (Confirmation)                      |
              +--------------------------------------------------------+
              | sub-code        0                                      |
              +--------------------------------------------------------+

              +--------------------------------------------------------+
              | Error Code       01 (Admission control failure)        |
              +--------------------------------------------------------+
              | sub-code          2 (bandwidth not available)          |
              +--------------------------------------------------------+

              +--------------------------------------------------------+
              | Error Code       02 (Policy Control Failure)           |
              +--------------------------------------------------------+
              | sub-code (new)    1 (Unauthorized Sender)              |
              | sub-code (new)    2 (Unauthorized Receiver)            |
              +--------------------------------------------------------+

              +--------------------------------------------------------+
              | Error Code       03 (No path information for this Resv)|
              +--------------------------------------------------------+
              | sub-code          0                                    |
              +--------------------------------------------------------+

              +--------------------------------------------------------+
              | Error Code       04 (No sender info. for this Resv)    |
              +--------------------------------------------------------+
              | sub-code          0                                    |
              +--------------------------------------------------------+

              +--------------------------------------------------------+
              | Error Code       06 (Unknown Reservation style)        |
              +--------------------------------------------------------+
              | sub-code          0                                    |
              +--------------------------------------------------------+


              +--------------------------------------------------------+
              | Error Code       12 (Service Preempted)                |
              +--------------------------------------------------------+
              | sub-code (new)    1 (Network Initiated, normal)        |
              +--------------------------------------------------------+
```

Sub-code 1 (Network Initialed, normal) SHALL be used for a network initiated deletion if there does not exist more specific Error Value to described the reason for the deletion.

```
+--------------------------------------------------------+
| Error Code       13 (Unknown Object Class)             |
+--------------------------------------------------------+
| Error Value       c-number and c-type                  |
+--------------------------------------------------------+


+--------------------------------------------------------+
| Error Code       14 (Unknown C-Type)                   |
+--------------------------------------------------------+
| Error Value       c-number and c-type                  |
+--------------------------------------------------------+


+--------------------------------------------------------+
| Error Code       21 (Traffic Control Error)           |
+-------------------------------------------------------+
| sub-code          2 (service unsupported)             |
| sub-code          3 (bad flowspec value)              |
| sub-code          4 (bad tspec value)                 |
+-------------------------------------------------------+

+-------------------------------------------------------+
| Error Code       23 (RSVP System Error)               |
+-------------------------------------------------------+
| sub-code (new)    1 (Max retransmission exceeded)     |
+-------------------------------------------------------+
```

```
Sub-code 1 SHALL be used to indicate an error when the re-
transmission of a RSVP message has exceeded the allowed number.
```

```
          +-------------------------------------------------------+
          | Error Code       24 (Routing Problem)                 |
          +-------------------------------------------------------+
          | sub-code          5 (No route to destination)         |
          | sub-code          6 (Unacceptable Label values)       |
          | sub-code          9 (Label allocation failure)        |
          +-------------------------------------------------------+


          +-------------------------------------------------------+
          | Error Code       25 (Notify Error)                    |
          +-------------------------------------------------------+
          | sub-code                                              |
          +-------------------------------------------------------+
```

6.3.14 MESSAGE_ID

The MESSAGE_ID and MESSAGE_ID_ACK objects are used to support acknowledgments and reliable RSVP message delivery. They are defined in [] [9].

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class-Num (23)|   C-Type (1)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Flags (0x01) |                      Epoch                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Message_Identifier                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The meaning and value of the above fields is shown below.

Flags (8 bits): For the SuperComm Interim UNI the “ACK_Desired” (0x01) flag will always be set to indicate that the sender requests the receiver to send an acknowledgment for the message.

Epoch (24 bits): A value that indicates when the Message_Identifier sequence has reset. SHOULD be randomly generated each time the network element reboots or the RSVP agent is restarted. The value SHOULD NOT be the same as was used when the node was last operational. This value MUST NOT be changed during normal operation.

Message_Identifier (32 bits): The Message Identifier MUST be unique on a per object generator's **IPCC** address basis. When combined with the message generator's IPCC address, the Message_Identifier field uniquely identifies a message. The values placed in this field change incrementally and only decrease when the Epoch changes or when the value wraps.

Message identification and acknowledgment is done on a per hop basis. *MESSAGE_ID_ACK objects may be sent piggy-backed in unrelated RSVP messages or in RSVP Ack messages*.

6.3.15 MESSAGE_ID_ACK

The MESSAGE_ID_ACK object is used to acknowledge the receipt of messages containing MESSAGE_ID objects that were sent with the ACK_Desired flag set. Note that this flag is always set for the SuperComm Interim UNI.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class-Num (24)|   C-Type (1)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Flags (0x00) |                      Epoch                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Message_Identifier                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

No flags are currently defined. This field MUST be set to zero in transmission and ignored on receipt. The Epoch and Message_Identifier fields are copied from the message being acknoweledged.

6.4 Refresh Considerations

This section summarizes the relation of the SuperComm Interim UNI to the RSVP Refresh Reduction procedures described in [9]. It also add some clarifications regarding the operation of RSVP refresh messages.

For the purpose of the SuperComm Interim UNI, only the message acknowledgment procedures of [9] should be supported. This is achieved through the use of the Ack message (6.2.8), the MESSAGE_ID object (6.3.14) and the MESSAGE_ID_ACK object (6.3.15). In particular, message acknowledgments can be carried either in ACK messages or piggybacked in other RSVP messages. **For the purpose of the SuperComm Interim UNI RSVP messages other than the ACK message SHOULD include no more than one MESSAGE_ID_ACK objects**, as stated in section 6.2. ACK messages may carry more than one MESSAGE_ID_ACK objects.

The following extensions are *not required*:

- RSVP Bundle Messages
- Summary Refresh Extension (Srefresh messages)
- MESSAGE_ID_NACK object

The following information from [9] is relevant to the SuperComm Interim UNI.

For the purpose of reliable message transmission, RSVP messages are categorized into two types: trigger and refresh messages. Trigger messages are those RSVP messages that advertise state or any other information not previously transmitted. Trigger messages include messages advertising new state, or a modification to an existing RSVP session or reservation.

Refresh messages represent previously advertised state and contain exactly the same objects and same information as a previously transmitted message, and are sent over the same path. Only Path and Resv messages can be refresh messages. Refresh messages are identical to the corresponding previously transmitted message, with some possible exceptions. Specifically, the checksum field, the flags field and the INTEGRITY object may differ in refresh messages.
When a node is sending a refresh message with a MESSAGE_ID object, it SHOULD use the same Message_Identifier value that was used in the RSVP message that first advertised the state being refreshed. When a node is sending a trigger message, the Message_Identifier value MUST have a value that is greater than any other value previously used with the same Epoch field value. A value is considered to have been used when it has been sent in any message using the associated IP address with the same Epoch field value.

Processing of refresh messages: Upon receipt of a Path refresh message a UNI-C SHOULD NOT schedule a Resv refresh. Resv refresh messages are asynchronously generated at each refresh interval or in response to Path messages modifying state.

7 Test Procedures

7.1 Connection created and deleted by source UNI-C - Mandatory

Action	Description		Specification Reference
Source Deletion			
Initiate Connection	Request a bi-directional OC48 or STM-16. Verify that the connection exists, and is maintained for longer than 3 refresh intervals.	Mandatory	3.1
Verify Transport	Check that a physical connection has been setup between the correct source and destination ports by transmitting bi-directional data across the connection. Check the correct level of transparency is provided for the connection. Data can be generated by the clients or a test instrument connected to ingress and egress ports.	Mandatory	
Delete Connection by Source	Request the deletion of the existing connection from the source client by sending a PathTear message.	Mandatory	3.2

7.2 Connection deletion by destination UNI-C - Mandatory

Action	Description		Specification Reference
Destination Deletion			
Accept Connection	Respond to a request for a bi-directional connection, verify that the connection exists, and is maintained for longer than 3 refresh intervals.	Mandatory	3.1
Delete Connection - Destination	Request the deletion of the existing connection from the destination client.	Mandatory	3.3

7.3 Connection deletion by Network - Optional

Action	Description		Specification Reference
Network Deletion			
Initiate Connection	Request a bi-directional OC48 or STM-16. Verify that the connection exists, and is maintained for longer than 3 refresh intervals.	Optional	3.1
Delete Connection - Network	Request the deletion of the existing connection from the ONE.	Optional	3.4

7.4 Connection Refused by Network - Optional

Action	Description		Specification Reference
Network Rejection			
Initiate Impossible Connection	Request a bi-directional OC48 or STM-16 to an ONA that does not exist in the address table of the ONE. Verify that the network responds with an appropriate error.	Optional	3.5

7.5 Connection Refused by destination UNI-C - Optional

Action	Description		Specification Reference
Destination Rejection			
Initiate Connection	Request a bi-directional OC48 or STM-16 to an ONA that refuses the connection. Verify the correct response.	Optional	3.6

8 Message Retransmission

Retransmission mechanisms should follow the procedures defined in section 6 of the Refresh Reductions Draft. This draft defines an exponential back-off-procedure for message retransmission as well as a maximum retry limit. Specifically, the following parameters are defined:

- Rapid retransmission interval: the initial retransmission interval for unacknowledged messages. Default value is 500ms.
- Rapid retry limit: the maximum number of times a message will be transmitted without being acknowledged. Default value is 3.
- Increment value Delta: the ratio of two successive retransmission intervals. The default value is 2.

Note: the above values are the default ones specified in [9] and [13]. For the SuperComm demonstration higher values of the rapid retransmission interval may be used by mutual agreement between UNI-C and UNI-N vendors. In particular, the values of 2sec and 3sec can be employed in conjunction with the default refresh interval of 30 sec.

According to the default values, the first retransmission interval will be 500ms, the second 1 sec and the third 2sec.

Figure 13~~Figure 12~~ depicts the retransmission of an unacknowledged Path message.

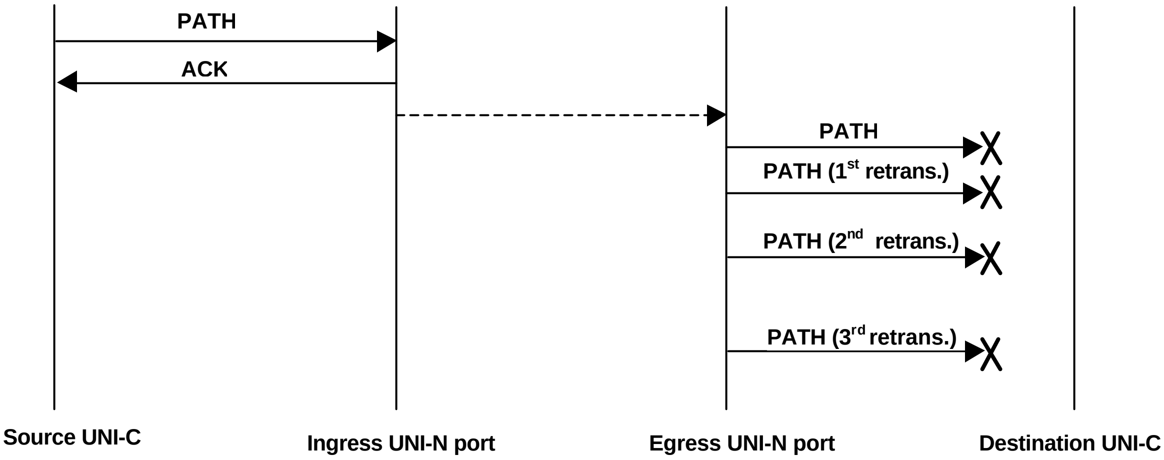
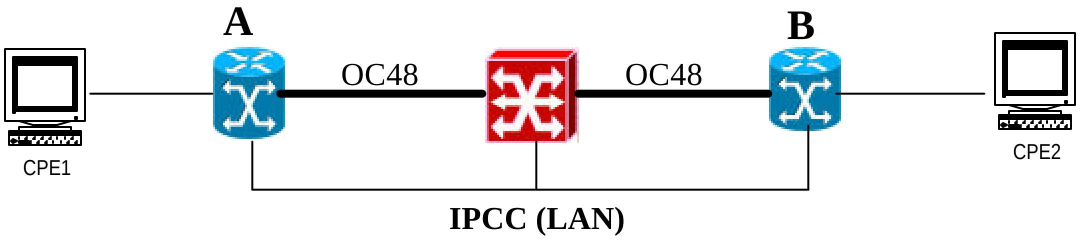


Figure 13: Retransmission of unacknowledged Path message

9 Potential Applications of the UNI Protocol

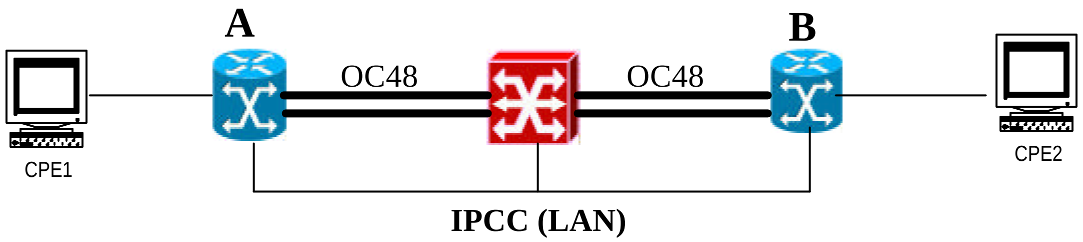
The following is a list of potential applications of the OIF UNI protocol. They have been provided by the carrier group and are listed for informational purposes. Vendors, may use these to demonstrate successful operation of UNI signaling. However, support of these applications is NOT required for successful participation in the interoperability event.

9.1 Bandwidth On-Demand for Video Transmission or Video Conferencing



In this scenario device CPE1 needs to set up a video conference with CPE2. In order to do this, Network Element A (UNI-C) needs to setup a connection with B (UNI-C) through UNI signaling. Once the conference ends, the connection is terminated.

9.2 Dynamic Bandwidth Allocation in Response to Congestion



In this scenario, UNI-Cs A and B are connected to a UNI-N device by a two OC-48 interfaces. A single OC-48 connection is set-up between A and B to carry user traffic. At some time congestion occurs between A and B. In response to this, A requests over the UNI the set-up of second connection between A and B. Congestion could be emulated by traffic injected by (packet) traffic generators connected to network elements A and/or B. In the above figure; traffic could be generated by devices CPE1 and CPE2.

10 Optional Neighbor Discovery and IP Control Channel Maintenance

The following procedures are optional for the SuperComm Interim UNI.

In the following text “Node ID” identifies the UNI-C or UNI-N IPCC address.

Neighbor discovery and IP control channel maintenance are based on the Link Management Protocol (LMP). An “LMP adjacency” is formed between the UNI-C and UNI-N using the Config message exchange. Once the configuration process is complete, the LMP Hello messages are used to maintain the control channel.

The following message encoding is defined for the SuperComm Interim UNI demonstration. Venders MUST encode messages using the values and encoding order defined below.

10.1 IP header

The format and field value of the IP header that carries an LMP message is as described in Section 6.1.1 with Protocol Id = 140 (this is a temporary value, as IANA has not yet assigned a value).

10.2 LMP Header

The format and field of the LMP header for the reference configuration is shown below:

```
+---------------------------------------------------------+
| LMP Header                                              |
+---------------------------------------------------------+
| Version                    1                            |
| Flags                      bit vector (see below)       |
| Msg Type                   (see below)                  |
| LMP Length                 in bytes (include LMP header)|
| Checksum                   (SHALL compute)              |
| Channel/Link Id            (see below)                  |
+---------------------------------------------------------+
```

```
Version:  4 bits.
Flags:  8 bits.  Bits 1, 2, and 3 MAY be set.
Msg Type:     8 bits.  The allowed message types {1, 2, 3, 4, 14,
15 16}.
LMP Length: 16 bits.
Checksum: 16 bits.
```

```
Channel/Link Id:    32 bits.  For Msg Types {1, 2, 3, 4} this is
             the IPCC address.  For Msg Types {14, 15, 16}, this is
             the ONA address.
```

10.3 Config Message (msg type 1)

The Config message is used in the negotiation phase of LMP and has the following format:
<Config Message> ::= <Common Header> <Config>

For UNI SuperComm Interoperability, the Config Object only includes the HelloConfig TLV. The format of the Config Object is thus as follows:

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Node ID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         MessageId                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|           1                 |               4               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         HelloInterval         |      HelloDeadInterval        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Node ID: 32 bits. This is the loopback address of the node.

MessageId: 32 bits.

When combined with the CCId, the MessageId field uniquely identifies the message. This value is incremented and only decreases when the value wraps. This is used for message acknowledgment.

For SuperComm Interoperability, this HelloConfig TLV is non-negotiable.

HelloInterval: 16 bits. This indicates how frequently the Hello packets will be sent and is measured in milliseconds (ms).

HelloDeadInterval: 16 bits. If no Hello packets are received within the HelloDeadInterval, the control channel is assumed to have failed. This parameter is measured in milliseconds (ms).

10.4 ConfigAck Message (msg type 2)

The ConfigAck message is used to indicate the receipt of the Config message and agreement on all parameters. The ConfigAck message has the following format:
<ConfigAck Message> ::= <Common Header> <ConfigAck>

For UNI SuperComm Interoperability, the Config Object only includes the HelloConfig TLV. The format of the Config Object is thus as follows:

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Node ID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         MessageId                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Rcv Node ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Rcv CCId                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Node ID: 32 bits.

This is the Node ID for the node sending the ConfigAck message.

MessageId: 32 bits.

This is copied from the Config message being acknowledged.

Rcv Node ID: 32 bits.

This is copied from the Config message being acknowledged.

Rcv CCId: 32 bits

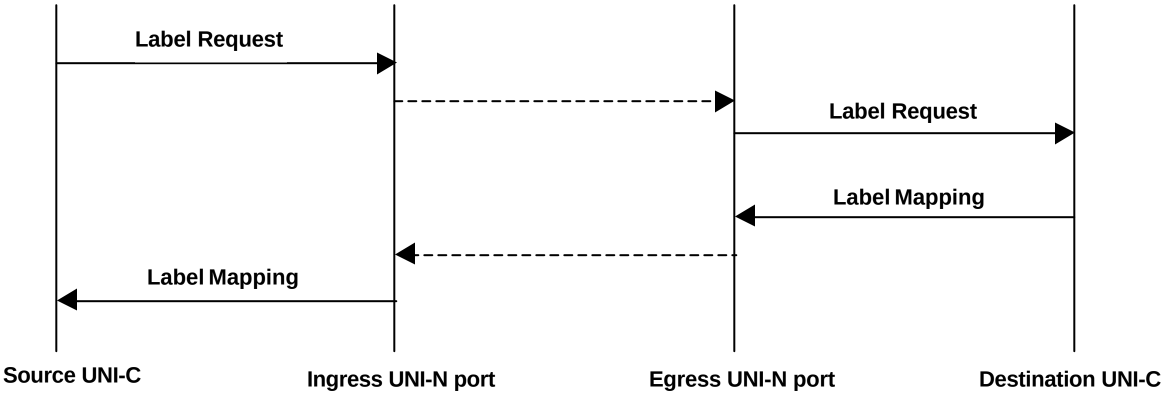
This is the Control Channel Id copied from the Common Header of the Config message being acknowledged.

11 LDP-Based Signaling

This section provides the implementation of the OIF UNI test scenarios using LDP signaling. The Messages and procedures presented here are in agreement with the latest protocol modifications proposed in [1].

11.1 *Signaling Test Scenarios*

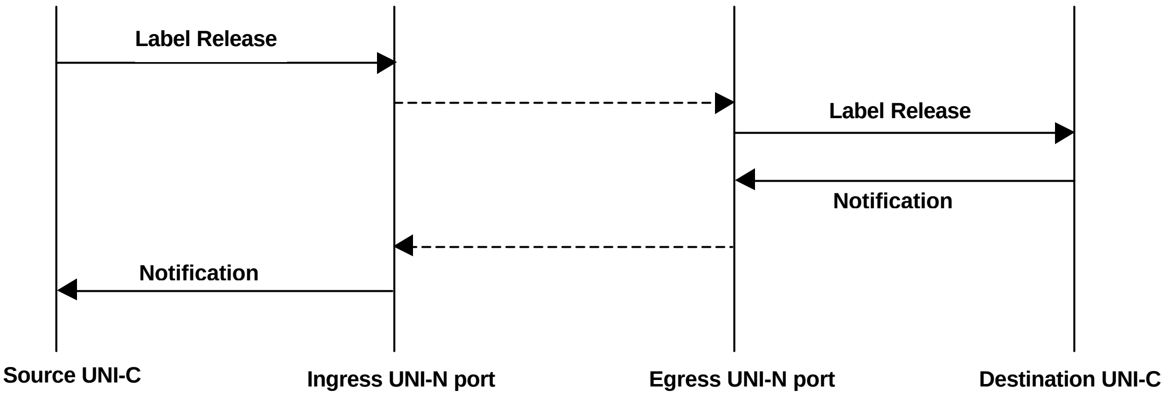
11.1.1 Successful Connection Set-Up



It is assumed that the optical network has set up the connection before sending the Label Mapping Message to the source UNI-C. Therefore the source UNI-C must not start data transmission before the Label Mapping Message is received.

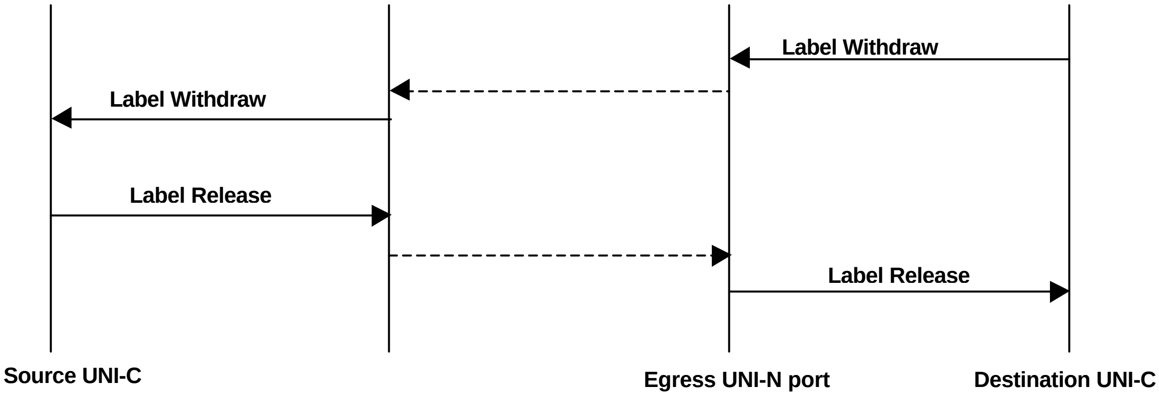
With the current LDP extensions there is no explicit indication to the destination UNI-C that the connection has been established. One way to deal with this situation is for the destination UNI-C not to send any data before it has received data from the source UNI-C. The other way is to define a new LDP message, similar to ResvConf in RSVP, to fulfill this function. The addition of new message is for further study.

11.1.2 Connection Tear-Down Initiated by Source UNI-C



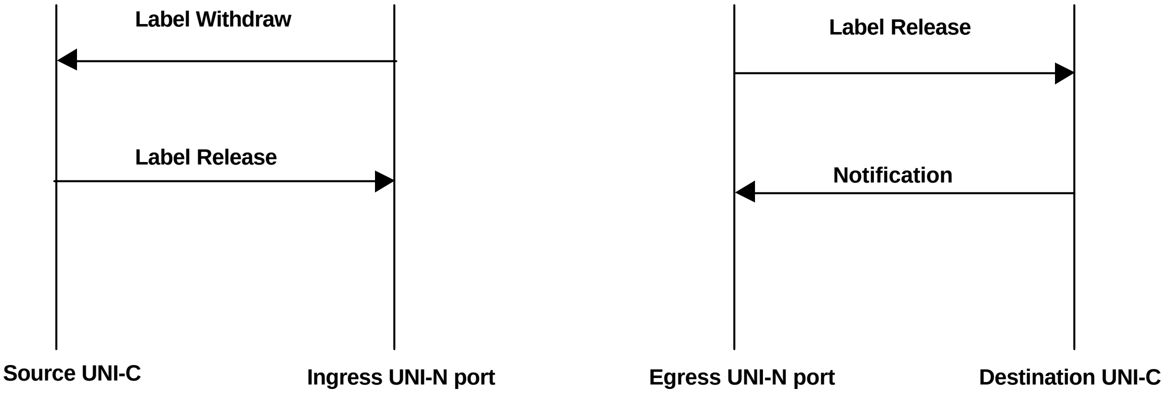
To conform with the semantics of LDP messages [2], the Label Release Message is used here connection deletion. The connection delete acknowledgement in this case is realized with a Notification Message with a status code `connection_delete_success`. The Ingress UNI-N port does not send the Notification Message to the source UNI-C before it receives an indication from the network that the connection has been successfully removed.

11.1.3 Connection Tear-Down Initiated by Destination UNI-C



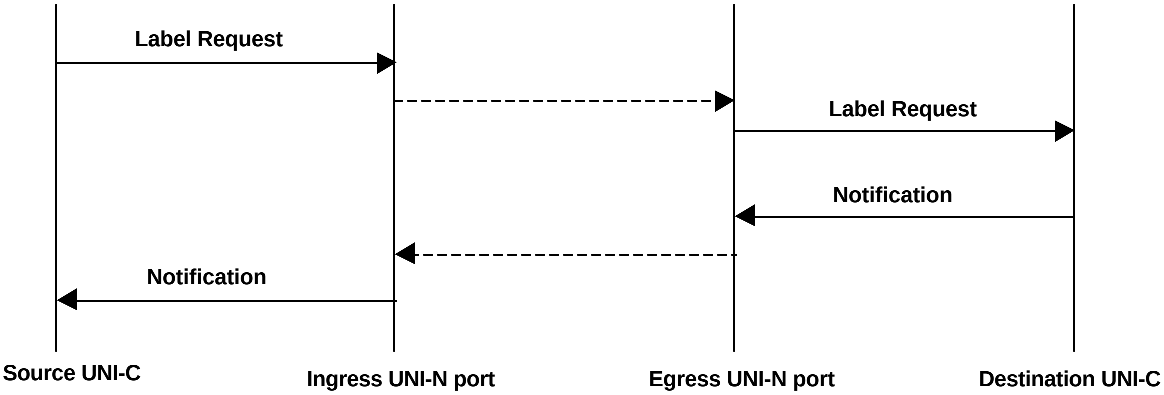
The Label Withdraw Message is used by the destination UNI-C for deleting the connection. In this case the acknowledgement of the connection deletion is achieved by sending a Label Release Message. Note that the Egress UNI-N side sends a Label Release Message acknowledging the deletion of the connection after it receives from the network the indication that connection deletion has been successfully achieved.

11.1.4 Connection Tear Down Initiated by the Network



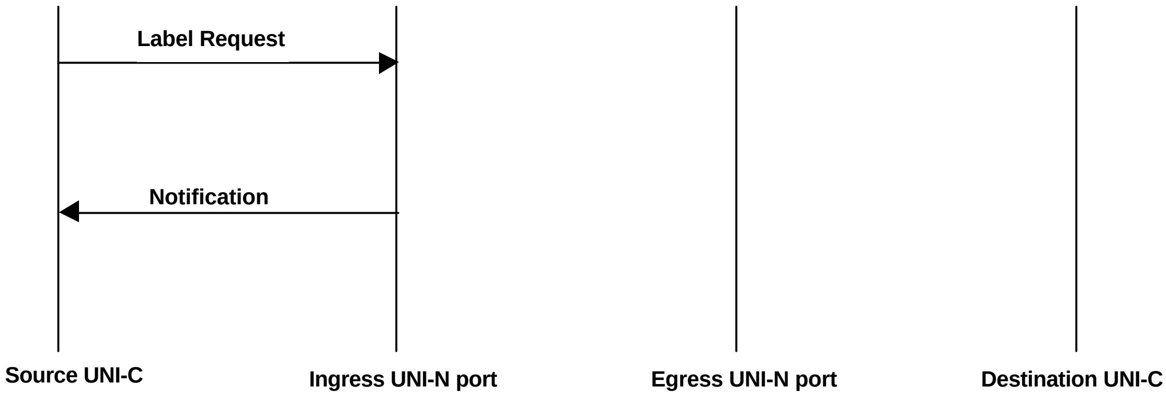
The network initiates connection tear down by simultaneously sending Label Withdraw Message and Label Release Message to source UNI-C and destination UNI-C respectively.

11.1.5 Connection Set Up Rejection by Destination UNI-C



The destination UNI-C responds to the Label Request Message by a Notification message that indicates the cause of the connection request rejection.

11.1.6 Connection Set Up Rejection by the Network



The network responds with a Notification Message that includes the appropriate status code for why the connection set up can not be completed.

11.2 LDP Message Encoding for SuperComm Demonstration

11.2.1 IP Header for UNI LDP Messages

```
+---------------------------------------------------------+
| IP Header for UNI LDP messages                          |
+---------------------------------------------------------+
|  Version                    4                           |
|  IHL                        5                           |
|  Type of Service            0111b (Network control) ??  |
|  Total Length               message length              |
|  Identification             (as defined by RFC791)      |
|  Flags                      (as defined by RFC791)      |
|  Fragment Offset            (as defined by RFC791)      |
|  Time to Live               >1                          |
|  Protocol                   6                           |
|  Header Checksum            (as defined by RFC791)      |
|  Source Address             UNI-C/UNI-N IPCC (IPv4)     |
|  Destination Address        UNI-N/UNI-C IPCC (IPv4)     |
+---------------------------------------------------------+
```

11.2.2 LDP Messages

The following LDP messages are mandatory for LDP signaling for the SuperComm demo. All other LDP messages are optional

Label Request Message

- Message Id
- FEC TLV
- Src ONA TLV
- Dest ONA TLV
- Generalized Label TLV
- Connection Id TLV
- UpStream Label TLV

Label Mapping Message

- Message Id
- FEC TLV
- Generalized Lable TLV
- UNI Label Request Message Id TLV
- Connection Id TLV

Label Release Message

- Message Id
- FEC TLV
- Connection Id

Label Withdraw Message

- Message Id
- FEC TLV
- Connection Id

Notification

- Message Id
- FEC TLV
- Connection Id TLV
- Status TLV

11.2.3 LDP Message Encoding for SuperComm Demonstration

All the LDP messages for SuperComm Demonstration have the following format.

```
 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|   Message Type              |      Message Length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Message ID                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                     Mandatory Parameters                      |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

11.2.3.1 Label Request Message Encoding

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|  Label Request (0x0401)     |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               Message ID                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              FEC TLV                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Source ONA TLV                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Destination ONA TLV                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Connection Id TLV                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Generalized Label Request TLV                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Up Stream Label TLV                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

For the purpose of the SuperComm demo, the source and the destination ONA are assigned unique IPv4 addresses. In those cases where the ONA address refer to more than one physical port, a 32-bit port identifier is used for port identification. The port identifier is encoded as an integer and must be unique within the network element.

The encoding for the source ONA TLV 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|0|   Source ONA (0x0950)     |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           IPV4 Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Port Id                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The encoding for the destination ONA 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|0|   Dest ONA (0x0951)       |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           IPV4 Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Port Id                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The encoding for the Connection Id TLV 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|0| Connection Id(0x0952)     |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Reserved                        | ActFl |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Connection Id                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

For the SuperComm demonstration, the connection Id is 32-bit unsigned integer.

The format for the Generalized Label request for SONET/SDH 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|        0x0901             |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Signal Type  |      RGT      |             RNC               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               RT                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Reserved            |          Multiplier           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The values for the Signal Type, RGT, and RNC are as described in sections 6.3.6.2-6.3.6.5 in the RSVP chapter. Both RT and Multiplier are not used at the demonstration.

The format of the label for SONET/SDH 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|        0x0902             |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               S               |   U   |   K   |   L   |   M   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The values for S, U, K, L, and M are as described in section 6.3.1
The Up Stream Label has the same format as the Label TLV. The Up Stream Label uses Type=0x0906.

11.2.3.2 Label Mapping Message Encoding

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|  Label Mapping (0x0400)     |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Message ID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            FEC TLV                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Label Request Message Id TLV                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Connection Id TLV                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Gneralized Label TLV                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

11.2.3.3 Label Release Message Encoding

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|  Label Release (0x0403)     |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Message ID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            FEC TLV                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Connection Id TLV                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

11.2.3.4 Label Withdraw Message Encoding

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|  Label Withdraew (0x0402)   |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              Message ID                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               FEC TLV                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Connection Id TLV                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

11.2.3.5 The Notification Message Encoding

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|  Notification (0x0001)      |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              Message ID                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Connection Id TLV                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Status TLV                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the Status TLV 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|0| Status (0x0300)           |      Length                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Status Code                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Message ID                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Message Type             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the status code 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|0|                 Status Data                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The status data is a 30-bit unsigned integer that specifies the status information.

If non-zero, 32-bit Message ID identifies the peer message to which the Status TLV refers. If zero, no specific peer message is being identified.

Message Type, if non-zero, the type of the peer message to which the Status TLV refers. If zero, the Status TLV does not refer to any specific message type.

11.2.3.5.1 Status Codes

The Status codes are as defined in [2].

12 References

1. User Network Interface (UNI) 1.0 Signaling Specification, *OIF Contribution OIF2000.125.3*, December 2000.
2. Cary Wright, “User Network Interface (UNI) 1.0 Interoperability Test Plan”, *OIF Contribution oif2001.81*, January 2001.
3. D. Awdu r, D-H. Gan, T. Li, G. Swallow and V. Srinivasan, “Extensions to RSVP for LSP Tunnels,” Internet Draft (Work in Progress), draft-ietf-mpls-rsvp-lsp-tunnel-08.txt, February 2001.
4. P. Ashwood-Smith, et. al, “Generalized MPLS - Signaling Functional Description,” Internet Draft (Work in Progress), draft-ietf-mpls-generalized-mpls-signaling-00.txt, November, 2000.
5. P. Ashwood-Smith, et. al, “Generalized MPLS - Signaling Functional Description,” Internet Draft (Work in Progress), draft-ietf-mpls-generalized-mpls-signaling-02.txt, March 2001 .
6. B. Mack-Crane, et al., “Enhancements to GMPLS Signaling for Optical Technologies,” Internet Draft (Work in Progress), draft-mack-crane-gmpls-signaling-enchancements-00.txt, November, 2000.
7. R. Braden, Ed., “Resource Reservation Protocol (RSVP) - Version 1 Functional Specification,” IETF RFC 2205, September, 1997.
8. P. Ashwood-Smith, et. al, “Generalized MPLS - RSVP-TE Signaling Functional Description,” Internet Draft (Work in Progress), draft-ietf-mpls-generalized-rsvp-te-00.txt, November, 2000.
9. L. Berger, et al., “RSVP Refresh Overhead Reduction Extensions,” Internet Draft (Work in Progress), draft-ietf-rsvp-refresh-reduct-05.txt, June 2000.
10. D. Pendarakis, “OIF Tampa Meeting UNI Addressing Sub-group Meeting Recommendations”, *OIF Contribution OIF2001.121*, January 2001.
11. Fong Liaw, Personal Communication.
12. J. Wroclawski, “The Use of RSVP with IETF Integrated Services”, RFC 2210, September 1997.
13. L. Berger, et al., “RSVP Refresh Overhead Reduction Extensions,” RFC 2961, April 2001.