The ATM Forum

Technical Committee

PNNI Addendum for Path and Connection Trace Version 1.1 (PACT 1.1)

With Revision Marks
Relative to af-cs-0141.000

af-cs-0141.001
February 2004

(Contents are Identical to af-cs-0141.002)

Preface

During preparation of ~~this addendum~~PACT 1.0, the Control Signalling working group was chaired by Malcolm Wiles and his successor Gert Oster. The minutes at related working group meetings were recorded by Gert Oster and his successor Thomas Cornely. The editors of this addendum were Gregory F. Wetzel, W. Tony Lau, E. Mickey Spiegel, and Shawn McAllister. The editors would like to thank the following contributors for their help with this addendum as well as all participants of the Control Signalling working group for the many days and evenings spent discussing this addendum:

Sirak Bahlbi
David Cypher
Katherine Chan
Robert B. Dianda
Andrew Dolganow
Josef Ferchenbauer
Per Ferngren
Gilad Goren
Daniel Hernandez-Ortega
W. Tony Lau
Gerhard Maegerl
Shawn McAllister
Roger Morley
Gert Oster
Lew Park
Ronald Parker
E. Mickey Spiegel
Ted Tedijanto
Gregory F. Wetzel

For PACT 1.1, the Control Signalling working group was chaired by Gert Oster and E. Mickey Spiegel. The minutes at related working group meetings were recorded by Thomas Cornély. The editor of this addendum was Peter Roberts. The editor would like to thank the following contributors for their help with this addendum as well as all participants of the Control Signalling working group for the many days and evenings spent discussing this addendum:

Thomas Cornély
Carl Rajsic
Robert B. Dianda
E. Mickey Spiegel
John Rutemiller

This specification uses three levels for indicating the degree of compliance necessary for specific functions, procedures, or coding. They are indicated by the use of key words as follows:

- **Requirement:** "Shall" indicates a required function, procedure, or coding necessary for compliance. The word "shall" used in text indicates a conditional requirement when the operation described is dependent on whether or not an objective or option is chosen.
- **Objective:** "Should" indicates an objective which is not required for compliance, but which is considered desirable.
- **Option:** "May" indicates an optional operation without implying a desirability of one operation over another. That is, it identifies an operation that is allowed while still maintaining compliance.

Preface to PACT 1.1

The PACT 1.1 specification is contained in af-cs-0141.001 and af-cs-0141.002. The contents of both documents are strictly identical; the only difference being that af-cs-0141.001 contains revision marks to the existing PACT 1.0 text of af-cs-0141.000. In the unlikely case of discrepancies between the two documents, the text of af-cs-0141.002 shall have precedence over the text of af-cs-0141.001.

The main enhancements of PACT 1.1 are listed in Section 1.3.

Table of Contents

Table of Figures

1 Introduction

1.1 Overview

[INFORMATIVE]

This addendum to PNNI v1.~~0~~ 1 “Private Network-Network Interface Specification Version 1.~~0~~1” [1] contains the description and specification of the Path and Connection Trace features. The Path and Connection Trace features provide control plane mechanisms that can be used to quickly and efficiently determine the logical nodes and logical links that new and existing connections traverse. This information is typically used for network management, in particular to debug network faults. Path Trace and Connection Trace are two separate features. They do not interact with each other. For both Path and Connection Trace, the trace is initiated at the “trace source node”, although the connection or party may have been initiated upstream from the trace source node, and tracing terminates at the “trace destination node”, although the connection or party may progress beyond the trace destination node.

The Path Trace feature is used for new connections and parties in the process of being established. Tracing may be initiated by network management for the sole purpose of determining paths that new connections and parties might take. Alternatively, these can be connections and parties that are already being established for other purposes (e.g. they may be traced to trouble shoot connection establishment problems). In the former case, the connection or party typically is never established; when the trace destination node is reached, the connection or party may be cleared rather than being connected. The Path Trace feature does not require new messages but requires the addition of the new Trace transit list information element to several PNNI signalling messages, including SETUP, ADD PARTY, CONNECT, ADD PARTY ACKNOWLEDGE, RELEASE, RELEASE COMPLETE, DROP PARTY and ADD PARTY REJECT. The Path Trace feature requires new procedures in addition to the standard PNNI call and connection control procedures.

The Connection Trace feature is used to collect information on existing connections and parties that have already been established. The Connection Trace feature requires two new messages, TRACE CONNECTION and TRACE CONNECTION ACKNOWLEDGE, and one new information element, the Trace transit list information element. The Connection Trace feature does not interact with existing call and connection control messages and procedures.

PNNI Addendum for Path and Connection Trace Version 1.1 adds the following capabilities over PNNI Addendum for Path and Connection Trace Version 1.0:

- the capability to collect Policy Routing related information during a Path Trace or Connection Trace. The Trace transit list information element includes fields to record the Network Service Categories (NSCs) supporting the connection. Refer to Policy Routing Version 1.0 [4] for a description of how NSCs are used within the network.
- the capability to trace the labels of interworking LSPs supporting an ATM connection established over an MPLS network using the procedures defined in ATM-MPLS Network Interworking Signalling Specification Version 1.0 [7]

1.2 Scope

[NORMATIVE]

The scope of this document is to specify control plane mechanisms that can be used to quickly and efficiently determine the logical nodes and logical links that new and existing connections traverse in PNNI networks. Additional information such as connection identifiers (e.g. VPIs and VCIs) may also be collected.

Path and connection trace are separate optional features of PNNI v1.~~0~~ 1 [1].

A switch supporting path trace shall implement both the path trace procedures for Point-to-Point calls and the path trace procedures for Point-to-Multipoint calls. The switch shall support path tracing of virtual channel connections (SVCCs, soft PVCCs) and virtual path connections (SVPCs, soft PVPCs).

A switch supporting connection trace shall implement both the connection trace procedures for Point-to-Point calls and the connection trace procedures for Point-to-Multipoint calls. The switch shall support connection tracing of virtual channel connections (SVCCs, soft PVCCs) and virtual path connections (SVPCs, soft PVPCs).

The decision of whether to trace a new or existing connection or party is controlled through network configuration. One configuration mechanism is defined in this specification as the path and connection trace SNMP MIB. Usage of the resulting trace information is outside the scope of this specification.

A switch supporting path trace and policy routing [4] shall be capable of tracing NSCs during a path trace. A switch supporting connection trace and policy routing shall be capable of tracing NSCs during a connection trace.

A switch supporting path trace and ATM-MPLS network interworking signalling [7] shall be capable of tracing the labels of interworking LSPs during a path trace. A switch supporting connection trace and ATM-MPLS network interworking signalling shall be capable of tracing the labels of interworking LSPs during a connection trace.

1.2.1 Support of Path and Connection Trace Version 1.1 by PNNI 1.0 Nodes

A device supporting PNNI 1.0 [5] may implement the functionality defined in this addendum by treating this addendum as if it were an optional addendum to PNNI 1.0 [5], Policy Routing Version 1.0 [4], and PNNI 1.0 Errata and PICS [6]. No new PNNI 1.1 features are required by Path and Connection Trace Version 1.1.

1.3 Known Differences from PNNI Addendum for Path and Connection Trace Version 1.0

The PNNI Addendum for Path and Connection Trace Version 1.1 is an update to the PNNI Addendum for Path and Connection Trace Version 1.0. The main difference from PNNI Addendum for Path and Connection Trace Version 1.0 is that it includes:

- The ability to record the Policy Routing Network Service Categories supporting the connection.
- The ability to trace the labels of interworking LSPs supporting a call/connection.

2 Terminology

[NORMATIVE]

2.1 Acronyms

AINI	ATM Inter-Network Interface
ATM	Asynchronous Transfer Mode
B-ICI	B-ISDN Inter Carrier Interface
CO-BI	Connection-Oriented, Bearer-Independent
DLCI	Data Link Connection Identifier
DTL	Designated Transit List
ID	Identifier
IISP	Interim Inter-switch Signaling Protocol
INE	Interworking Network Element
ITU-T	International Telecommunication Union - Telecommunication standardization sector
IUT	Implementation Under Test
LSP	Label Switched Path
MIB	Management Information Base
MPLS	Multi-Protocol Label Switching
OUI	Organizational Unique Identifier
PICS	Protocol Implementation Conformance Statement
PNNI	Private Network-Network Interface
QoS	Quality of Service
Soft PVC	Soft Permanent Virtual Connection
SUT	System Under Test
SVC	Switched Virtual Connection
TTL	Trace Transit List
UNI	User-Network Interface
VCI	Virtual Channel Identifier
VPC	Virtual Path Connection
VPCI	Virtual Path Connection Identifier
VPI	Virtual Path Identifier

2.2 Definitions

Connection Trace — A control plane mechanism that determines the logical nodes and logical links traversed by existing connections and parties that have already been established, and supporting mechanisms that provide this information to network management systems.

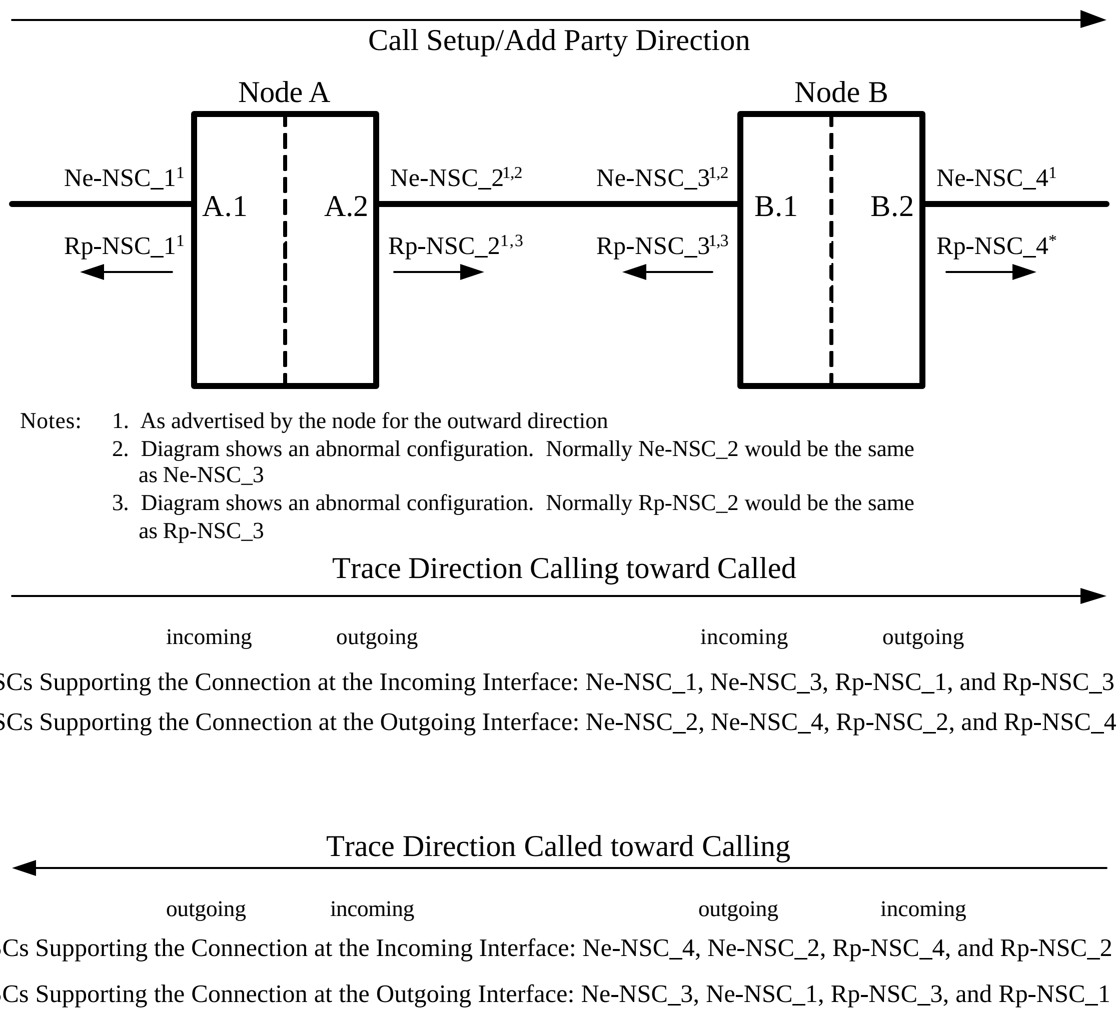
Incoming Interface — For a given node, the “incoming interface” refers to

- o for path trace, the interface from which the connection or party establishment message is received.
- o for connection trace, the interface from which the TRACE CONNECTION message is received.

NSCs Supporting The Connection At The Incoming Interface
The Ne-NSCs that are advertised by this node for the outgoing direction (in the PNNI routing sense) of the incoming interface (as defined in this section) and the Rp-NSCs that are advertised by this

	node for the outgoing direction of the selected resource partition for this connection on the incoming interface. Refer to figure 2-1.
NSCs Supporting The Connection At The Outgoing Interface	The Ne-NSCs that are advertised by this node for the outgoing direction (in the PNNI routing sense) of the outgoing interface (as defined in this section) and the Rp-NSCs that are advertised by this node for the outgoing direction of the selected resource partition for this connection on the outgoing interface. Refer to figure 2-1.
Outgoing Interface	For a given node, the "outgoing interface" refers to o for path trace, the interface on which the connection or party establishment message is sent out. o for connection trace, the interface on which the TRACE CONNECTION message is sent out.
Path Trace	A control plane mechanism that determines the logical nodes and logical links traversed by new connections and parties in the process of being established, and supporting mechanisms that provide this information to network management systems.
Trace Destination Node	The node at which connection trace or path trace is terminated for a given connection, when the trace completes normally. A trace destination node is a node whose outgoing interface for the connection is a trace destination interface
Trace Destination Interface	An interface on which a path or connection trace terminates when it completes normally. This interface is defined by any one of three conditions: 1. this interface directly supports the called party number (for path trace and connection trace towards the called party) or calling party number (for connection trace towards the calling party), e.g. Soft PVC called or calling party, 2. the next interface which the connection or party traverses (for connection trace), or the next interface on which the connection or party would be progressed towards the called party (for path trace), is not a PNNI interface (e.g., UNI, AINI, B-ICI, IISP), or 3. the next interface which the connection or party traverses (for connection trace), or the next interface on which the connection or party would be progressed towards the called party (for path trace), is administratively designated as a trace destination interface.
Trace Source Node	The node at which connection trace or path trace is initiated for a given connection. This node inserts a new Trace transit list information element into a SETUP or ADD PARTY message (for path trace), or originates a new TRACE CONNECTION message (for connection trace).
Trace Source Interface	The interface at the trace source node that is (administratively) designated as the starting point for path or connection trace of a given connection.

Saved Original Trace Transit List	The Trace transit list information element saved on the node after the ingress data has been encoded in the trace (either successfully or not).
Saved Modified Trace Transit List	The Trace transit list information element saved on the node after both ingress and egress data has been encoded in the trace (either successfully or not).



Notes:
1. As advertised by the node for the outward direction
2. Diagram shows an abnormal configuration. Normally Ne-NSC_2 would be the same as Ne-NSC_3
3. Diagram shows an abnormal configuration. Normally Rp-NSC_2 would be the same as Rp-NSC_3

NSCs Supporting the Connection at the Incoming Interface: Ne-NSC_1, Ne-NSC_3, Rp-NSC_1, and Rp-NSC_3

NSCs Supporting the Connection at the Outgoing Interface: Ne-NSC_2, Ne-NSC_4, Rp-NSC_2, and Rp-NSC_4

NSCs Supporting the Connection at the Incoming Interface: Ne-NSC_4, Ne-NSC_2, Rp-NSC_4, and Rp-NSC_2

NSCs Supporting the Connection at the Outgoing Interface: Ne-NSC_3, Ne-NSC_1, Rp-NSC_3, and Rp-NSC_1

Figure 2-1 Trace Direction and NSC Recording

3 Information Elements

The following information element definition is common for both the path trace and connection trace features.

3.1 Trace Transit List

[NORMATIVE]

The purpose of the Trace transit list information element is to indicate the logical nodes and logical links that a connection or party has traversed through peer groups at the lowest level. Sections 4 and 5 describe the messages and procedures for constructing trace transit lists.

8	7	6	5	4	3	2	1	Octet
Trace transit list								
1	1	1	0	1	1	1	0	1 (Note 1)
Information element identifier								
1 ext	Coding standard		IE Instruction Field					2
Length of trace transit list contents								3
Length of trace transit list contents (continued)								4
C	X	V	A	Ne	Rp	I	L	5 (Note 2)
Flags								
Trace status								6
0	0	0	0	1	0	1	1	7* (Note 1)
Trace Source logical port indicator								
Trace Source logical port identifier								7.1* to 7.4*
0	0	0	0	0	0	1	0	8* (Notes 1, 3, 4)
VPI/VCI indicator								
VPI								8.1*
VPI (continued)								8.2*
VCI								8.3*
VCI (continued)								8.4*
0	0	0	0	1	0	1	0	9* (Notes 1, 3, 5, 6)
DLCI indicator								
0 Ext	0 Spare	DLCI (Most significant 6 bits)						9.1*
0/1 Ext	DLCI (2nd most significant 4 bits)				0	0 Spare	0	9.2*
1 Ext	DLCI (3rd most significant 6 bits)						0 Spare	9.3* (Note 7)
0 Ext	DLCI (3rd most significant 7 bits)							9.3* (Note 8)
1 Ext	DLCI (4th most significant 6 bits)						0 Spare	9.4* (Note 8)
0	0	0	0	0	0	1	1	10* (Notes 1, 3, 9)

Call Reference indicator								
Call Reference Flag	Call Reference Value							10.1*
Call Reference Value (continued)								10.2*
Call Reference Value (continued)								10.3*
0	0	0	0	0	1	1	1	11* (Notes 1, 3, 10)
Endpoint Reference indicator								
Endpoint Reference Flag	Endpoint Reference Value							11.1*
Endpoint Reference Value (continued)								11.2*
0	0	0	0	0	0	0	1	12 (Notes 1, 3, 11)
Logical node / logical port indicator								
Logical node identifier								12.1 to 12.22
Logical port identifier								12.23 to 12.26
0	0	0	0	0	1	1	0	13* (Notes 1, 3, 12)
PNNI Trace Continuation Refusal Indicator								
0	0	0	0	1	1	0	1	14* (Notes 1, 3, 14)
Crankback Received at Trace Destination Node Indicator								
0	0	0	0	1	1	0	0	15* (Notes 1, 3, 11)
PNNI Crankback Gap indicator								
0	0	0	0	0	1	0	0	16* (Notes 1, 3, 11)
PNNI Crankback indicator								
Length of PNNI Crankback contents								16.1*
PNNI Crankback Cause								16.2*
Blocked Transit Type								16.3*
Blocked Transit Trace Information (format dependent on value of blocked transit type)								16.4* etc (Note 13)
0	0	0	0	0	1	0	1	17* (Notes 1, 3)
Vendor Specific indicator								
Length of Vendor specific contents								17.1*
Organizational Unique Identifier (OUI)								17.2* to 17.4*
Vendor specific information								17.5* etc.
0	0	0	0	1	1	1	0	18*(Notes 1,3,15)
Incoming Ne-NSC List Identifier								
Length of Incoming Ne-NSC List								18.1*
Incoming Ne-NSC Identifier Value								18.2*(Note 16)
								18.3*(Note 16)
0	0	0	0	1	1	1	1	19*(Notes 1,3,17)
Incoming Rp-NSC List Identifier								
Length of Incoming Rp-NSC List								19.1*
Incoming Rp-NSC Identifier Value								19.2*(Note 18)
								19.3*(Note 18)

0	0	0	1	0	0	0	0	20*(Notes 1,3,19)
Outgoing Ne-NSC List Identifier								
Length of Outgoing Ne-NSC List								20.1*
Incoming Ne-NSC Identifier Value								20.2*(Note 20) 20.3*(Note 20)
0	0	0	1	0	0	0	1	21*(Notes 1,3,21)
Outgoing Rp-NSC List Identifier								
Length of Outgoing Rp-NSC List								21.1*
Outgoing Rp-NSC Identifier Value								21.2*(Note 22) 21.3*(Note 22)
0	0	0	1	0	0	1	0	22*(Notes 1,23)
Interworking LSP Labels Indicator								
0	0	0	0	Receive Interworking LSP Label (Most significant 4 bits)				22.1*
Spare								
Receive Interworking LSP Label (2nd most significant 8 bits)								22.2*
Receive Interworking LSP Label (3rd most significant 8 bits)								22.3*
0	0	0	0	Transmit Interworking LSP Label (Most significant 4 bits)				22.4*
Spare								
Transmit Interworking LSP Label (2nd most significant 8 bits)								22.5*
Transmit Interworking LSP Label (3rd most significant 8 bits)								22.6*

Note 1 – The trace transit list is an ordered list. Interpretation of octet groups within the trace transit list information element is position dependent. A node may add at most one of each octet group, in order of ascending octet group number, each time it processes a message, with the following exceptions:

- Octet groups 18, 19, and 22, if present, appear after any octet groups 7 through 11 and before octet group 12.
- Octet groups 20 and 21, if present, appear immediately following octet group 12.
- A node may add multiple instances of octet group 17. When octet group 17 is present, it must immediately follow either
  - octet 6,
  - an octet group 12, unless octet group 12 is immediately followed by octet group 20 or 21,
  - an octet group 20, unless octet group 20 is immediately followed by octet group 21,
  - an octet group 21,
  - an octet group 16, or
  - another octet group 17.
- A node attempting alternate routing may add instances of octet groups 12, 20, 21, 13, 14, 15, 16, ~~through~~ and 17, in the proper order, after an octet group 16, or octet 15.
- The trace destination node may add instances of octet groups 8 through 11 or 22 (see Section 4.3.1.2)~~-~~, in order, after octet group 12, 20, 21, or one or more octet groups 16 that themselves follow octet group 12, 20, or 21. These are in addition to any instances of octet groups 8 through 11 or 22 that it adds prior to octet group 12 according to the normal sequence.

Note 2 – Flags C and X~~-~~ shall be treated as spare bits (i.e., set to zero when originated and ignored upon reception) in TRACE CONNECTION and TRACE CONNECTION

ACKNOWLEDGE messages. Flags V and L shall be treated as a spare bit in CO-BI SETUP messages and TRACE CONNECTION and TRACE CONNECTION ACKNOWLEDGE messages for bearer independent (CO-BI) calls. Flag X shall not be set to "1" when either Flag V or Flag A is set to "1". ~~All spare bits (bits 1 to 4) shall be ignored upon receipt and forwarded transparently if the information element is progressed.~~

Note 3 – Octet groups 8-~~17~~ 22 (in their entirety) may appear multiple times.

Note 4 – Octet group 8 shall be added when the interface is ATM, if and only if the Flag V for VPI/VCI trace is set to "1", with the following exception. This octet group shall not be used if the message is CO-BI SETUP or the message is TRACE CONNECTION or TRACE CONNECTION ACKNOWLEDGE tracing a bearer independent (CO-BI) connection.

Note 5 – Octet group 9 may appear (in its entirety) as many as 2 times only if the Flag V is set to "1".

Note 6 – The DLCI is 2 or 3 or 4 octets (i.e. 10 or 16 or 23 bits respectively). The standard default length of the DLCI is two octets. The extension bit mechanism is used to indicate non-default length and thus to determine the total length of the DLCI. The trace source node must be capable of interpreting this encoding, even if it does not support Frame Relay, so that it can properly decode the Trace transit list information element returned from the trace destination node.

Note 7 – This octet shall be included only when bilateral agreements allow a three octet DLCI (16 bits).

Note 8 – These octets shall both be included only when bilateral agreements allow a four octet DLCI (23 bits).

Note 9 – Octet group 10 shall be added when the connection leg on the interface is switched, if and only if the Flag A for Call/Endpoint Reference Value trace is set to "1".

Note 10 – Octet group 11 shall be added if and only if this is a point-to-multipoint connection and the Flag A for Call/Endpoint Reference Value trace is set to "1".

Note 11 – Octet group 16, or octet 15 followed by octet group 16, may be added only if the Flag C for Crankback is set to "1".

Note 12 – Octet 13 may be added only by a node which refused continuation of the trace.

Note 13 – The length of the Blocked Transit Trace Information shall be determined as the content of octet 16.1 minus 2 octets.

Note 14 – Octet 14 may be added only by a node which received a crankback from a non-PNNI interface when acting as trace destination node.

Note 15 – Octet group 18 may be added by the trace source node only if the Flag Ne for Ne-NSC tracing is set to "1" and by the intermediate and destination trace nodes only if both the Flags Ne and I are set to "1".

Note 16 – Octets 18.2 and 18.3 may be repeated multiple times, subject to the Trace transit list information element not exceeding its maximum length

Note 17 – Octet group 19 may be added by the trace source node only if the Flag Rp for Rp-NSC tracing is set to "1" and by the intermediate and destination trace nodes only if both the Flags Rp and I are set to "1".

Note 18 – Octets 19.2 and 19.3 may be repeated multiple times, subject to the Trace transit list information element not exceeding its maximum length

Note 19 – Octet group 20 may be added only if the Flag Ne for Ne-NSC tracing is set to "1".

Note 20 – Octets 20.2 and 20.3 may be repeated multiple times, subject to the Trace transit list information element not exceeding its maximum length

Note 21 – Octet group 21 may be added only if the Flag Rp for Rp-NSC tracing is set to "1".

Note 22 – Octets 21.2 and 21.3 may be repeated multiple times, subject to the Trace transit list information element not exceeding its maximum length

Note 23 – Octet group 22 shall be added when the connection leg on the interface is an ATM-MPLS network interworking interface, if and only if the Flag L for LSP label trace is set to "1", with the following exception. This octet group shall not be used if the message is CO-BI SETUP or the message is TRACE CONNECTION or TRACE CONNECTION ACKNOWLEDGE tracing a bearer independent (CO-BI) connection.

Figure 3-1 Trace transit list information element

Coding standard (octet 2)

Bits 7 6	Meaning
1 1	ATM Forum specific

Flags (octet 5)

Bit 8	C-flag Meaning
0	No crankback trace is requested
1	Crankback trace is requested

Note: Setting the C flag to 0 has the following consequences to the final Trace transit list information element returned to the source node:

- If present, the Trace transit list information element will describe only information of the final path, i.e. no data for the failed paths will be recorded.
- RELEASE, RELEASE COMPLETE, DROP PARTY, ADD PARTY REJECT will include the Trace transit list information element only if the SETUP or ADD PARTY reached the trace destination node and the trace destination node received RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT without a Crankback information element in response to the SETUP or ADD PARTY.

Bit 7	X-flag Meaning
0	Do not clear call at trace destination
1	Clear call at trace destination

Bit 6	V-Flag Meaning
0	Do not trace VPI/VCI or DLCI values
1	Trace VPI/VCI and DLCI values

Bit 5	A-Flag Meaning
0	Do not trace Call/Endpoint Reference Values
1	Trace Call/Endpoint Reference Values

Bit 4	Ne-flag Meaning
0	Do not trace Ne-NSCs supporting the connection.
1	Trace Ne-NSCs supporting the connection.

Bit 3	Rp-flag Meaning
0	Do not trace Rp-NSCs supporting the connection.
1	Trace Rp-NSCs supporting the connection.

Bit 2	I-flag Meaning
0	Do not trace NSCs supporting the connection at the incoming interface of a node.
1	Trace Ne-NSCs (Rp-NSCs) supporting the connection at the incoming interface of a node if the Ne flag (Rp flag respectively) is set to “1”.

Bit 1	L-flag Meaning
0	Do not trace labels of Interworking LSPs supporting the call/connection.
1	Trace labels of Interworking LSPs supporting the call/connection.

Trace status (octet 6)

Bits 8 7 6 5 4 3 2 1	Meaning
0 0 0 0 0 0 0 0	trace in progress
0 0 0 0 0 0 0 1	trace completed normally
0 0 0 0 0 0 1 0	trace incomplete
0 0 0 0 0 0 1 1	trace has exceeded information element length limitations
0 0 0 0 0 1 0 0	trace has exceeded message length limitations

Note that a Trace status value of "trace in progress" is the only valid value in SETUP, ADD PARTY and CO-BI SETUP messages. Any other value shall be treated as an error (see section 4.3.1.2).

Trace Source Logical Port Identifier (octets 7.1 to 7.4)

The trace source logical port identifier uniquely identifies a logical port of the logical node at which the path or connection trace is initiated. The combination of the Trace Source Logical Port Identifier and the logical node identifier from the next Logical node / logical port entry in the Trace transit list information element unambiguously identifies a logical link. The logical port identifier is coded in binary, with a length of 4 octets. If the originating port is not a PNNI, the port ID that is added to the Trace transit list information element shall either be set to zero or shall specify the originating logical port. Section 5.3.4/PNNI 1.0 describes the abstract syntax and semantics of logical port identifiers.

VPI/VCI (octets 8.1 to 8.4)

Octets 8.1 and 8.2 contain the Virtual Path Identifier (not the VPCI) of the connection. The VPI value shall be coded in the low order 12 bits and the high order 4 bits (octet 8.1 bits 8 to 5) shall be coded to all zeros.
Octets 8.3 and 8.4 contain the Virtual Channel Identifier of the connection. When the connection being traced is a Virtual Path Connection (VPC), octets 8.3 and 8.4 shall be coded to all zeros.

The VPI/VCI value typically refers to the PNNI link indicated in the last octet group 12 that precedes this value, except in situations where gaps exist in the path or connection trace (refer to Section 3.2). The VPI value is the actual VPI (not the VPCI) used for the connection on the incoming side of the interface, with one exception: for the trace destination interface the VPI value is the actual VPI used for the connection on the outgoing side of the interface.

When the VPI/VCI value precedes all logical node/port identifiers in the Trace transit list information element, the values specify the VPI/VCI on the trace source interface, which may or may not be a PNNI interface. When the VPI/VCI value succeeds the last logical node/port identifier in the Trace transit list information element and the Trace status indicates “trace completed normally”, the value specifies the VPI/VCI on the trace destination interface, which may or may not be a PNNI interface.

DLCI value (octets 9.1 to 9.4)

Octets 9.1 to 9.4 contain the Data Link Connection Identifier assigned for the connection on the Frame Relay interface, coded as shown in Figure 10-18/X.76 in [2]. The default length of the DLCI value is two octets (10 bits). By bilateral agreements, some networks may support DLCI length of three or four octets.

Call Reference Flag (octets 10.1)

Bit	Call reference flag
8	Meaning
0	Call reference assigned by the node adding the call reference information to the Trace transit list information element
1	Call reference not assigned by the node adding the call reference information to the Trace transit list information element

Call Reference Value (octets 10.1 to 10.3)

Octets 10.1 (bits 1 to 7) to 10.3 contain a call reference value encoded as in the standard Call reference information element.

The call reference value typically refers to the PNNI link indicated in the last octet group 12 that precedes this value, except in situations where gaps exist in the path or connection trace (refer to Section 3.2). When the call reference value precedes all logical node/port identifiers in the Trace transit list information element, the value specifies the call reference on the trace source interface, which may or may not be a PNNI interface. When the call reference value succeeds the last logical node/port identifier in the Trace transit list information element and the Trace status indicates “trace completed normally”, the value specifies the call reference on the trace destination interface, which may or may not be a PNNI interface.

Endpoint Reference Flag (octet 11.1)

Bit 8	Endpoint reference flag Meaning
0	Endpoint reference assigned by the node adding the endpoint reference information to the Trace transit list information element
1	Endpoint reference not assigned by the node adding the endpoint reference information to the Trace transit list information element

Endpoint Reference Value (octets 11.1 to 11.2)

Octets 11.1 (bits 1 to 7) to 11.2 contain an endpoint reference value encoded as in the standard Endpoint reference information element.

The endpoint reference value refers to the same PNNI link as the call reference value that immediately precedes the endpoint reference.

Logical Node Identifier (octets 12.1 to 12.22)

The logical node identifier uniquely identifies a lowest level logical node that the connection is to transit or has transited. It is coded in binary, with a length of 22 octets. Section 5.3.3/PNNI 1.0 describes the abstract syntax and semantics of logical node identifiers.

Logical Port Identifier (octets 12.23 to 12.26)

The logical port identifier uniquely identifies a logical port of the logical node that the connection is to transit or has transited. The combination of Logical Node Identifier and Logical Port Identifier unambiguously identifies a logical link. The logical port identifier is coded in binary, with a length of 4 octets. If the port is not a PNNI interface, the port ID that is added to the Trace transit list information element shall either be set to zero or shall specify the next link over which the connection or party would be or has been progressed. Section 5.3.4/PNNI 1.0] describes the abstract syntax and semantics of logical port identifiers.

PNNI Trace Continuation Refusal indicator (octet 13)

Presence of this octet indicates that the node identified by the preceding logical node (octet group 12) has refused to participate in the trace.

Crankback Received at Trace Destination Node Indicator (octet 14)

Presence of this octet indicates that the node identified by the previous logical node entry (octet group 12) in the trace, when acting as trace destination node, received a crankback for this call from a non-PNNI interface.

PNNI Crankback Gap indicator (octet 15)

Presence of this octet indicates that the call has been subject to crankback, but that no Trace transit list information element was returned by the node initiating crankback.

Length of PNNI Crankback contents (octet 16.1)

The Length of PNNI Crankback contents includes the lengths of the PNNI Crankback Cause (octet 16.2), Blocked Transit Type (octet 16.3), and Blocked Transit Information (octets 16.4 etc.) fields. The length does not include the length of octet 16.1 itself.

PNNI Crankback Cause (octet 16.2)

Octet 16.2 is encoded as octet 7 in the Crankback information element, Section 6.4.6.3/PNNI 1.0, and uses the same crankback cause codepoints.

Blocked Transit Type (octet 16.3)

Bits 8 7 6 5 4 3 2 1	Meaning	Length of Blocked Transit Trace Information
0 0 0 0 0 0 1 0	Call or party has been blocked at the succeeding end of this interface	0
0 0 0 0 0 0 1 1	Blocked node	22
0 0 0 0 0 1 0 0	Blocked link	48

Blocked Transit Trace Information (octets 16.4 etc.)

The blocked transit trace information format depends on the blocked transit type. A node encodes identical blocked transit trace information in both the Trace transit list information element and the Crankback information element when a call clearing message with a crankback is being generated.

For Blocked transit type = “blocked node identifier”

Blocked node identifier	16.4 to 16.25

Blocked node identifier (octets 16.4 to 16.25)

Octets 16.4 to 16.25 are encoded as octets 6.1 to 6.22 of the Crankback information element, Section 6.4.6.3/PNNI 1.0, when the Blocked transit type is “blocked node identifier”. The Blocked node identifier identifies the logical node at which the connection or party has been blocked. It is coded in binary, with a length of 22 octets. Section 5.3.3/PNNI 1.0 describes the syntax of logical node identifiers.

For Blocked transit type = “blocked link identifier”

Blocked link’s preceding node identifier	16.4 to 16.25
Blocked link’s port identifier	16.26 to 16.29
Blocked link’s succeeding node identifier	16.30 to 16.51

Blocked link’s preceding node identifier (octets 16.4 to 16.25)

Octets 16.4 to 16.25 are encoded as octets 6.1 to 6.22 of the Crankback information element, Section 6.4.6.3/PNNI 1.0, when the Blocked transit type is “blocked link identifier”. The Blocked link’s preceding node identifier identifies the logical node preceding a link at which the connection or party has been blocked. It is coded in binary, with a length of 22 octets. Section 5.3.3/PNNI 1.0 describes the syntax of logical node identifiers.

Blocked link’s port identifier (octets 16.26 to 16.29)

Octets 16.26 to 16.29 are encoded as octets 6.23 to 6.26 of the Crankback information element, Section 6.4.6.3/PNNI 1.0, when the Blocked transit type is “blocked link identifier”. The Blocked link’s port identifier identifies the logical port of the blocked link’s preceding node identifier. The combination of the Blocked link’s preceding node identifier and the Blocked link’s port identifier unambigously identifies the link at which the connection or party has been blocked. The logical port identifier is coded in binary, with a length of 4 octets. Section 5.3.4/PNNI 1.0 describes the syntax of logical port identifiers.

Blocked link’s succeeding node identifier (octets 16.30 to 16.51)

Octets 16.30 to 16.51 are encoded as octets 6.27 to 6.48 of the Crankback information element, Section 6.4.6.3/PNNI 1.0, when the Blocked transit type is “blocked link identifier”. The Blocked link’s succeeding node identifier identifies the logical node succeeding a link at which the connection or party has been blocked. It is coded in binary, with a length of 22 octets. Section 5.3.3/PNNI 1.0 describes the syntax of logical node identifiers.

Length of Vendor specific contents (octet 17.1)

The length of vendor specific contents includes the lengths of the OUI (octets 17.2 to 17.4) and Vendor specific information (octets 17.5 etc.) fields. The length does not include the length of octet 17.1 itself.

Organizational Unique Identifier (OUI) (octets 17.2 to 17.4)

The Organizational Unique Identifier (OUI) coded in octets 17.2 to 17.4 unambiguously identifies an authority responsible for defining the contents of the vendor specific information.

Vendor Specific Information (octets 17.5 etc.)

Octets 17.5 etc. shall contain at most 8 octets of data. The syntax and semantics of the Vendor specific information (octets 17.5 etc) are beyond the scope of this specification.

Length of Incoming Ne-NSC List (octet 18.1)

The length of incoming Ne-NSC list contents in octets, i.e. excluding the octets used for the Incoming Ne-NSC List length and identifier.

Incoming Ne-NSC Identifier Value (octets 18.2 and 18.3)

The Ne-NSCs supporting the connection at the incoming interface of the node.

Length of Incoming Rp-NSC List (octet 19.1)

The length of the incoming Rp-NSC list contents in octets, i.e. excluding the octets used for the Incoming Rp-NSC List length and identifier.

Incoming Rp-NSC Identifier Value (octets 19.2 and 19.3)

The Rp-NSCs supporting the connection at the incoming interface of the node.
The minimum value of an Rp-NSC Identifier is 0. The Rp-NSC Identifier value 0 is referred to as Rp-NSC Bare and identifies “bare resources”.

Length of Outgoing Ne-NSC List (octet 20.1)

The length of the outgoing Ne-NSC list contents in octets, i.e. excluding the octets used for the Outgoing Ne-NSC List length and identifier.

Outgoing Ne-NSC Identifier Value (octets 20.2 and 20.3)

The Ne-NSCs supporting the connection at the outgoing interface of the node.

Length of Outgoing Rp-NSC List (octet 21.1)

The length of the outgoing Rp-NSC list contents in octets, i.e. excluding the octets used for the Outgoing Rp-NSC List length and identifier.

Outgoing Rp-NSC Identifier Value (octets 21.2 and 21.3)

The Rp-NSCs supporting the connection at the outgoing interface of the node.
The minimum value of an Rp-NSC Identifier is 0. The Rp-NSC Identifier value 0 is referred to as Rp-NSC Bare and identifies "bare resources".

Interworking LSP Labels (octets 21.2 and 21.3)

The labels used for packet transmissions across the interface. The terms "receive label" and "transmit label" are from the perspective of the node that added this octet group in the information element. The "receive label" corresponds to the label that is used to forward traffic in the same direction as the tracing message (either SETUP, ADD PARTY, or TRACE CONNECTION) over the interface. The transmit label refers to the label used for packets sent in the reverse direction to that of the receive label. When the labels precede all logical node/port identifiers in the Trace transit list information element, the values specify the labels used on the trace source interface. When the labels follow the last logical node/port identifiers in the Trace transit list information element and the Trace status indicates "trace completed normally", the values specify the labels used on the trace destination interface.

3.2 Rules for Interpreting the Trace Transit List Information Element

[INFORMATIVE]

The following rules apply for interpretation of the final Trace transit list information element returned to the trace source node:

- If the Trace status field is set to “trace in progress”, this is to be interpreted as if the value were set to “trace incomplete”.
- The presence of a Trace Continuation refusal indicator reveals that the node identified by the previous logical node entry (octet group 12) refused continuation of the trace.
- If any combination of call reference, endpoint reference, ~~and~~ either VPI/VCI or DLCI values, incoming Ne-NSC list, incoming Rp-NSC list, and interworking LSP labels appear before the first logical node / logical port entry, they represent the values on the trace source interface where the trace was initiated.
- If the Pass along request bit is set to “pass along request”, there may be gaps between two logical node / logical port entries. These gaps are not identified in the Trace transit list information element. The location of gaps can be discerned by network management only by using its knowledge of the network topology. The first of the two logical node / logical port entries identifies the link before the gap. The last link in the gap, used to reach the node identified in the next logical node / logical port entry, is not identified in the Trace transit list information element.
- Except when there is a gap in the Trace transit list information element, when VPI/VCI or call/endpoint reference values appear between two logical node / logical port entries, they represent the values on the link between the logical nodes specified in the logical node / logical port entries. The link is identified in the first logical node / logical port entry. The VPI value represents the actual VPI used for the connection on the succeeding side of the interface.
- When there is a gap in the Trace transit list information element and a VPI/VCI~~or~~, call/endpoint reference entry, incoming Ne-NSC list, incoming Rp-NSC list, or interworking LSP labels appears between the two logical node / logical port entries, they represent the values used on the last link in the gap. The VPI value represents the actual VPI used for the connection on the interface where the tracing message (SETUP, ADD PARTY, or TRACE CONNECTION) was received at the second logical node.~~succeeding side of the interface~~.
- If the Trace status is set to “trace completed normally” and any combination of outgoing Ne-NSC list, outgoing Rp-NSC list, interworking LSP labels, call reference, endpoint reference, and either VPI/VCI or DLCI values appear after the last logical node / logical port entry, they represent the values on the trace destination interface on the trace destination node. The VPI value is the actual VPI used for the connection on the outgoing side of the destination interface.
- Presence of PNNI crankback information (octet group 16) indicates that the node identified by the previous logical node / logical port entry initiated call clearing with crankback, with one exception, as identified in the next bullet item. VPI/VCI values may or may not be present immediately before the previous logical node / logical port entry, even if the V flag is set. Call/endpoint reference values may or may not be present immediately before the previous logical node / logical port entry, even if the A flag is set. The node may have initiated call clearing before incorporating these fields into the Trace transit list information element.
- When the PNNI crankback information (octet group 16) immediately follows a PNNI Crankback Gap indicator, this indicates that the crankback information was added by the node identified by the previous logical node / logical port entry (octet group 12) in the Trace transit list information element. This node is not the node at which the call was blocked. The crankback information is the information received by this node in the Crankback information element from a node which did not return a Trace transit list information element and that most likely did not support this feature.

- A logical node / logical port entry following the PNNI crankback information (octet group 16) indicates a node that attempted alternate routing.
- Interpretation of the blocked transit information is performed as follows:
  - "call or party has been blocked at the succeeding end of this interface" – the logical node / logical port entry immediately before the PNNI crankback information (octet group 16) identifies the node that initiated call clearing with crankback, at the succeeding side of the blocked interface. The logical node / logical port entry one before that indicates the node at the preceding side of the blocked interface and the corresponding port ID for the blocked interface.
  - "blocked node" – the blocked node is identified by the Blocked node identifier encoded as part of the Blocked transit identifier in the Crankback data in the Trace transit list information element.
  - "blocked link" – the blocked link (or adjacency) is identified by the Blocked link's preceding node identifier, the Blocked link's port identifier , and the Blocked link's succeeding node identifier encoded as part of the Blocked transit identifier in the Crankback data in the Trace transit list information.
- The "Crankback received at trace destination node" indicator indicates that the trace destination node, which is identified by the previous logical node / logical port entry, received a call clearing message including crankback indication from the trace destination interface, which is a non-PNNI interface (e.g. AINI). One such indicator is present for each crankback received by the trace destination node.
- When one or more vendor specific information fields occur immediately after the Trace status field, they were added by the trace source node.
- When one or more vendor specific information fields occur after a logical node / logical port entry, they were added by the logical node identified by that logical node / logical port entry.
- When one or more vendor specific information fields occur after a crankback field, they were added by the logical node identified by the logical node / logical port entry previous to the crankback field.

4 Path Trace

4.1 Path Trace Description

[INFORMATIVE]

Path tracing is used to determine the logical nodes and logical ports traversed during connection establishment. Path tracing is supported for both point-to-point and point-to-multipoint connections. Point-to-point path tracing is used to trace the path of a proposed point-to-point connection from a given trace source node towards a called party number. Point-to-multipoint path tracing is used to trace the path of a branch from a given trace source node on a new or existing point-to-multipoint connection towards a new party.

Path tracing can be initiated in two ways:

1. Test connections or parties in support of path tracing can be initiated at the trace source node by a network management action towards a called party numb er, or

2. an administrative action can cause connections and parties to be traced according to a network specific policy (e.g. all new connections and parties at a given UNI, or all new Soft PVCs originated at the trace source node). Such a policy can be defined using the atmTraceFilterTable defined in Annex C.

In the first case (i.e. test connections or parties), the trace typically indicates that the connection or party used to perform the trace should be cleared immediately upon reaching the trace destination node. However, if a new point-to-multipoint test connection with multiple branches is desired, the branches must be left up until all the branches have been added. If the test connection is to be left up and the VPI/VCI can be assigned at the other side of the interface, there may be some possibility of VPI/VCI collision for SETUP messages received on this interface, when the other side of the interface does not know that this VPI/VCI value is being used.

In the second cas e, the connection or party is typically left up.

When a connection or party is to be immediately cleared, it is terminated by a trace destination node within the PNNI network. When the connection (or party) is to be left up, the trace aspect of the connection (or party) is terminated by the trace destination node, but the connection (or party) itself is progressed to the actual destination node.

Given the connection or party establishment information, the trace source node computes a path, inserts the corresponding DTLs in a SETUP or ADD PARTY message in the usual fashion and adds its trace information including its node ID and outgoing interface port ID to the Trace transit list information element. The trace information may also include crankback information, VPI/VCI and DLCI values, call reference values, ~~and~~ endpoint reference values (for parties), incoming Ne-NSC and Rp-NSC lists, outgoing Ne-NSC and Rp-NSC lists, and interworking LSP labels, if indicated by the flags in the Flag octet of the Trace transit list information element. It then sends the message on the outgoing interface.

A node other than the trace source node or trace destination node that is traversed by this message performs the procedures normally followed for a SETUP or ADD PARTY message, fills in the trace information and forwards the message to the next node.

At the trace destination node, the trace information is kept to be transferred into the corresponding CONNECT, ADD PARTY ACKNOWLEDGE, RELEASE[*], RELEASE COMPLETE[*] , DROP PARTY[*] or ADD PARTY REJECT[*] message if the connection or party is intended to be left up, or into a RELEASE, RELEASE COMPLETE or ADD PARTY REJECT message if the connection or party is to be immediately cleared.

The following figures illustrate sample message flows for the Path Trace feature.

[*] These messages occur for reasons due to normal call processing, not because of the trace feat ure. For example, call establishment may fail further downstream, beyond the trace destination node.

Figure 4-1 depicts an example message sequence for the case where a point-to-point SVC call between two ATM UNIs is being traced.

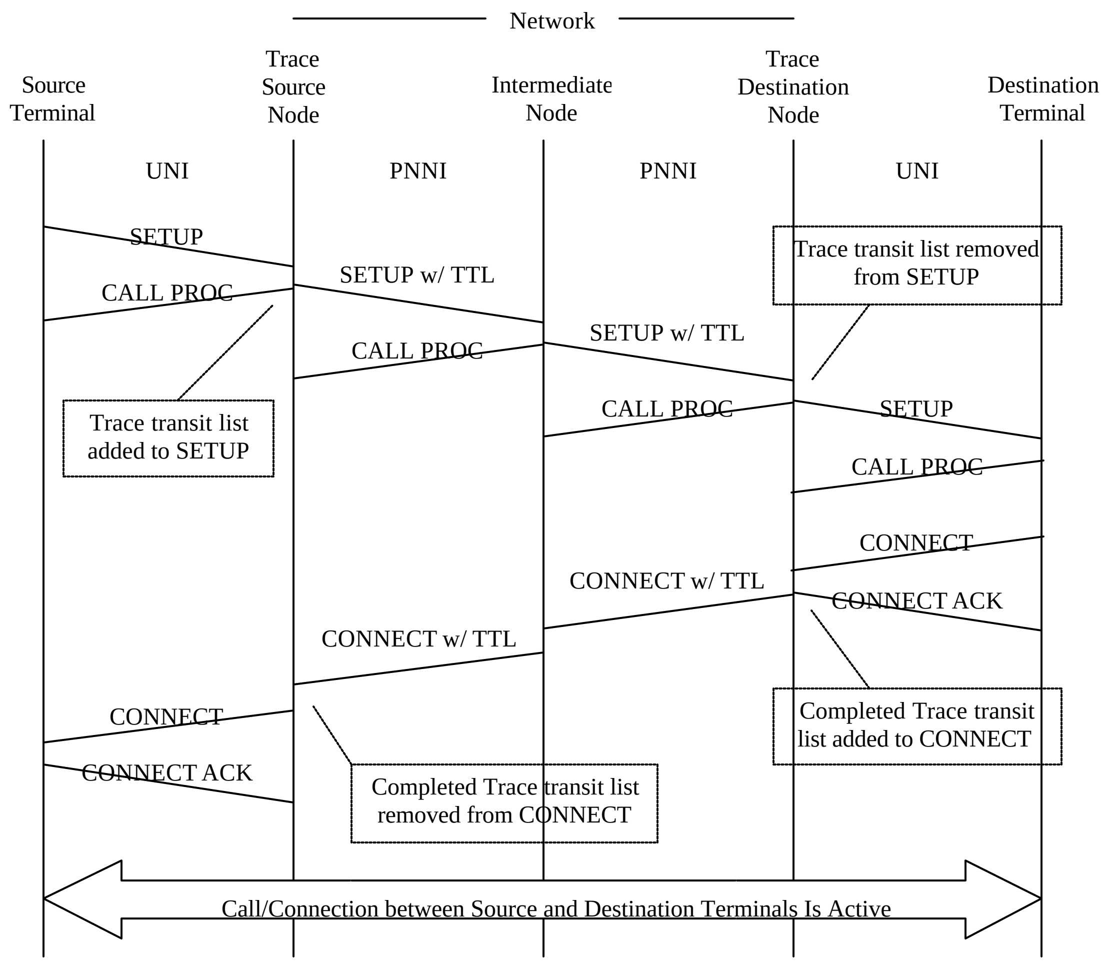


Figure 4-1 Sample Message Sequence for Path Trace for a user-initiated point-to-point SVC between ATM endpoints (without Call Clearing at the Trace Destination Node)

The steps in tracing the connection are as follows:

1. The trace source node receives a SETUP message over a UNI. Since it has been configured to trace connections from this UNI, the trace source node adds a Trace transit list information element to the SETUP message. The flags in the Trace transit list information element are set based on configuration in the trace source node, and the Trace status field is set to "trace in progress". The trace source node then adds the following information to the Trace transit list information element:
   a) A port ID identifying the trace source UNI interface.
   b) If the V flag is set, the VPI/VCI from the trace sourceUNI interface.
   c) If the A flag is set, the call reference value from the trace source UNI interface.
   d) If the Ne flag is set and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface. The setting of the I flag does not influence the inclusion of the incoming interface report at the trace source node.
   e) If the Rp flag is set and the connection is supported by a specific resource partition of the incoming interface, the list of Rp-NSCs supporting the connection at the incoming interface. The setting of the I flag does not influence the inclusion of the incoming interface report at the trace source node.

f) If the L flag is set and the incoming interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs used for the connection,
g) Its own node ID and the outgoing port ID.
h) If the Ne flag is set and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.
i) If the Rp flag is set and the connection is supported by a specific resource partition of the outgoing interface, the list of Rp-NSCs supporting the connection at the outgoing interface,

The trace source node then ~~adds its own node ID and its outgoing port ID and~~ forwards the SETUP message to the succeeding node.

2. The intermediate node receives the SETUP message. Since the Trace transit list information element is present, its adds the following information:
a) If the V flag is set, the VPI/VCI value on the incoming interface. These are added by the succeeding side in order to facilitate tracing over links which are non-assigning on the preceding side.
b) If the A flag is set, the call reference value on the incoming interface.
c) If both the I and Ne flags are set and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface.
d) If both the I and Rp flags are set and the connection is supported by a specific resource partition of the incoming interface, the list of Rp-NSCs supporting the connection at the incoming interface,
e) If the L flag is set and the incoming interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs used for the connection,
f) Its own node ID and the outgoing port ID.
g) If the Ne flag is set and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.
h) If the Rp flag is set and the connection is supported by a specific resource partition of the outgoing interface, the list of Rp-NSCs supporting the connection at the outgoing interface.

The intermediate node then ~~adds its own node ID and the outgoing port ID and~~ forwards the SETUP message to the succeeding node.

3. The trace destination node adds information about the incoming link (VPI/VCI, call reference, incoming Ne-NSCs, incoming Rp-NSCs, interworking labels) to the Trace transit list information element in the same manner as intermediate nodes. It then adds its own node ID and outgoing port ID. Information about the destination UNI interface is also added as follows:
a) If the Ne flag is set and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.
b) If the Rp flag is set and the connection is supported by a specific resource partition of the outgoing interface, the list of Rp-NSCs supporting the connection at the outgoing interface.
~~a~~c) If the V flag is set, the VPI/VCI used on the destination UNI.
~~b~~d) If the A flag is set, the call reference used on the destination UNI.
e) If the L flag is set and the outgoing interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs used for the connection,

Since the X flag is not set, the trace destination node removes and saves the Trace transit list information element and forwards the SETUP message out the destination UNI without the Trace transit list information element.

4. Upon receiving the CONNECT message, the trace destination node sets the Trace status field in the saved Trace transit list information element to “trace completed normally”, places the updated Trace transit list information element into the CONNECT message, and progresses it towards the calling party.

5. Intermediate nodes progress the CONNECT message normally without any additional actions.

6. At the trace source node, the Trace transit list information element is removed from the CONNECT message and saved for retrieval by network management. The CONNECT message is then progressed out the originating UNI to the source terminal without the Trace transit list information element.

Figure 4-2 depicts an example message sequence for the case where an SPVC between two Frame Relay endpoints is being traced.

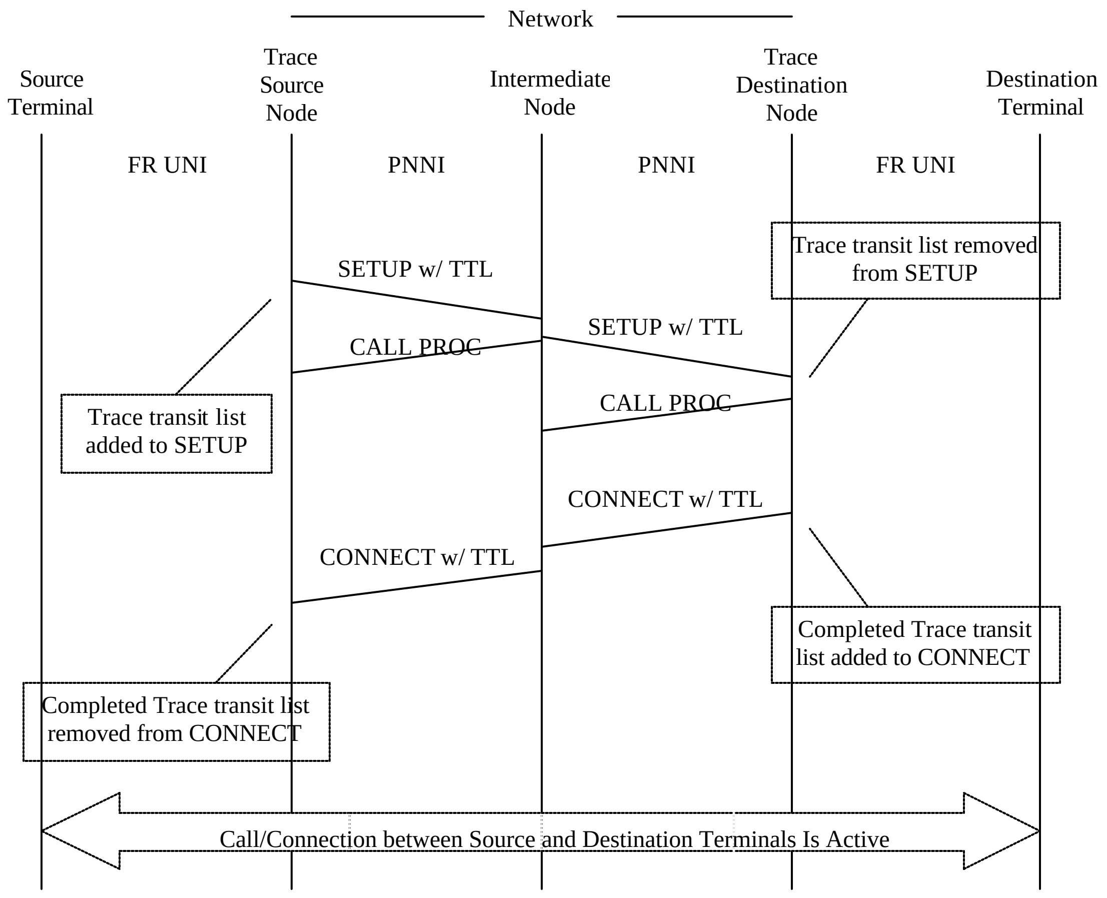

Network
Source Terminal
Trace Source Node
Intermediate Node
Trace Destination Node
Destination Terminal
FR UNI
PNNI
PNNI
FR UNI
Trace transit list removed from SETUP
SETUP w/ TTL
CALL PROC
SETUP w/ TTL
Trace transit list added to SETUP
CALL PROC
CONNECT w/ TTL
CONNECT w/ TTL
Completed Trace transit list added to CONNECT
Completed Trace transit list removed from CONNECT
Call/Connection between Source and Destination Terminals Is Active

Figure 4-2 Sample Message Sequence for Path Trace for network initiated connections (Soft PVCs) between Frame Relay endpoints (without Call Clearing at the Trace Destination Node)

The steps in tracing the connection are as follows:

1. The trace source node adds a Trace transit list information element to a SETUP message. The flags in the Trace transit list information element are set based on configuration in the trace source node and the Trace status field is set to "trace in progress". The trace source node then adds the following information to the Trace transit list information element:
   a) A port ID identifying the trace source FR UNI interface.
   b) If the V flag is set, since the trace source interface is a Frame Relay interface, the DLCI from the trace source interface.
   c) If the Ne flag is set and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface. The setting of the I flag does not influence the inclusion of the incoming interface report at the trace source node.
   d) If the Rp flag is set and the connection is supported by a specific resource partition of the incoming interface, the list of Rp-NSCs supporting the connection at the incoming interface . The setting of the I flag does not influence the inclusion of the incoming interface report at the trace source node.
   e) Its own node ID and the outgoing port ID.
   f) If the Ne flag is set and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.

g) If the Rp flag is set and the connection is supported by a specific resource partition of the outgoing interface, the list of Rp-NSCs supporting the connection at the outgoing interface .

The trace source node then ~~adds its own node ID and its outgoing port ID and~~ forwards the SETUP message to the succeeding node.

2. Actions taken by the intermediate node are as specified in step 2 for Figure 4-1.

3. The trace destination node adds information about the incoming link (VPI/VCI, call reference, incoming Ne-NSCs, incoming Rp-NSCs, interworking labels) to the Trace transit list information element in the same manner as intermediate nodes. It then adds its own node ID and outgoing port ID. Information about the destination FR UNI interface is also added as follows:
a) If the Ne flag is set and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.
b) If the Rp flag is set and the connection is supported by a specific resource partition of the outgoing interface, the list of Rp-NSCs supporting the connection at the outgoing interface.
~~a~~c) If the V flag is set, since the destination interface is a Frame Relay interface, the DLCI from the destination interface.

   The trace destination node sets the Trace status field to "trace completed normally", places the updated Trace transit list information element into the CONNECT message and progresses it towards the calling party.

4. Intermediate nodes progress the CONNECT message normally without any additional actions.

5. At the trace source node, the Trace transit list information element is removed from the CONNECT message and saved for retrieval by network management.

Figure 4-3 demonstrates a point-to-point network management initiated test connection. The intent is to have the connection progress through the network towards the called party and to then be released at the trace destination node before reaching the called party.

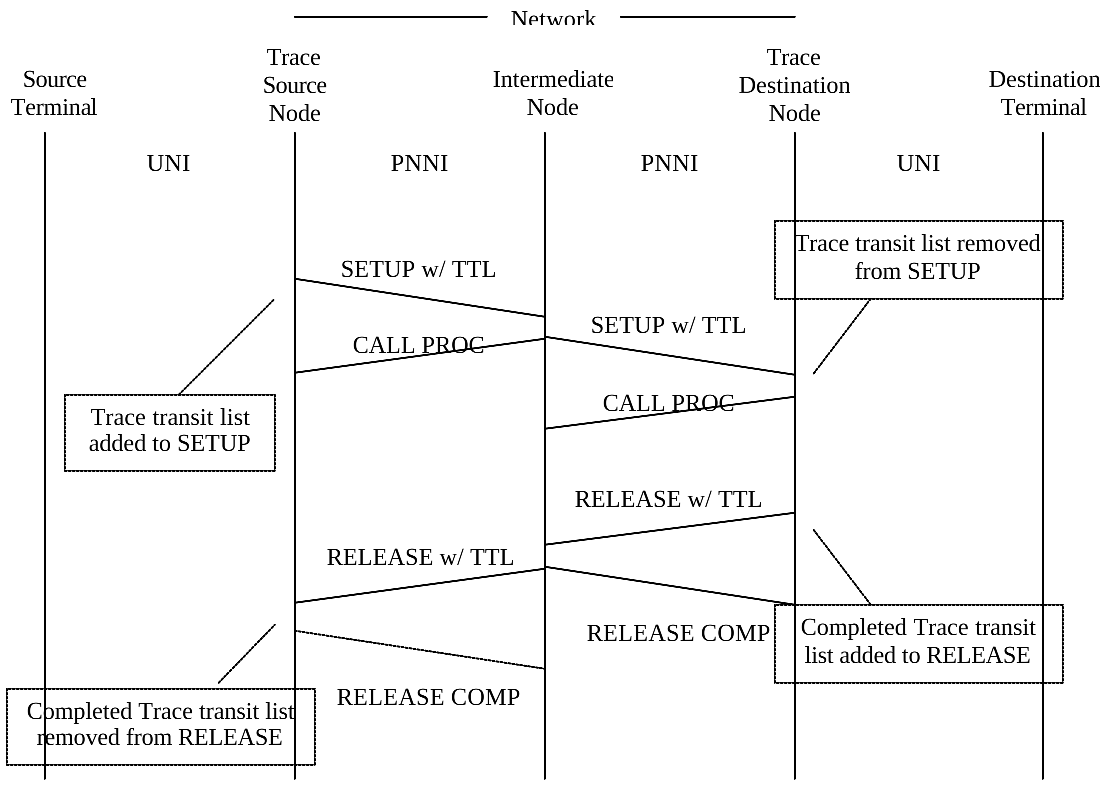


Figure 4-3 Sample Message Sequence for Path Trace wi th Call Clearing at the Trace Destination Node

The steps in tracing the connection are as follows:

1. Network management instructs the trace source node to launch a test connection. This request contains the called party number, all necessary traffic parameters and the trace flags.
2. The trace source node creates a SETUP message from the parameters received from network management. It includes a Trace transit list information element with the appropriate trace flags. The trace source node then adds the following information to the Trace transit list information element:
   a) A port ID identifying the trace source interface specified by network management.
   b) Its own node ID and outgoing port ID.
   The SETUP message is then progressed to the succeeding node.
3. The intermediate node receives the SETUP message and adds its own node ID and the outgoing port ID and forwards the SETUP message to the succeeding node.
4. The trace destination node adds its own node ID and outgoing port ID.

   Since the X flag is set, the trace destination node sets the Trace status field to “trace completed normally”, creates a RELEASE message, adds the Trace transit list information element to the RELEASE message and sends it towards the calling party.
5. Intermediate nodes progress the RELEASE message normally without any additional actions.
6. At the trace source node, the Trace transit list information element is removed from the RELEASE message and saved for retrieval by network management. The RELEASE message is not progressed further by the trace source node.

4.2 Additions to PNNI Signalling Messages

[NORMATIVE]

4.2.1 CONNECT

Figure 6-5/PNNI 1.0 CONNECT Message Contents is augmented with the following:

Information Element	Reference	Type	Length
Trace transit list	3.1	O(1)	38-1466 (2)

Note 1 - Included when using path trace functionality.

Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to "1"; flags Ne, Rp, I, and L set to "0'; no Vendor specific information; and 6 crankbacks with blocked transit type coded as "blocked link identifier"; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to "1" and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-4 Additional CONNECT message content

4.2.2 RELEASE

Figure 6-6/PNNI 1.0 RELEASE message content is augmented with the following:

Information Element	Reference	Type	Length
Trace transit list	3.1	O(1)	38-1466(2)

Note 1 - Included when using path trace functionality.

Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to "1"; flags Ne, Rp, I, and L set to "0'; no Vendor specific information; and 6 crankbacks with blocked transit type coded as "blocked link identifier"; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to "1" and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-5 Additional RELEASE message content

4.2.3 RELEASE COMPLETE

Figure 6-7/PNNI 1.0 RELEASE COMPLETE message content is augmented with the following:

Information Element	Reference	Type	Length

Trace transit list	3.1	O(1)	38-1466(2)

Note 1 - Included when using path trace functionality.

Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to “1”; flags Ne, Rp, I, and L set to “0’; no Vendor specific information; and 6 crankbacks with blocked transit type coded as “blocked link identifier”; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to “1” and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-6 Additional RELEASE COMPLETE message content

4.2.4 SETUP

Figure 6-8/PNNI 1.0 SETUP message content is augmented with the following:

Information Element	Reference	Type	Length
Trace transit list	3.1	O(1)	38-1466(2)

Note 1 - Included when using path trace functionality.

Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to “1”; flags Ne, Rp, I, and L set to “0’; no Vendor specific information; and 6 crankbacks with blocked transit type coded as “blocked link identifier”; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to “1” and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-7 Additional SETUP message content

4.2.5 ADD PARTY

Figure 6-19/PNNI 1.0 ADD PARTY message content is augmented with the following:

Information Element	Reference	Type	Length
Trace transit list	3.1	O(1)	38-1466(2)

Note 1 - Included when using path trace functionality.

Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to “1”; flags Ne, Rp, I, and L set to “0’; no Vendor specific information; and 6 crankbacks with blocked transit type coded as “blocked link identifier”; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to “1” and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-8 Additional ADD PARTY message content

4.2.6 ADD PARTY ACKNOWLEDGE

Figure 6-20/PNNI 1.0 ADD PARTY ACKNOWLEDGE message content is augmented with the following:

Information Element	Reference	Type	Length
Trace transit list	3.1	O(1)	38-1466(2)

Note 1 - Included when using path trace functionality.
Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to “1”; flags Ne, Rp, I, and L set to “0”; no Vendor specific information; and 6 crankbacks with blocked transit type coded as “blocked link identifier”; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to “1” and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-9 Additional ADD PARTY ACKNOWLEDGE message content

4.2.7 ADD PARTY REJECT

Figure 6-22/PNNI 1.0 ADD PARTY REJECT message content is augmented with the following:

Information Element	Reference	Type	Length
Trace transit list	3.1	O(1)	38-1466(2)

Note 1 - Included when using path trace functionality.
Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to “1”; flags Ne, Rp, I, and L set to “0”; no Vendor specific information; and 6 crankbacks with blocked transit type coded as “blocked link identifier”; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to “1” and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-10 Additional ADD PARTY REJECT message content

4.2.8 DROP PARTY

Figure 6-23/PNNI 1.0 DROP PARTY message content is augmented with the following:

Information Element	Reference	Type	Length
Trace transit list	3.1	O(1)	38-1466(2)

Note 1 - Included when using path trace functionality.

Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to "1"; flags Ne, Rp, I, and L set to "0"; no Vendor specific information; and 6 crankbacks with blocked transit type coded as "blocked link identifier"; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to "1" and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-11 Additional DROP PARTY message content

4.2.9 CO-BI SETUP

Figure 26-2/GSS 1.0 [3] CO-BI SETUP message content is augmented with the following:

Information Element	Reference	Type	Length
Trace transit list	3.1	O(1)	38-1466(2)

Note 1 - Included when using path trace functionality.

Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network. 1466 octets correspond to the length of a Trace transit list information element with 25 nodes; flags A, V, and C set to "1"; flags Ne, Rp, I, and L set to "0"; no Vendor specific information; and 6 crankbacks with blocked transit type coded as "blocked link identifier"; for a point-to-multipoint call. In this scenario, if NSCs were to be recorded by setting flags Ne, Rp, and I to "1" and the average number of Ne-NSCs supporting the connection at an interface is two and the average number of Rp-NSCs supporting the connection at an interface is four, then the maximum length required would need to be increased to 2373 octets.

Figure 4-12 Additional CO-BI SETUP message content

4.3 Path Trace Procedures

[NORMATIVE]

The procedures defined in the ATM Forum PNNI Specification apply for connection and party establishment with path trace functionality. Only the additional procedures required for path trace functionality are defined in this specification.

4.3.1 Processing SETUP, CO-BI SETUP and ADD PARTY Messages

In the following sections, the term "SETUP message" is understood to apply to both SETUP and CO-BI SETUP messages except in cases where the V flag is discussed, since the V flag is treated as a spare bit for CO-BI SETUP messages.

4.3.1.1 Initiating Path Trace

The trace source node initiates the path trace by inserting a Trace transit list information element in the SETUP or ADD PARTY message before progressing the message towards the called party. The trace transit list records the path traversed by the SETUP or ADD PARTY message from the trace source node to the trace destination node. The trace source node controls what path information is recorded by setting the C (crankback), V (VPI/VCI), ~~and~~ A (call/endpoint reference), Ne (Ne-NSC lists), Rp (Rp-NSC lists), I (Incoming NSCs), and L (interworking LSP labels) flags in the Trace transit list information element. If the C flag is set to 0, only information (node IDs and port IDs) of the final path, i.e., not including any failed paths, is recorded. If the C flag is set to 0 and the connection or party fails to reach the trace destination node, no path trace information will be returned. If the C flag is set to 1, all attempted paths including the final one are recorded (regardless of whether the call succeeds or fails). If the V flag is set to 1, the VPI/VCI values used on each link are recorded. In the case of Frame Relay endpoints, the DLCI values of those endpoints are also recorded. If the A flag is set to 1, the call reference values used on each link, and endpoint reference values for point-to-multipoint connections, are recorded. If the Ne flag is set to 1, the Ne-NSCs supporting the connection are recorded. If the Rp flag is set to 1, the Rp-NSCs supporting the connection are recorded. If the I flag is set to 1, the NSCs supporting the connections at the incoming interface of the nodes are recorded. If the L flag is set to 1 and the interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs are recorded.

By setting the X flag to 1, the trace source node indicates to the trace destination node that the connection or party shall not be progressed further and that it shall be cleared rather than connected. If the X flag is set to 1, the trace source node shall not set the V flag or the A flag to 1.

After adding the Flags field to the Trace transit list information element, the trace source node shall add the Trace status field with the value set to "trace in progress". Trace information shall then be added to the Trace transit list information element by following the procedures specified in Section 4.3.1.2.

If a Trace transit list information element is present in the received setup or add party indication, this node shall continue the received trace (following the procedures of Section 4.3.1.2), ignoring any network specific policy that triggers a new path trace (i.e. this node shall not act as a trace source node).

4.3.1.2 Processing a Trace Transit List

The following procedures shall apply

- at the trace source node,
- whenever a SETUP or ADD PARTY message is received with a Trace transit list information element, and
- when a node initiates alternate routing in response to crankback, though not necessarily all of these procedures apply (see Section 0).

If the Trace status field is not set to "trace in progress", the Trace transit list information element shall be saved as the Saved Original Trace transit list and the Saved Modified Trace transit list and removed from the outgoing SETUP or ADD PARTY message. In addition:

- If the X flag is set to 1, call clearing shall be initiated using normal PNNI procedures. The cause code in the Cause information element shall be set to cause #31 "normal, unspecified." In addition, the Trace transit list information element from the SETUP or ADD PARTY message, shall be copied into the RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message before progressing the message towards the trace source node.
- Otherwise, the remaining procedures in this section shall be ignored.

The following rules, in this order, shall be followed to determine what entries shall be added to the Trace transit list information element by each node:

1. If this node is the trace source node, the Trace source port identifier field with the value either set to zero (if the trace source interface is not a PNNI interface and no logical port ID is assigned to this interface) or identifying the trace source interface from which the path trace is initiated;
2. If the V flag is set to 1 and
   - this node is the trace source node and the trace source interface is an ATM interface, a VPI/VCI entry for the trace source interface (note that what is traced is the actual VPI used for the connection on this side of the trace source interface, not the VPCI),
   - this node is the trace source node, the trace source interface is a Frame Relay interface and a 1:1 mapping is used between the Frame Relay and the ATM connection, a DLCI entry for the trace source interface,
   - this node is the trace source node, and the trace source interface is neither an ATM interface, nor a Frame Relay interface (e.g. Circuit Emulation Service), a VPI/VCI entry may be added for the trace source interface,
   - this node is not the trace source node, a VPI/VCI entry for the incoming interface on which the SETUP or ADD PARTY message was received (note that what is traced is the actual VPI used for the connection on this side of the incoming interface, not the VPCI);
3. If the A flag is set to 1 and
   - this node is the trace source node, the connection leg on the trace source interface is switched, and the trace source interface supports a call reference that can be specified in 3 octets, a call reference entry for the trace source interface including the endpoint reference field for point-to-multipoint connections,
   - this node is not the trace source node, a call reference entry for the incoming interface on which the SETUP or ADD PARTY message was received, including the endpoint reference field for point-to-multipoint connections;
4. If the Ne flag is set to 1, this node does not refuse to participate in path traces received over the incoming interface,
   - this node is the trace source node and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface.
   - this node is not the trace source node, the I flag is set to 1, and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface.
5. If the Rp flag is set to 1, this node does not refuse to participate in path traces received over the incoming interface,
   - this node is the trace source node and
     - the resources supporting the connection at the incoming interface are tagged by Rp-NSCs, the list of Rp-NSCs supporting the connection at the incoming interface.

    - the resources supporting the connection at the incoming interface are not tagged by any Rp-NSCs, a single Incoming Rp-NSC value set to "Rp-NSC Bare".
    - the incoming interface is tagged by all Rp-NSCs, a single Incoming Rp-NSC value set to 0xFFFF.
- this node is not the trace source node, the I flag is set to 1 and
    - the resources supporting the connection at the incoming interface are tagged by Rp-NSCs, the list of Rp-NSCs supporting the connection at the incoming interface.
    - the resources supporting the connection at the incoming interface are not tagged by any Rp-NSCs, a single Incoming Rp-NSC value set to "Rp-NSC Bare".
    - the incoming interface is tagged by all Rp-NSCs, a single Incoming Rp-NSC value set to 0xFFFF.

6. If the L flag is set to 1, this node does not refuse to participate in path traces received over the incoming interface, and the incoming interface on which the SETUP or ADD PARTY message was received is an ATM-MPLS network interworking interface, the labels of the interworking LSPs to be used for the connection.

~~4.~~7. The node shall save the current Trace Transit List information element locally as the Saved Original Trace transit list

~~5.~~8. If this node refuses to participate in path traces received over the incoming interface, a logical node / logical port entry including this node's lowest-level logical node identifier. The logical port identifier shall be set to zero. In addition, the Trace Continuation refusal indicator shall be added to the Trace transit list information elementand

- If the X flag is set to 0, the Trace transit list information element shall be saved locally as the Saved Modified Trace transit list and removed from the SETUP or ADD PARTY message before progressing the message towards the called party. The remaining procedures in this section shall be ignored.
- If the X flag is set to 1, call clearing shall be initiated using normal PNNI procedures. The cause code in the Cause information element shall be set to cause #31 "normal, unspecified." In addition, the Trace transit list information element from the SETUP or ADD PARTY message, as modified by the procedures above, shall be copied into the RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message before progressing the message towards the trace source node and the Trace status shall be set to "trace incomplete";

~~6.~~9. One logical node / logical port entry where the logical node ID specifies the lowest level logical node and

- if this node is not the trace destination node, the logical port specifies the outgoing interface on which the SETUP or ADD PARTY message will be progressed,
- if this node is the trace destination node, the logical port is either set to zero (if the next link is not a PNNI interface and no logical port ID is assigned to this interface) or specifies the next link that would be traversed by the connection or party;

10. If the Ne flag is set to 1 and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.

11. If the Rp flag is set to 1 and
    - the resources supporting the connection at the outgoing interface are tagged by Rp-NSCs, the list of Rp-NSCs supporting the connection at the outgoing interface.
    - the resources supporting the connection at the outgoing interface are not tagged by any Rp-NSCs, a single Outgoing Rp-NSC value set to "Rp-NSC Bare".

- the outgoing interface is tagged by all Rp-NSCs, a single Outgoing Rp-NSC value set to 0xFFFF.

~~7.~~12. If this node is the trace destination node and the V flag is set to 1:

- If the trace destination interface is an ATM interface, a VPI/VCI entry for the trace destination interface;
- If the trace destination interface is a Frame Relay interface and a 1:1 mapping is used between the Frame Relay and the ATM connection, a DLCI entry for the trace destination interface;
- If the trace destination interface is not an ATM interface or a Frame Relay interface (e.g. Circuit Emulation Service), a VPI/VCI entry may be added for the trace destination interface.

13. If this node is the trace destination node, the L flag is set to 1, and the trace destination interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs to be used for the connection.

~~8.~~14. If this node is the trace destination node, the A flag is set to 1, the connection leg on the trace destination interface is switched, and the trace destination interface supports a call reference that can be specified in 3 octets, a call reference entry for the trace destination interface including the endpoint reference for point-to-multipoint connections;

The numbered procedures above may be interrupted at any point due to error, failure, or other conditions (due to standard PNNI call processing) that result in call or party clearing, with or without crankback. In this case the procedures specified in Section 4.3.3 shall be followed. Before the procedures of that section are applied, if the Trace transit list information element has not yet been saved above as the Saved Original Trace transit list, the current Trace information element shall be saved as the Saved Original Trace transit list. In addition, if the Trace transit list information element has not yet been saved above as the Saved Modified Trace transit list, the current Trace transit list information element shall be saved as the Saved Modified Trace transit list.

The procedures described in Section 4.3.7 shall be followed to check for Trace transit list information element length exceeded errors and message length exceeded errors.

The Trace transit list information element resulting from the above procedures shall be saved locally as the Saved Modified Trace transit list.

If this is not the trace destination node, the SETUP or ADD PARTY message shall be progressed with the updated Trace transit list information element. Note that in the case where this node refuses to participate in the trace, the Trace transit list information element will no longer be present in the SETUP or ADD PARTY message (see step 5 above).

If this is the trace destination node, then

- If the X flag is set to 0, the Trace transit list information element shall be removed before progressing the SETUP or ADD PARTY message towards the called party user.
- Otherwise, (the X flag is set to 1) the procedures in Section 4.3.2 apply.

4.3.2 Path Trace Clearing At The Trace Destination Node

The following steps shall be performed at the trace destination node when the X flag is set to "1" and after the node applied the procedures of section 4.3.1.2:

- While performing the procedures of section 4.3.1.2,the trace destination node shall verify that it has reachability to the next transit identified in the DTL stack (if this is not the DTL Terminator) or to the called party number or specified transit network (if this is the DTL Terminator). If the trace destination node does not have reachability, or if, according to local information (including applicable policies), it does not have sufficient resources to accept the connection or party on either the incoming interface or on any interface over which the connection or party may be progressed, call clearing procedures shall be followed as specified in Section 4.3.3, and the remaining procedures in this section shall be ignored.

- The Saved Modified Trace transit list shall be copied to a RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message. If the Trace status in the message is set to “trace in progress”, it shall be set to “trace completed normally”.
- The cause code in the Cause information element shall be set to cause #31 “*normal, unspecified*.”
- Standard PNNI procedures for call clearing shall apply, causing the RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message to be sent towards the trace source node.

4.3.3 Procedures at a Node Initiating Call Clearing

The following procedures shall apply when error, failure, or other conditions (due to standard PNNI call processing) are encountered that result in call or party clearing, with or without crankback, with the following exception: These procedures are not always applicable when the protocol errors described in Sections 6.5.6/PNNI 1.0 and 6.5.7/PNNI 1.0 lead to call clearing.

Note that the procedures described below can be invoked as a result of processing either a received SETUP or ADD PARTY message, or as a result of receiving a call clearing message where alternate routing was attempted but failed (e.g. an entry border node propagating a call clearing), or as a result of receiving a call clearing message where a “link blocked” crankback is generated from a "Call or party has been blocked at the succeeding end of this interface" crankback.. As such, the "current message" referenced below could be any of SETUP, ADD PARTY, RELEASE, RELEASE COMPLETE, ADD PARTY REJECT. If the C flag in the Saved Original Trace transit list information element is set to 0, then normal call or party clearing and crankback procedures shall be followed. The Trace transit list information element shall not be included in the RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message generated by this node. If this is the trace source node, then the procedures of Section 4.3.4.4 shall apply, otherwise no further processing of the Trace transit list information element shall be performed.

If the C flag in the Saved Original Trace transit list is set to 1 then

- If the Trace status field in the Saved Modified Trace transit list is other than “trace in progress”, the Saved Modified Trace transit list shall be copied to the RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message. If this is the trace source node, then the procedures of Section 4.3.4.4 shall apply, otherwise no further processing of the Trace transit list information element shall be performed.
- If the Trace status field in the Saved Modified Trace transit list is set to “trace in progress”, then in addition to the normal call or party clearing and crankback procedures the Saved Modified Trace transit list information element shall be placed into the RELEASE, RELEASE COMPLETE or ADD PARTY REJECT message generated by the node towards the calling party and shall be modified as follows:
    - If this node did not refuse path trace and no logical node / logical port entry has been added to the Trace transit list information element for this node during processing of the current message, one logical node / logical port entry shall be added with the logical node ID of the lowest level node. The logical port ID shall be set to zero.
    - If this node is the DTL originator or the call or party is cleared without crankback, then the Trace status field in the Trace transit list information element shall be set to “trace incomplete”, and no crankback field shall be added to the Trace transit list information element.
    - If the call or party is cleared with crankback and this node is not the DTL originator, then: a crankback field shall be added to the Trace transit list information element with the appropriate Crankback Cause, Blocked Transit Type and Blocked Transit Trace Information values from the generated Crankback information element
    - If this is the trace source node, then the procedures of Section 4.3.4.4 shall apply.

4.3.4 Procedures at a Node Receiving a Call Clearing Message

If the call clearing message is being sent towards the called party or if a CONNECT or ADD PARTY ACKNOWLEDGE message has already been sent towards the calling party, then the procedures of this section shall not apply.

When a node receives a call clearing message, the procedures of Section 4.3.4.1 or 4.3.4.3 shall be applied as appropriate. Additionally, if the node is also the trace source node, the procedures of Section 4.3.4.4 shall be applied as well.

4.3.4.1 Procedures When Receiving a Call Clearing Message at a Node That Is Not The Trace Destination Node

The procedures for handling a call clearing message are broken into three groups:

1. Receipt of a call clearing message without a Crankback information element.
2. Receipt of a call clearing message containing a Crankback information element when this node is not allowed to perform alternate routing and will simply progress the crankback backward.
3. Receipt of a call clearing message containing a Crankback information element when this node is allowed to perform alternate routing.

1. If the received RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message does not contain a Crankback information element, then

- If the call clearing message contains a Trace transit list information element, then
  - If this node refused to participate in the path trace, then
    - If the C flag in the Saved Original Trace transit list information element is set to 1, then the Saved Modified Trace transit list information element shall replace the Trace transit list information element in the received call clearing message.
    - Otherwise, the Trace transit list information element shall be removed from the call clearing message.
  - If this is not the trace source node, the Trace transit list information element shall be passed in the call clearing message.
  - Otherwise (this is the trace source node), the procedures of Section 4.3.4.4 shall apply.
- If the call clearing message does not contain a Trace transit list information element then
  - If the C flag in the Saved Original Trace transit list information element is set to 1, then the Saved Modified Trace transit list information element shall be included in the call clearing message. If the Trace status is "trace in progress", then the Trace status shall be set to "trace incomplete".
  - If the C flag in the Saved Original Trace transit list information element is set to 0, then normal call clearing procedures apply and no Trace transit list information element shall be included in the call clearing message.
  - If this node is the trace source node, the procedures of Section 4.3.5.3 shall apply.

2. If the received RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message contains a Crankback information element, and none of the following apply:

- the Blocked transit type is "call or party has been blocked at the succeeding end of this interface", or
- this node is DTL originator, or
- this node is an entry border node that generated DTLs for this call of equal or higher level than the crankback level,

then:

- If this node refused to participate in the path trace or this is the trace destination node, then if there is a Trace transit list information element in the received call clearing message, it shall be deleted.
- If there is a Trace transit list information element in the RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message, then
    - If the C flag in the Saved Original Trace transit list is set to 1, then
        - If this is the trace source node, then the procedures of Section 4.3.4.4 shall apply.
        - Otherwise, the Trace transit list information element in the message shall be passed towards the trace source node.
    - If the C flag in the Saved Original Trace transit list is set to 0, then the Trace transit list information element shall be removed from the call clearing message.
- If there is no Trace transit list information element in the RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message, then.
    - If the C flag in the Saved Original Trace transit list information element is set to 0 then normal call clearing procedures apply and no Trace transit list information element shall be included in the call clearing message.
    - If the C flag in the Saved Original Trace transit list information element is set to 1, then the Saved Modified Trace transit list information element shall be included in the call clearing message.
        - If the Trace status in the Saved Modified Trace transit list information element is "trace in progress", then a Crankback gap indicator field shall be appended to the Trace transit list information element, followed by a Crankback field with the appropriate Crankback Cause, Blocked Transit Type, and Blocked Transit Trace Information values from the received Crankback information element.
        - Otherwise (the Trace status in the Saved Modified Trace transit list information element is not "trace in progress"), nothing further is added to the Trace transit list information element at this time.
    - If this is the trace source node, then the procedures of Section 4.3.4.4 shall apply.

3. If the received RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message contains a Crankback information element, and at least one of the following applies:

- the Blocked transit type is "call or party has been blocked at the succeeding end of this interface",
- this node is DTL originator, or
- this node is an entry border node that generated DTLs for this call of equal or higher level than the crankback level,

then the procedures of Section 4.3.4.2 shall apply with the Current Trace transit list information element determined as follows:

- If this node refused to participate in the path trace, then there is no Current Trace transit list information element and the remaining procedures of this section shall not apply.
- If the C flag in the Saved Original Trace transit list information element is set to 0 then
    - If the Trace status in the Saved Original Trace transit list information element is set to "trace in progress", then the Saved Original Trace transit list shall be used as the Current Trace transit list information element.
    - If the Trace status in the Saved Original Trace transit list information element is not set to "trace in progress", then there is no Current Trace transit list information element.

- If the C flag in the Saved Original Trace transit list information element is set to 1, then
  - If this is not the trace destination node and if a Trace transit list information element is present in the received RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message, that Trace transit list information element shall be saved as the Saved Modified Trace transit list information element. If the Trace status is set to "trace in progress", then this Trace transit list information element shall be used as the Current Trace transit list information element, otherwise, there is no Current Trace transit list information element.
  - If this is the trace destination node or if no Trace transit list information element is present in the received RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message, then:
    - If the Trace status in the Saved Modified Trace transit list information element is "trace in progress", then a Crankback gap indicator field shall be appended to the Saved Modified Trace transit list information element, followed by a Crankback field with the appropriate Crankback Cause, Blocked Transit Type, and Blocked Transit Trace Information values from the received Crankback information element. The resulting Saved Modified Trace transit list information element shall be used as the Current Trace transit list information element.
    - Otherwise (the Trace status in the Saved Modified Trace transit list information element is not "trace in progress"), there is no Current Trace transit list information element.

4.3.4.2 Procedures at a Node Allowed to Perform Alternate Routing

Examples of nodes allowed to perform alternate routing are: DTL originators, some entry border nodes, nodes receiving a crankback indicating "Call or party has been blocked at the succeeding end of this interface" and nodes receiving AINI crankback.

A node that attempts alternate routing shall apply the following procedures in addition to standard PNNI crankback and alternate routing procedures:

- If there is a Current Trace transit list information element, then the Current Trace transit list shall be copied into the new SETUP or ADD PARTY message, and the Trace transit list procedures specified in Section 4.3.1.2 (commencing with step 6) shall apply.

If the node does not attempt alternate routing or if alternate routing fails for any reason, the procedures of section 4.3.3 shall apply, if they have not yet been applied as part of the above procedures, before the call clearing message is propagated towards the trace source node. If the C flag is set to 1 in the Saved Original Trace transit list, then these procedures will cause a new logical node / logical port entry and crankback field to be added to the message's Trace transit list information element.

4.3.4.3 Procedures at a Trace Destination Node

If a RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message is received at a trace destination node and the received message does not contain a Crankback information element, then

- The Saved Modified Trace transit list information element shall be copied into the call clearing message.
- If the Trace status field in the message is set to "trace in progress", then it shall be set to "trace completed normally".
- Normal call clearing procedures shall then apply.

If a RELEASE, RELEASE COMPLETE or ADD PARTY REJECT message received at a trace destination node contains a Crankback information element and it was received over a PNNI interface, then the procedures of Sections 4.3.4.1 or 4.3.4.3 shall apply.

If a RELEASE, RELEASE COMPLETE or ADD PARTY REJECT message received at a trace destination node contains a Crankback information element and it was received over a non-PNNI interface, then

- A "Crankback received at trace destination node" indicator shall be added to the Saved Modified Trace transit list information element.
- The procedures of Section 4.3.4.2 shall then apply with the Current Trace transit list set to the Saved Modified Trace transit list.

4.3.4.4 Procedures at a Trace Source Node

The trace source node shall remove the Trace transit list information element before progressing a RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message towards the calling party if the message contains the information element. If the Trace status field is set to "trace in progress", then it shall be changed to "trace incomplete".

4.3.5 Procedures at a Node Receiving a Call Accept Message

When a node receives a call accept message, the procedures of Section 4.3.5.1 or 4.3.5.2 shall be applied as appropriate. Additionally, if the node is also the trace source node, the procedures of Section 4.3.5.3 shall be applied as well.

4.3.5.1 Procedures at a Node That Is Not The Trace Destination Node

If the received CONNECT or ADD PARTY ACKNOWLEDGE message does not contain a Trace transit list information element or this node refused to participate in path tracing, then

- The Saved Modified Trace transit list information element shall be included in the CONNECT or ADD PARTY ACKNOWLEDGE message.
- If the Trace status field is set to "trace in progress", then the Trace status field in the message shall be set to “trace incomplete”.

Otherwise, the Trace transit list information element from the received CONNECT or ADD PARTY ACKNOWLEDGE message remains unmodified.

4.3.5.2 Procedures at a Trace Destination Node

- The following procedures shall apply:The Saved Modified Trace transit list information element shall be included in the CONNECT or ADD PARTY ACKNOWLEDGE message.
- If the Trace status field is set to "trace in progress", then the Trace status field in the message shall be set to “trace completed normally”.

4.3.5.3 Procedures at a Trace Source Node

The trace source node shall remove the Trace transit list information element before progressing a CONNECT or ADD PARTY ACKNOWLEDGE message towards the calling party. If the Trace status field is set to "trace in progress", then it shall be changed to "trace incomplete".

4.3.6 Trace Transit List Information Element Content Validation

Nodes other than the trace source node shall not perform any content validation on those portions of a received Trace transit list information element commencing with the Trace source port identifier. This allows for future extensions to the Trace transit list information element. Only the information element length and Trace status may be modified. New fields may be appended to the Trace transit list information element.

4.3.7 Length Errors

The following procedures apply:

- at any point during the trace, if the length of the Trace transit list information element exceeds the maximum length, or
- when the maximum message length is exceeded at any point during message processing (not necessarily during processing of the Trace transit list information element, for example while adding Designated transit list information elements) and a Trace transit list information element is present in the message.

The node shall set the Trace status to "trace has exceeded information element length limitations" or "trace has exceeded message length limitations," as appropriate.

If the Trace transit list information element indicates that the call is to be released at the trace destination (i.e., the X flag is set to 1), then the call shall be released immediately by sending a RELEASE, RELEASE COMPLETE or ADD PARTY REJECT message including the Trace transit list information element. The cause code in the Cause information element shall be set to cause #31 “normal, unspecified.”

Otherwise, the Trace transit list information element shall be removed from the message and saved locally as the Saved Modified Trace transit list. In addition, if the Trace transit list information element has not yet been saved above as the Saved Original Trace transit list, the current Trace transit list information element shall be saved as the Saved Original Trace transit list.

4.4 Compatibility with Nodes Not Supporting This Feature

[NORMATIVE]

If the X flag in the Trace transit list information element is set to 0, the trace source node shall set the IE instruction field flag in the Trace transit list information element to “follow explicit instructions” and the IE action indicator to “discard information element and proceed”.

If the X flag in the Trace transit list information element is set to 1, the trace source node shall set the IE instruction field flag in the Trace transit list information element to “follow explicit instructions” and the IE action indicator to “clear call”.

For all messages other than SETUP, ADD PARTY, and CO-BI SETUP, the Pass along request bit shall be set to "pass along request".

Vendor equipment shall support the ability to set the Pass along request bit in SETUP, ADD PARTY, and CO-BI SETUP messages to "pass along request". It is recommended that networks be configured so that the Pass along request bit is set to “pass along request”. This will increase the chances for partial trace information to be returned when the connection or party traverses one or more nodes that do not support path trace functionality. If the Pass along request bit is ever set to “pass along request” in a given network, all nodes at the edge of the network should support path trace functionality. Otherwise there will be some risk of trace information being exposed outside of the network.

5 Connection Trace

5.1 Connection Trace Description

[INFORMATIVE]

Connection tracing is used to determine the logical nodes and logical links traversed by an existing connection. The trace information may also include VPI/VCI and DLCI values, call reference values ~~and~~, endpoint reference values, incoming Ne-NSC and Rp-NSC lists, outgoing Ne-NSC and Rp-NSC lists, and interworking LSP labels, if indicated by the flags in the Flag octet of the Trace transit list information element. Both point-to-point and point-to-multipoint connections are supported.

Connection tracing may be carried out in either direction: towards the calling party, or towards the called party. Connection tracing returns the path from the trace source node, which may be any node on the connection, to the trace destination node. Trace information is accumulated in the TRACE CONNECTION message sent from the trace source node to the trace destination node. The trace destination node returns the trace information in the TRACE CONNECTION ACKNOWLEDGE message.

Figure 5-1 demonstrates tracing of an established point-to-point SVC connection between ATM UNIs from the calling party to the called party.

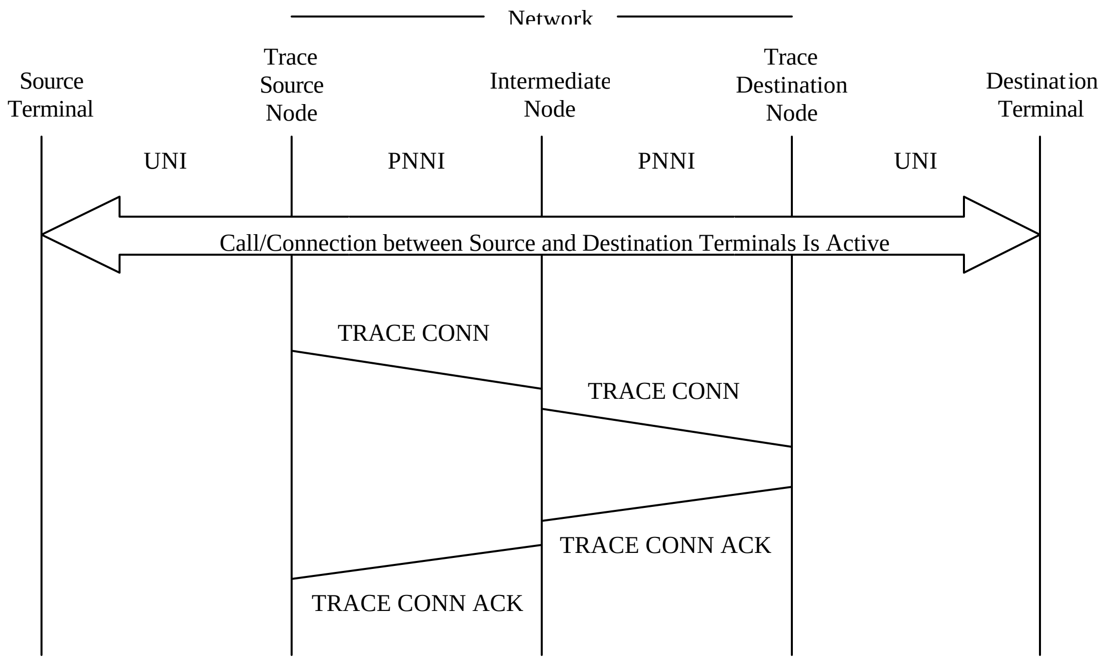


Figure 5-1 Sample Message Sequence for Connection Trace Towards the Called Party

The steps in tracing the connection are as follows:

1. Network management requests the trace source node to trace a given connection by providing the trace source interface and VPI/VCI or trace source interface and call reference, the trace direction and the trace flags.
2. The trace source node creates a TRACE CONNECTION message and adds a Trace transit list information element along with the flags provided by network management. The following information is then added:
   a) A port ID identifying the trace source UNI interface.

b) If the V flag is set, the VPI/VCI used on the trace source interface.
c) If the A flag is set, the call reference used on the trace source interface.
d) If the Ne flag is set and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface. The setting of the I flag does not influence the inclusion of the incoming interface report at the trace source node.
e) If the Rp flag is set and the connection is supported by a specific resource partition of the incoming interface, the list of Rp-NSCs supporting the connection at the incoming interface. The setting of the I flag does not influence the inclusion of the incoming interface report at the trace source node.
f) If the L flag is set and the incoming interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs used for the connection,
g) Its own node ID and the outgoing port ID.
h) If the Ne flag is set and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.
i) If the Rp flag is set and the connection is supported by a specific resource partition of the outgoing interface, the list of Rp-NSCs supporting the connection at the outgoing interface.

The trace source node then ~~adds its own node ID and its outgoing port ID and~~ forwards the TRACE CONNECTION message to the next node.

3. The intermediate node receives the TRACE CONNECTION message and adds the following information:
a) If the V flag is set, the VPI/VCI value on the incoming interface.
b) If the A flag is set, the call reference value on the incoming interface.
c) If both the I and Ne flags are set and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface.
d) If both the I and Rp flags are set and the connection is supported by a specific resource partition of the incoming interface, the list of Rp-NSCs supporting the connection at the incoming interface.
e) If the L flag is set and the incoming interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs used for the connection,
f) Its own node ID and the outgoing port ID.
g) If the Ne flag is set and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.
h) If the Rp flag is set and the connection is supported by a specific resource partition of the outgoing interface, the list of Rp-NSCs supporting the connection at the outgoing interface.

The intermediate node then ~~adds its own node ID and the outgoing port ID and~~ forwards the TRACE CONNECTION message to the next node.

4. The trace destination node adds information about the incoming interface (VPI/VCI, call reference, incoming Ne-NSCs, incoming Rp-NSCs, interworking LSP labels) to the Trace transit list information element in the same manner as intermediate nodes. It then adds its own node ID and outgoing port ID. Information about the destination UNI interface is also added as follows:
a) If the Ne flag is set and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.
b) If the Rp flag is set and the connection is supported by a specific resource partition of the outgoing interface, the list of Rp-NSCs supporting the connection at the outgoing interface.
c~~a~~) If the V flag is set, the VPI/VCI used on the destination UNI.
d~~b~~) If the A flag is set, the call reference used on the destination UNI.
e) If the L flag is set and the outgoing interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs used for the connection,

The trace destination node sets the Trace status field to “trace completed normally”, creates a TRACE CONNECTION ACKNOWLEDGE message, adds the Trace transit list information element to the TRACE CONNECTION ACKNOWLEDGE message and sends it back towards the trace source node.

5. Intermediate nodes forward the TRACE CONNECTION ACKNOWLEDGE message without any further processing

6. At the trace source node, the Trace transit list information element is removed from the TRACE CONNECTION ACKNOWLEDGE message and saved for retrieval by network management. The TRACE CONNECTION ACKNOWLEDGE message is not progressed further by the trace source node.

Figure 5-2 demonstrates tracing of an established point-to-point SVC connection between ATM UNIs from the called party to the calling party. The steps followed are analogous to those described for Figure 5-1.

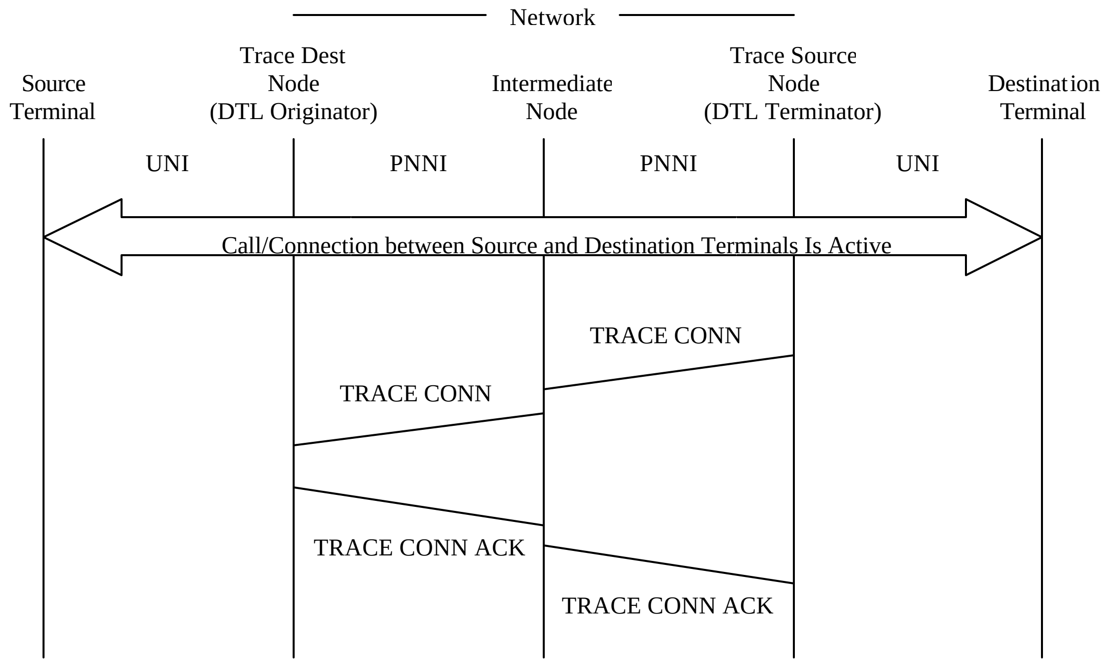

Network
Source Terminal
Trace Dest Node (DTL Originator)
Intermediate Node
Trace Source Node (DTL Terminator)
Destination Terminal
UNI
PNNI
PNNI
UNI
Call/Connection between Source and Destination Terminals Is Active
TRACE CONN
TRACE CONN
TRACE CONN ACK
TRACE CONN ACK

Figure 5-2 Sample Message Sequence for Connection Trace Towards the Calling Party

5.2 Messages

[NORMATIVE]

Table 5-1 shows the new messages added for the connection trace capability.

Table 5-1 Messages Used with ATM Connection Trace

Message	Reference
TRACE CONNECTION	5.2.1
TRACE CONNECTION ACKNOWLEDGE	5.2.2

Table 5-2 shows the new codings used for these messages. These are in addition to those of Table 6-1/PNNI 1.0 in Section 6.3.1/PNNI 1.0, Table 6-2/PNNI 1.0 in Section 6.3.2/PNNI 1.0 and Table 6-3/PNNI 1.0 in Section 6.3.3/PNNI 1.0.

Table 5-2 Connection Trace Additional Codings

Message type (octet 1)

Bits	Meaning

8	7	6	5	4	3	2	1	
1	0	0	0	1	1	0	0	TRACE CONNECTION
1	0	0	0	1	1	0	1	TRACE CONNECTION ACKNOWLEDGE

5.2.1 TRACE CONNECTION

This message is sent to trace the logical nodes and logical links traversed by an existing connection.

Message type:	TRACE CONNECTION
Direction:	Both
Significance:	Global

Information Element	Reference	Type	Length
Protocol discriminator	6.4.2/PNNI 1.0	M	1
Call reference	6.4.3/PNNI 1.0	M	4
Message type	6.4.4.1/PNNI 1.0	M	2
Message length	6.4.4.2/PNNI 1.0	M	2
Endpoint reference	6.4.8.1/PNNI 1.0	O(1)	7
Trace transit list	3.1	M	33-1466(2)

Note 1 - Included only when the call reference identifies a point-to-multipoint connection.
Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network.

Figure 5-3 TRACE CONNECTION message contents

5.2.2 TRACE CONNECTION ACKNOWLEDGE

This message is sent to acknowledge a TRACE CONNECTION message and to return the Trace Transit List information.

Message type:	TRACE CONNECTION ACKNOWLEDGE
Direction:	Both
Significance:	Global

Information Element	Reference	Type	Length
Protocol discriminator	6.4.2/PNNI 1.0	M	1
Call reference	6.4.3/PNNI 1.0	M	4
Message type	6.4.4.1/PNNI 1.0	M	2
Message length	6.4.4.2/PNNI 1.0	M	2
Endpoint reference	6.4.8.1/PNNI 1.0	O(1)	7
Trace transit list	3.1	M	33-1466(2)

Note 1 - Included if the Endpoint reference information element was included in the TRACE CONNECTION message being responded to. The endpoint reference value must be the same as the value in the TRACE CONNECTION message being responded to.

Note 2 - A larger maximum length value may be supported as a network specific option. The maximum length should be consistent among all nodes in the network.

Figure 5-4 TRACE CONNECTION ACKNOWLEDGE message contents

5.3 Connection Trace Procedures

[NORMATIVE]

5.3.1 Processing a TRACE CONNECTION Message

In the following procedures, the sections describing V flag processing do not apply to the Connection Trace of Connection Oriented Bearer Independent (CO-BI) calls.

The trace source node initiates the connection trace by creating a TRACE CONNECTION message including a Trace transit list information element, to be sent on the outgoing interface in the appropriate direction. The connection or party to be traced is identified by the call reference and, for parties of point-to-multipoint connections, the endpoint reference. The trace transit list records the path traversed by the SETUP or ADD PARTY message from the trace source node to the trace destination node. The trace source node controls what path information is recorded by setting the V (VPI/VCI) ~~and~~ , A (call/endpoint reference), Ne (Ne-NSC lists), Rp (Rp-NSC lists), I (Incoming NSCs), and L (interworking LSP labels) flags in the Trace transit list information element. If the V flag is set to 1, the VPI/VCI values used on each link are recorded. In the case of Frame Relay endpoints, the DLCI values of those endpoints are also recorded. If the A flag is set to 1, the call reference values used on each link, and endpoint reference values for point-to-multipoint connections, are recorded. The flags C and X in the Trace transit list information element shall be set to zero when originating a TRACE CONNECTION message and shall be ignored during processing. If the Ne flag is set to 1, the Ne-NSCs supporting the connection are recorded. If the Rp flag is set to 1, the Rp-NSCs supporting the connection are recorded. If the I flag is set to 1, the NSCs supporting the connections at the nodes' incoming interfaces are recorded. If the L flag is set to 1 and the interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs.

After adding the Flags field to the Trace transit list information element, the trace source node shall add the Trace status field with the value set to "trace in progress". Trace information shall then be added to the Trace transit list information element as described in the following procedures.

Nodes other than the trace source node shall not perform any content validation on those portions of the received Trace transit list information element after the Trace status field. This allows for future extensions to the Trace transit list information element. Only the information element length and Trace status may be modified. New fields may be appended to the Trace transit list information element.

Each node traversed by the TRACE CONNECTION message, including the trace source node, shall add the following fields to the Trace transit list information element, in this order, before progressing the message:

1. If a TRACE CONNECTION message is received over an incoming interface, then:

   - If the connection or party is in the null state, then the TRACE CONNECTION message shall be treated as an unexpected message and the procedures of section 6.5.6.3.2/PNNI1.0 shall apply.
   - If the connection or party is in a clearing state, then the TRACE CONNECTION message shall be discarded and no further action shall be taken.
   - Otherwise, if the connection is not in the Active call state or the party is not in the Active party state on the incoming interface, then the node shall set the Trace status to "trace incomplete", copy the Trace transit list information element from the TRACE CONNECTION message to a TRACE

CONNECTION ACKNOWLEDGE message, and send it on the incoming interface and processing is complete.

- Otherwise, if the Trace status in the Trace transit list information element is not "trace in progress", then the node shall copy the Trace transit list information element from the TRACE CONNECTION message to a TRACE CONNECTION ACKNOWLEDGE message and send it on the incoming interface and processing is complete.

2. If this node is the trace source node and the direction of the trace is incoming from the trace source interface, the Trace source port identifier field with the value either set to zero (if the Trace source interface is not a PNNI interface and no logical port ID is assigned to this interface) or identifying the trace source interface from which the path trace is initiated;
3. If the V flag is set to 1 and
   - this node is the trace source node and the trace direction is specified as the incoming direction from the trace source interface, then:
     - if the trace source interface is an ATM interface, a VPI/VCI entry for the trace source interface (note that what is traced is the actual VPI used for the connection on this side of the trace source interface, not the VPCI),
     - if the trace source interface is a Frame Relay interface and a 1:1 mapping is used between the Frame Relay and the ATM connection, a DLCI entry for the trace source interface,
     - if the trace source interface is neither an ATM interface, nor a Frame Relay interface (e.g. Circuit Emulation Service), a VPI/VCI entry may be added for the trace source interface,
   - this node is not the trace source node, a VPI/VCI entry for the interface on which the TRACE CONNECTION message was received (note that what is traced is the actual VPI used for the connection on this side of the interface, not the VPCI);
4. If the A flag is set to 1 and
   - this node is the trace source node, the trace direction is specified as the incoming direction from the trace source interface, the connection leg on the trace source interface is switched, and the trace source interface supports a call reference that can be specified in 3 octets, a call reference entry for the trace source interface including the endpoint reference field for point-to-multipoint connections,
   - this node is not the trace source node, a call reference entry for the interface on which the TRACE CONNECTION message was received, including the endpoint reference field for point-to-multipoint connections;
5. If the Ne flag is set to 1, this node does not refuse to participate in connection traces received over the incoming interface,
   - this node is the trace source node and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface.
   - this node is not the trace source node, the I flag is set to 1, and the incoming interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the incoming interface.
6. If the Rp flag is set to 1, this node does not refuse to participate in connection traces received over the incoming interface,
   - this node is the trace source node and
     - the resources supporting the connection at the incoming interface are tagged by Rp-NSCs, the list of Rp-NSCs supporting the connection at the incoming interface.
     - the resources supporting the connection at the incoming interface are not tagged by any Rp-NSCs the "Rp-NSC Bare" value.

    - the incoming interface is tagged by all Rp-NSCs the "Rp-NSC Bare" value.

- this node is not the trace source node, the I flag is set to 1

    - the resources supporting the connection at the incoming interface are tagged by Rp-NSCs, the list of Rp-NSCs supporting the connection at the incoming interface.

    - the resources supporting the connection at the incoming interface are not tagged by any Rp-NSCs the "Rp-NSC Bare" value.

    - the incoming interface is tagged by all Rp-NSCs the "Rp-NSC Bare" value.

7. If the L flag is set to 1, this node does not refuse to participate in connection traces received over the incoming interface, and the incoming interface on which the TRACE CONNECTION message was received is an ATM-MPLS network interworking interface, the labels of the interworking LSPs to be used for the connection.

~~5.~~8. If this node refuses to participate in connection traces received over the incoming interface, a logical node / logical port entry including this node's lowest-level logical node identifier. The logical port identifier shall be set to zero. In addition, the Trace Continuation refusal indicator shall be inserted into the Trace transit list information element and the Trace status field shall be set to "trace incomplete". Rather than continuing to progress the TRACE CONNECTION message, a TRACE CONNECTION ACKNOWLEDGE message shall be returned on the interface over which the TRACE CONNECTION message was received. The Trace transit list information element from the TRACE CONNECTION message, as modified by the procedures above, shall be copied into the TRACE CONNECTION ACKNOWLEDGE message before progressing the message towards the trace source node;

~~6.~~9. One logical node / logical port entry where the logical node ID specifies the lowest level logical node and

- if this node is not the trace destination node, the logical port specifies the outgoing interface on which the TRACE CONNECTION message will be progressed,

- if this node is the trace destination node, the logical port is either set to zero (if the next link is not a PNNI interface and no logical port ID is assigned to this interface) or specifies the next link traversed by the connection or party;

10. If the Ne flag is set to 1 and the outgoing interface is tagged by Ne-NSCs, the list of Ne-NSCs supporting the connection at the outgoing interface.

11. If the Rp flag is set to 1 and

- the resources supporting the connection at the outgoing interface are tagged by Rp-NSCs, the list of Rp-NSCs supporting the connection at the outgoing interface.

- the resources supporting the connection at the outgoing interface are not tagged by any Rp-NSCs the "Rp-NSC Bare" value.

- the outgoing interface is tagged by all Rp-NSCs the "Rp-NSC Bare" value.

~~7.~~12. If the connection is not in the Active call state or the party is not in the Active party state on the outgoing interface, then the node shall set the Trace status to "trace incomplete", copy the Trace transit list information element from the TRACE CONNECTION message to a TRACE CONNECTION ACKNOWLEDGE message, the and send it on the incoming interface and processing is complete. If the outgoing interface is not known, then the logical port ID entry added in step 6 shall be coded to zero.

~~8.~~13. If this node is the trace destination node and the V flag is set to 1:

- If the trace destination interface is an ATM interface, a VPI/VCI entry for the trace destination interface;

- If the trace destination interface is a Frame Relay interface and a 1:1 mapping is used between the Frame Relay and the ATM connection, a DLCI entry for the trace destination interface;

- If the trace destination interface is not an ATM interface or a Frame Relay interface (e.g. Circuit Emulation Service), a VPI/VCI entry may be added for the trace destination interface.

14. If this node is the trace destination node, the L flag is set to 1, and the trace destination interface is an ATM-MPLS network interworking interface, the labels of the interworking LSPs to be used for the connection.

~~9.~~15. If this node is the trace destination node, the A flag is set to 1, the connection leg on the trace destination interface is switched, and the trace destination interface supports a call reference that can be specified in 3 octets, a call reference entry for the trace destination interface including the endpoint reference for point-to-multipoint connections.

The trace destination node shall acknowledge a TRACE CONNECTION message by sending a TRACE CONNECTION ACKNOWLEDGE message on the incoming interface over which the TRACE CONNECTION message was received. The Trace transit list information element in the TRACE CONNECTION ACKNOWLEDGE message shall be derived by copying the content of the Trace transit list information element from the TRACE CONNECTION message, after adding appropriate fields as described above, and setting the Trace status field to “trace completed normally”.

The procedures of Section 5.3.3 shall be followed to check for Trace transit list information element length exceeded errors and message length exceeded errors.

During the procedures above, if the node cannot progress the TRACE CONNECTION message towards the trace destination node due to conditions not stated above, the following procedures shall apply:

- The node shall copy the Trace transit list information element from the TRACE CONNECTION message to a TRACE CONNECTION ACKNOWLEDGE message,

- If the Trace status is "trace in progress", then the node shall set the Trace status in the Trace transit list information element to “trace incomplete”

- The node shall send the TRACE CONNECTION ACKNOWLEDGE message on the incoming interface.

5.3.2 Processing a TRACE CONNECTION ACKNOWLEDGE Message

When a node other than the trace source node receives a TRACE CONNECTION ACKNOWLEDGE message, it progresses it across the incoming interface over which the TRACE CONNECTION message was received. The node shall not perform any content validation on the received Trace transit list information element.

When the trace source node receives a TRACE CONNECTION ACKNOWLEDGE message, it shall examine the first logical node entry in the received Trace transit list information element. If the first logical node id of the trace transit list information element corresponds to the trace source node’s logical node id, it shall save the Trace transit list information and terminate the progression of the message. If the first logical node id of the trace transit list does not correspond to the trace source node’s id, the Trace transit list information element shall not be saved and the node shall not act as the trace source node.

5.3.3 Length Errors

The following procedures apply:

- at any point during the trace, if the length of the Trace transit list information element exceeds the maximum length, or

- when the maximum message length is exceeded at any point during processing of a TRACE CONNECTION message (not necessarily during processing of the Trace transit list information element).

The node shall set the Trace status to "trace has exceeded information element length limitations" or "trace has exceeded message length limitations," as appropriate.

The node shall copy the Trace transit list information element from the TRACE CONNECTION message to a TRACE CONNECTION ACKNOWLEDGE message and shall send the TRACE CONNECTION ACKNOWLEDGE message on the incoming interface.

5.4 Compatibility with Nodes Not Supporting This Feature

[NORMATIVE]

The trace source node shall set the Message instruction field flag in the TRACE CONNECTION and TRACE CONNECTION ACKNOWLEDGE messages to “follow explicit instructions” and the Message action indicator to either “discard and ignore” or “discard and report status”. The IE instruction field flag in the Trace transit list information element shall be set to “follow explicit instructions” and the IE action indicator to either “discard message, and ignore” or “discard message, and report status”.

The Pass along request bit shall be set to "pass along request" in TRACE CONNECTION ACKNOWLEDGE messages.

Vendor equipment shall support the ability to set the Pass along request bit in TRACE CONNECTION messages to "pass along request". It is recommended that networks be configured so that the Pass along request bit in the TRACE CONNECTION message and in the Trace transit list information element be set to “pass along request”. This will allow for partial trace information to be returned when the connection or party traverses one or more nodes that do not support connection trace functionality. However, no trace information will be returned, regardless of the setting of the Pass along request bit, when the intended trace destination node does not support connection trace functionality. If the Pass along request bit is ever set to “pass along request” in a given network, all nodes at the edge of the network should support connection trace functionality. Otherwise there will be some risk of trace information being exposed outside of the network.

If the Pass along request bit is set to “no pass along request”, and the connection or party traverses one or more nodes that do not support connection trace functionality, then no trace information will be returned.

6 References

[1] *Private Network-Network Interface Specification Version 1.1~~0~~*, The ATM Forum Technical Committee, af-pnni-0055.002, April 2002~~0, March 1996~~.

[2] *Network-to-network interface between public data networks providing the frame relay data transmission service, Amendment 1: switched virtual circuits*, ITU-T Recommendation X.76 - Amendment 1, August 1997.

[3] *PNNI Addendum on PNNI/B-QSIG Interworking and Generic Functional Protocol for the Support of Supplementary Services*, The ATM Forum Technical Committee, af-cs-0102.000, October 1998.

[4] *Policy Routing Version 1.0*, The ATM Forum Technical Committee, af-cs-0195.000, April 2003

[5] *Private Network-Network Interface Specification Version 1.0*, The ATM Forum Technical Committee, af-pnni-0055.000, March 1996

[6] *PNNI v1.0 Errata and PICS*, The ATM Forum Technical Committee, af-pnni-0081.000, May 1997

[7] *ATM-MPLS Network Interworking Signalling Specification, Version 1.0*, The ATM Forum Technical Committee, af-cs-0197.000, August 2003

Annex A Protocol Implementation Conformance Statement (PICS) for PNNI 1.~~0~~ 1 Path Trace

A.1 Introduction

To evaluate conformance of a particular implementation, it is necessary to have a statement of which capabilities and options have been implemented. Such a statement is called a Protocol Implementation Conformance Statement (PICS).

A.1.1 Scope

This document provides the PICS proforma for the Addendum to PNNI 1.~~0~~ 1 for the support of Path and Connection Trace, as specified in this document in compliance with the relevant requirements, and in accordance with the relevant guidelines, given in ISO/IEC 9646-2 [CTMI]. In most cases, statements contained in notes in the specification, which were intended as information, are not included in the PICS.

A.1.2 Normative References

[1] ISO/IEC 9646-1: 1994, Information technology – Open systems interconnection – Conformance testing methodology and framework – Part 1: General Concepts (See also ITU Recommendation X.290 (1995)).

[2] ISO/IEC 9646-2:1994, Information technology – Open systems interconnection – Conformance testing methodology and interconnection – Part 2: Abstract test suite specification (See also ITU telecommunication X.291 (1995)).

[3] PNNI Addendum for Path and Connection Trace V1.1, The ATM Forum Technical Committee, af-cs-0141.002, October 2003

~~[4] Private Network to Network Interface V1.0, The ATM Forum Technical Committee, af-pnni-0055.000~~

~~[5] Private Network to Network Interface V 1.0, The ATM Forum Technical Committee, af-pnni-0081.000~~

[~~GFS~~4] PNNI Addendum on PNNI/B-QSIG Interworking and Generic Functional Protocol for Support of Supplementary Service, AF-CS-0102.000, October 1998.

[5] Policy Routing Version 1.0, The ATM Forum Technical Committee, af-cs-0195.000, April 2003

[6] ATM-MPLS Network Interworking Signalling Specification, Version 1.0, The ATM Forum Technical Committee, af-cs-0197.000, August 2003

A.1.3 Definitions

This following terms defined in ISO/IEC 9646-1[CTMF] are used in this document:

- A Protocol Implementation Conformance Statement (PICS) is a statement made by the supplier of an implementation or system, stating which capabilities have been implemented for a given protocol.
- A PICS proforma is a document, in the form of a questionnaire, designed by the protocol specifier or conformance test suite specifier, which when completed for an implementation or system becomes the PICS.

A.1.4 Acronyms

I.E.	Information Element
IUT	Implementation under test
M	Mandatory requirements (these are to be observed in all cases)
N/A	Not supported, not applicable, or the conditions for status are not met.
O	Optional (may be selected to suit the implementation, provided that any requirements applicable to the options are observed)
O.n	Optional, but support is required for either at least one or only one of the options in the group labelled with the same numeral "n".
PICS	Protocol Implementation Conformance Statement
SUT	System under test

A.1.5 Conformance

The supplier of a protocol implementation which is claimed to conform to the ATM Forum PNNI signalling Addendum for the support of Path Trace is required to complete a copy of the PICS proforma provided in this document and is required to provide the information necessary to identify both the supplier and the implementation.

A.2 Identification of the Implementation

Implementation Under Test (IUT) Identification

IUT Name:____

IUT Version: ____

____

System Under Test (SUT) Identification

SUT Name:____

Hardware Configuration: ____

____

____

Operating System: ____

Product Supplier

Name:____

Address:____

____

Telephone Number: ____

Facsimile Number: ____

Email Address:____

Additional Information: ____

Client

Name:____

Address:
____

____

Telephone Number: ____

Facsimile Number: ____

Email Address:____

Additional Information: ____

PICS Contact Person

Name:____

Address: ____

____

Telephone Number: ____

Facsimile Number: ____

Email Address:____

Additional Information: ____

PICS/System Conformance Statement

Provide the relationship of the PICS with the System Conformance Statement for the system:
____
____
____

Identification of the protocol

This PICS proforma applies to the following:
The sections pertaining to Path Trace in the *PNNI Addendum for Path and Connection Trace Version 1.0*.

A.3 PICS Proforma

A.3.1 Global statement of conformance

The implementation described in this PICS meets all of the mandatory requirements of the reference protocol.

[ ] YES
[ ] NO

Note: Answering "No" indicates non-conformance to the specified protocol. Non-supported mandatory capabilities are to be identified in the following tables, with an explanation by the implementor explaining why the implementation is non-conforming.

A.3.2 Instructions for Completing the PICS Proforma

The PICS Proforma is a fixed-format questionnaire. Answers to the questionnaire should be provided in the rightmost columns, either by simply indicating a restricted choice (such as Ye s or No), or by entering a value or a set of range of values.

A supplier may also provide additional information, categorised as exceptional or supplementary information. These additional information should be provided as items labelled X.<i> for exceptional information, or S.<i> for supplemental information, respectively, for cross reference purposes, where <i> is any unambiguous identification for the item. The exception and supplementary information are not mandatory and the PICS is complete without such information. The presence of optional supplementary or exception information should not affect test execution, and will in no way affect interoperability verification. The column labelled 'Reference' gives a pointer to sections of the protocol specification for which the PICS Proforma is being written.

A.3.3 Major Capability (MC)

Item Number	**Item Description**	**Status**	**Condition for status**	**Reference**	**Support**
MC 1	Does the IUT support path trace?	M		4	Yes__No__

A.3.4 Subsidiary Capabilities (SC)

Item Number	**Item Description**	**Status**	**Condition for status**	**Reference**	**Support**
SC 1	Does the IUT support path trace for point-to-point connections?	M		4	Yes__No__
SC 2	Does the IUT support path trace for point-to-multipoint connections?	M		4	Yes__No__
SC 3	Does the IUT support all trace status codes?	M		3.1	Yes__No__
SC 4	Does the IUT support crankback tracing?	M		3.1; 4.3.1.2; 4.3.3; 4.3.4	Yes__No__
SC 5	Does the IUT support Path Trace with out crankback information?	M		3.1; 4.3.1.2; 4.3.3; 4.3.4	Yes__No__
SC 6	Does the IUT support call clearing at trace destination?	M		3.1; 4.3.2	Yes__No__
SC 7	Does the IUT support VPI/VCI tracing?	M		3.1; 4.3.1.2	Yes__No__
SC 8	Does the IUT support Frame Relay interface as the trace source/destination interface ?	O		3.1; 4.3.1.2	Yes__No__
SC 9	Is the IUT capable of recognising and	M		3.1; 4.3.1.2	Yes__No__

	interpreting the encoding of DLCIs?				
SC 10	Does the IUT support call reference value tracing?	M		3.1; 4.3.1.2	Yes__No__
SC 11	Does the IUT support endpoint reference value tracing?	M		3.1; 4.3.1.2	Yes__No__
SC 12	Does the IUT support insertion of Vendor Specific Information?	O		3.1; 4.3.1.2	Yes__No__
SC 13	Is the IUT capable of recognising the presence and length of the Vendor Specific octet group ?	M		3.1; 4.3.1.2	Yes__No__
SC 14	Does the IUT support the tracing of Connection Oriented Bearer Independent (CO-BI) calls?	M	MC2.3 from [~~GFS~~4]	4.3.1	Yes__No__
SC 15	Does the IUT support PNNI Crankback Gap tracing?	M		3.1; 4.3.1.2	Yes__No__
SC 16	Can the IUT be configured to set the pass along request bit in the TTL information element to pass along request?	M		4.4	Yes__No__
SC 17	Does the IUT support the refusal of trace for a call with a TTL information element?	O		3.1;4.3.1.2	Yes__No__
SC 18	Does the IUT recognise the PNNI trace continuation refusal indicator?	M		3.1	Yes__No__
SC 19	Does the IUT support generation of Crankback Received at trace destination node indicator?	O		3.1; 4.3.4.3	Yes__No__
SC 20	Does the IUT recognise the Crankback Received at Trace Destination node indicator?	M		3.1	Yes__No__
SC 21~~C 21~~	Does the IUT support Ne-NSC value tracing?	M	MCP1 from [5]	3.1; 4.3.1.2	Yes__No__
SC 22	Does the IUT support Rp -NSC value tracing?	M	MCP1 from [5]	3.1; 4.3.1.2	Yes__No__
SC 23	Does the IUT support incoming interface NSC value tracing?	M	MCP1 from [5]	3.1; 4.3.1.2	Yes__No__
SC 24	Does the IUT support interworking LSP label tracing?	M	MCP1 from [6]	3.1; 4.3.1.2	Yes__No__
Comments: the term ‘Support’ means the ability to perform all aspect of this capability.					

A.3.5 Trace Transit List Format (TTLF)

Item	**Item Description**	**Status**	**Condition for status**	**Reference**	**Support**
TTLF 1	Is the TTL information element encoded according to the format described in Section 3.1?	M		3.1	Yes__No__
TTLF 2	Does the IUT support a maximum TTL information element length of at least 1466 octets?	M		4.2.1	Yes__No__
TTLF 3	Does the IUT support a maximum TTL information element length that is larger than 1466 octets?	O		4.2.1	Yes__No__
TTLF 4	If the IUT supports a TTL information element that is greater than 1466 octets, what is the maximum size of TTL information element supported?	M	TTLF 3	4.2.1	
TTLF 5	Does the IUT add at most one of each octet group into the TTL, in order of ascending octet group number, each time it processes a message, with the following exceptions:	M		3.1 Note 1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	• Octet groups 18, 19, and 22, if present, appear after any octet groups 7 through 11 and before octet group 12. • Octet groups 20 and 21, if present, appear immediately following octet group 12. • A node may add multiple instances of octet group 17. When octet group 17 is added , it must immediately follow either • Octet 6, • An octet group 12, unless octet group 12 is immediately followed by octet group 20 or 21, • An octet group 20, unless octet group 20 is immediately followed by octet group 21, • An octet group 21, • An octet group 16, or • Another octet group 17~~6~~. • A node attempting alternate routing may add instances of octet groups 12 20, 21, 13, 14, 15, 16, and~~through~~ 17, in the proper order, after an octet group 16 or octet 15. • The trace destination node may only add instances of octet groups 8 through 11 or 22 in order (according to the procedures specified in 4.3.1.2), after octet group 12, 20, 21, or one or more octet groups 16~~17~~ that themselves follow octet group 12, 20, 21, in addition to any instances of octet groups 8 through 11 or 22 (according to the procedures specified in 4.3.1.2) that it adds prior to octet group 12 according to the normal sequence.				
TTLF 6	Is the IUT when acting as the trace source node capable of interpreting the encoding of the DLCI (octet group 9), even if it doesn't support Frame Relay?	M	SC 8	3.1; 4.3.1.2	Yes__No__
TTLF 7	Does the IUT support the addition of multiple NSCs in each of the NSC lists (octet groups 18, 19, 20, 21)?	M	SC 21, SC 22, SC 23	3.1; 4.3.1.2	Yes__No__
Comments					

A.3.6 Procedures and Errors for Path Trace (PEPT)

Item	Item Description	Status	Condition for status	Reference	Support
PEPT 1	Is the IUT when acting as the trace source node capable of setting the C, V and A flags to '0' and '1'?	M		4.3.1.1	Yes__No__
PEPT 2	Does the IUT when acting as the trace	M		4.3.1.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	source node set the V flag and the A flag to zero if the X flag is set to 1?				
PEPT 3	Does the IUT set the V flag to zero when tracing Connection Oriented Bearer Independent (CO-BI) calls?	M	SC 13	4.3.1	Yes__No__
PEPT 4	In the subsequent questions in this section, does the term SETUP message apply to both SETUP and CO-BI SET UP messages except incases where the V flag is set?	M	SC 13	4.3.1	Yes__No__
PEPT 5	After adding the flags field to the TTL information element ,Does the IUT when acting as the trace source node add the Trace Status field with a value set to “trace in progress”?	M		4.3.1.1	Yes__No__
PEPT 6	If a Trace transit list information element is present in the received setup or add-party indication, Does the IUT continue the received trace (following the procedures of Section 4.3.1.2), ignoring any network specific policy that triggers a new path trace (i.e. this node shall not act as a trace source node)?	M		4.3.1.1	Yes__No__
PEPT 7	Whenever a SETUP or ADD PARTY message is received containing a Trace transit list information element with the Trace Status field not set to “trace in progress”, does the IUT remove the TTL information element from the outgoing SETUP or ADD PARTY message and save it locally as Saved original TTL and the Saved Modified TTL?	M		4.3.1.2	Yes__No__
PEPT 8	If the Trace status field is not set to “trace in progress” and the X flag is set to ‘1’, does the IUT • initiate call clearing using normal PNNI procedures with cause code set to #31 “normal, unspecified.”? • Copy the TTL information element from the SETUP/ADD PARTY message into the RELEASE /RELEASE COMPLETE/ADD PARTY REJECT message before progressing the message towards the trace source node?	M		4.3.1.2	Yes__No__
PEPT 9	Does the IUT when acting as the trace source node add a Originating logical port identifier entry into the TTL information element with the value either set to zero(if the trace source interface is not a PNNI interface and no logical port ID is assigned to this interface) or identifying the trace source interface from which the path trace is initiated?	M		4.3.1.2	Yes__No__
PEPT 10	If the V flag is set to ‘1’ and the trace source interface is an ATM interface, does the IUT when acting as the trace source node add a VPI/VCI entry for the trace source	M	SC 6	4.3.1.2	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	interface?				
PEPT 11	If the V flag is set to 1 and the trace source interface is a FR interface and 1:1 mapping is used between the Frame Relay and ATM connections, does the IUT when acting as the trace source node add a DLCI entry for the trace source interface?	M	SC 7	4.3.1.2	Yes__No__
PEPT 12	If the V flag is set to 1 and the trace source interface is neither an ATM interface nor a FR interface (e.g. circuit emulation service), does the IUT when acting as the trace source node add a VPI/VCI entry for the trace source interface?	O	SC6	4.3.1.2	Yes__No__
PEPT13	If the V flag is set to '1' and the IUT is not the trace source node, does the IUT add a VPI/VCI entry for the incoming interface on which the SETUP/ADD PARTY message was received?	M	SC 6	4.3.1.2	Yes__No__
PEPT14	If the A flag is set to 1, the connection leg on the trace source interface is switched and the trace source interface supports a Call reference that can be specified in 3 octets, does the IUT when acting as the trace source node add a Call Reference entry for the trace source interface?	M	SC 9	4.3.1.2	Yes__No__
PEPT15	If the A flag is set to 1 and the IUT is not the trace source node, does the IUT add a Call Reference entry for the incoming interface on which the SETUP/CO-BI SETUP/ADD PARTY was received?	M	SC 9	4.3.1.2	Yes__No__
PEPT 16	If the A flag is set to 1, and the IUT is not the trace source node, does the IUT also include the endpoint reference value for point-to-multipoint connections for the interface on which the SETUP/ADD PARTY was received?	M	SC 10; SC 2	4.3.1.2	Yes__No__
PEPT 17	If the IUT refuses to participate in path traces received over the incoming interface, does the IUT add a Logical node/Logical port entry containing its lowest level logical node ID and a logical port identifier value set to zero?	M		4.3.1.2	Yes__No__
PEPT 18	If the IUT refuses to participate in path traces received over the incoming interface, does the IUT add the Trace Continuation refusal indicator to the TTL information element ?	M	SC 16	4.3.1.2	Yes__No__
PEPT 19	If the IUT refuses to participate in path traces received over the incoming interface and X flag is set to zero, does the IUT remove the TTL information element before progressing the SETUP or ADD PARTY message towards the called party?	M		4.3.1.2	Yes__No__
PEPT 20	If the IUT refuses to participate in path traces received over the incoming interface and the X flag is set to 1, does the IUT: ▪ initiate normal PNNI call clearing procedures with cause code in the cause information element set to cause #31	M		4.3.1.2	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	"normal, unspecified"? ▪ copy the TTL information element from the SETUP or ADD PARTY message into the RELEASE, RELEASE COMPLETE or ADD PARTY REJECT message before progressing the message towards the trace source node ? ▪ The Trace status shall be set to "trace incomplete"?				
PEPT 21	Does the IUT add a Logical node/logical port entry where the logical node ID specifies the lowest level logical node and logical port specifies either a) the outgoing interface on which the SETUP/ADD PARTY message will be progressed if the IUT is not the trace destination node? or b) zero or the next link that would be traversed by the connection/party if the IUT is the trace destination node ?	M		4.3.1.2	Yes__No__
PEPT 22	If the IUT is the trace destination node, the V flag is set to 1 and the trace destination interface is an ATM interface, does the IUT add a VPI/VCI entry for the trace destination interface?	M	SC 6	4.3.1.2	Yes__No__
PEPT 23	If the IUT is the trace destination node, the V flag is set to 1 and the trace destination interface is a FR interface and 1:1 mapping is used between the Frame Relay and ATM connection, does the IUT add a DLCI entry for the trace destination interface?	M	SC 7	4.3.1.2	Yes__No__
PEPT 24	If the IUT is the trace destination node, the V flag is set to 1 and the terminating interface is neither an ATM interface nor FR interface (e.g. circuit emulation service), does the IUT add a VPI/VCI entry for the terminating interface?	O	SC6	4.3.1.2	Yes__No__
PEPT 25	If the IUT is the trace destination node, the A flag is set to 1, the connection leg on the trace destination interface is switched and the trace destination interface supports a call reference that can be specified in 3 octets, does the IUT add a Call Reference entry of the trace destination interface, including the endpoint reference if the connection is point-to-multipoint?	M	SC 9	4.3.1.2	Yes__No__
PEPT 26	If a SETUP/ADD PARTY message is received with the X flag in the TTL information element set to zero, does the IUT, when acting as trace destination node, remove the TTL information element before progressing the SETUP/ADD PARTY message towards the called party?	M		4.3.1.2	Yes__No__
PEPT 27	If the IUT is the trace destination node, the X flag is set to 1,and according to local information, there are insufficient resources to accept the connection/party on either the incoming or any interface over which the	M	SC 5	4.3.2	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	connection/party may be progressed, does the IUT perform call clearing procedures as specified in section 4.3.3?				
PEPT 28	If the IUT is the trace destination node, the X flag is set to 1,and the IUT does not have reachability to the next transit identified in the DTL stack (if this is not the DTL Terminator) or to the called party number or specified transit network (if this is the DTL Terminator), does the IUT perform call clearing procedures as specified in section 4.3.3?	M	SC 5	4.3.2	Yes__No__
PEPT 29	If the X flag is set to 1 and after the IUT applied the procedures of section 4.3.1.2 , does the IUT when acting as the trace destination node • copy the Saved Modified TTL information element into a RELEASE/RELEASE COMPLETE/ADD PARTY REJECT message? • Set the Trace status to "trace completed normally", if the trace status was set to "trace in progress"?	M	SC 5	4.3.2	Yes__No__
PEPT 30	If the X flag is set to 1 and after the node applied the procedures of section 4.3.1.2, does the IUT when acting as the trace destination node set the cause code to #31 "normal, unspecified" and follow standard PNNI procedures for call clearing?	M	SC 5	4.3.2	Yes__No__
PEPT 31	If the C flag in the Saved Original TTL information element is set to zero, does the IUT follow standard PNNI call clearing and crankback procedures, in case of error or failure?	M	SC 4a	4.3.3	Yes__No__
PEPT 32	If the C flag in the Saved original TTL is set to zero, does the IUT - initiate call clearing by sending a RELEASE/RELEASE COMPLETE/ADD PARTY REJECT message without the TTL information element? - When acting as a trace source node follow the procedures of section (4.3.4.4)?	M	SC 4a	4.3.3	Yes__No__
PEPT 33	If the C flag in the Saved Original TTL is set to 1 and the trace status field in the Saved Modified TTL is other than "trace in progress", does the IUT - copy the Saved Modified TTL information element into the RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message? - When acting as a trace source node follow the procedures in section (4.3.4.5)?	M	SC4	4.3.3	Yes__No__
PEPT 34	If the trace status field in the Saved Modified TTL information element is set to "trace in progress", does the IUT place the	M	SC4	4.3.3	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	Saved Modified TTL information element into the RELEASE, RELEASE COMPLETE OR ADD PART REJECT message generated by the node toward the calling party and - if this node did not refuse to participate in this path trace and no logical node/logical port entry has been added to the TTL information element for this node during the processing of the current message, does the IUT add one logical node/logical port entry containing the logical node ID of the lowest level node and the logical port set to zero? - If this node is the DTL originator or the call/party is cleared without crankback, does the IUT set the Trace status field in the TTL information element to "trace incomplete" and no crankback field shall be added to the TTL information element? - If the call/party is cleared with crankback and this node is not the DTL originator, does the IUT add a crankback field to the TTL information element with the appropriate crankback cause and Blocked Transit Type values from the generated cranckback information element? - If this is the trace source node, does the IUT apply the procedures Section (4.3.4)?				
PEPT 35	If the call clearing message is being sent towards the called party or if a CONNECT or ADD PARTY ACKNOWLEDGE message has already been sent toward the calling party, does IUT ignore the procedures in section (4.3.4)?	M		4.3.4	Yes__No__
PEPT 36	If the received RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message does not contain a Crankback information element, the message contains TTL information element, the IUT refused to participate in the path trace and the C flag in the Saved Original TTL information element is set to '1', does the IUT replace the TTL information element in the received call clearing message by the Saved Modified TTL information element in the received call clearing message?	M		4.3.4.1	Yes__No__
PEPT 37	If the received RELEASE, RELEASE COMPLETE,DROP PARTY or ADD PARTY REJECT message does not contain a Crankback information element, the message contains TTL information element, the IUT refused to participate in the path trace and the C flag in the Saved Original TTL information element is set to '0', does	M		4.3.4.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	the IUT remove the TTL information element from the call clearing message?				
PEPT 38	If the received RELEASE, RELEASE COMPLETE, DROP PARTY OR ADD PARTY REJECT message does not contain a Crankback information element, the message contains TTL information element and the IUT is not a trace source node, does the IUT pass the TTL information element in the call clearing message?	M		4.3.4.1	Yes__No__
PEPT 39	If the received RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message does not contain a Crankback information element, the message contains TTL information element and the IUT is a trace source node, does the IUT follow the procedures in section 4.3.4.5?	M		4.3.4.1	Yes__No__
PEPT 40	If the received RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message does not contain a Crankback information element, the message does not contain TTL information element and the C flag in the Saved Original TTL information element is set to ‘1’, does the IUT - include the Saved Modified TTL information element in the call clearing message? - Set the Trace status field to “trace incomplete”, if the field is set to “trace in progress”?	M		4.3.4.1	Yes__No__
PEPT 41	If the received RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message does not contain a Crankback information element, the message does not contain TTL information element and the C flag in the Saved Original TTL information element is set to ‘0’, does the IUT - follow normal call clearing procedures? - not include TTL information element in the call clearing message?	M		4.3.4.1	Yes__No__
PEPT 42	If the received RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message does not contain a Crankback information element, the message does not contain TTL information element and the IUT is acting as a trace source node, does the IUT follow the procedures in section 4.3.4.5?	M		4.3.4.1	Yes__No__
PEPT 43	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not “call or party has been blocked at the succeeding end of this interface”, and	M	SC4	4.3.4.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	- the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message has a TTL information element, does the IUT act as if no TTL information element was received?				
PEPT 44	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not "call or party has been blocked at the succeeding end of this interface", and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message has a TTL information element, the C flag in the Saved Original TTL information element is set to '1' and the IUT is acting as a trace source node, does the IUT follow the procedures of section 4.3.4.5?	M	SC4	4.3.4.1	Yes__No__
PEPT 45	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not "call or party has been blocked at the succeeding end of this interface", and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and if the IUT refused to participate in the path trace or it is acting as a trace source destination node, there is a TTL information element in the call clearing message, does the IUT delete the TTL information element?	M	SC4, SC16	4.3.4.1	Yes__No__
PEPT 46	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not "call or party has been blocked at the succeeding end of this interface", and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message has a TTL information element, the C flag in the Saved Original TTL information element is set to '1' and the IUT is not the trace source node,	M	SC4	4.3.4.1	Yes__No__

Item	**Item Description**	**Status**	**Condition for status**	**Reference**	**Support**
	does the IUT pass the received TTL information element toward the trace source node in the call clearing message?				
PEPT 47	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not “call or party has been blocked at the succeeding end of this interface”, and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message has a TTL information element and the C flag in the Saved Original TTL information element is set to ‘0’, does the IUT remove the TTL information element from the call clearing message?	M	SC4	4.3.4.1	Yes__No__
PEPT 48	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not “call or party has been blocked at the succeeding end of this interface”, and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message does not contain a TTL information element and the C flag in the Saved Original TTL information element is set to ‘0’, does the IUT - follow normal call clearing procedures? - no TTL information element shall be included in the call clearing message?	M	SC4	4.3.4.1	Yes__No__
PEPT 49	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not “call or party has been blocked at the succeeding end of this interface”, and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message does not contain a TTL information , the C flag in the Saved Original TTL information element is set to ‘0’ , does the IUT apply normal call clearing procedures and no TTL information element shall be included in the call clearing message?	M	SC4	4.3.4.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
PEPT 50	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not "call or party has been blocked at the succeeding end of this interface", and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message does not contain a TTL information element and the C flag in the Saved Original TTL information element is set to '1', does the IUT include the Saved Modified TTL information element in the call clearing message?	M	SC4	4.3.4.1	Yes__No__
PEPT 51	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - the Blocked transit type is not "call or party has been blocked at the succeeding end of this interface", and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message does not contain a TTL information element, the C flag in the Saved Original TTL information element is set to '1' and the Trace status in the Saved Modified TTL information element is "trace in progress", does the IUT append a Crankback gap indicator in the TTL information element followed by crankback field with the appropriate Crankback cause, Blocked Transit type and Blocked Transit Trace information values from the received Crankback information element?	M	SC4	4.3.4.1	Yes__No__
PEPT 52	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and - The Blocked transit type is not "call or party has been blocked at the succeeding end of this interface", and - the IUT is not DTL originator, and - the IUT is not an entry border node that generated DTLs for this call of equal or higher level than the crankback level, and the received message does not contain a TTL information element, the C flag in the Saved Original TTL information element is set to '1', and the Trace status in the Saved Modified TTL information element is not "trace in progress", does the IUT perform no	M	SC4	4.3.4.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	further processing of the TTL information element?				
PEPT 53	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and if either - The Blocked transit type is "call or party has been blocked at the succeeding end of this interface", or - the IUT is a DTL originator, or - the IUT is an entry border node that generated DTLs for this call of equal or higher level than the crankback level, if the IUT refuses to participate in the path trace, does the IUT follow the procedures of section 4.3.4.2 and there is no current TTL information element?	M	SC4	4.3.4.1	Yes__No__
PEPT 54	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and if either - The Blocked transit type is "call or party has been blocked at the succeeding end of this interface", or - the IUT is a DTL originator, or - the IUT is an entry border node that generated DTLs for this call of equal or higher level than the crankback level, if the C flag in the Saved Original TTL information element is set to '0', does the IUT follow the procedures in section 4.3.4.2 with the current TTL determined as follows: - if the trace status in the Saved Original TTL information element is set to "trace in progress", use the Saved Original TTL information as the Current TTL information element ? if the Trace Status field in the Saved Original TTL is not set to "trace in progress" have no Current TTL information element?	M	SC4	4.3.4.1	Yes__No__
PEPT 55	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and if either - The Blocked transit type is "call or party has been blocked at the succeeding end of this interface", or - the IUT is a DTL originator, or - the IUT is an entry border node that generated DTLs for this call of equal or higher level than the crankback level, if the C flag in the Saved Original TTL information element is set to '1',if the IUT is not the trace destination node and if a TTL information element is present in the received RELEASE, RELEASE	M	SC4	4.3.4.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	COMPLETE or ADD PARTY REJECT message, does the IUT - save the received TTL information element as the Saved Modified TTL information element? - use the received TTL as a current TTL information element in the procedures of section 4.3.4.2, if the Trace Status field is in "trace in progress"? use no current TTL information element in the procedures of section 4.3.4.2, if the trace status field is not set to "trace in progress"?				
PEPT 56	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and if either - The Blocked transit type is "call or party has been blocked at the succeeding end of this interface", or - the IUT is a DTL originator, or - the IUT is an entry border node that generated DTLs for this call of equal or higher level than the crankback level, if the C flag in the Saved Original TTL information element is set to '1' , and if the Trace status in the Saved modified TTL information element is "trace in progress" and if either this is the trace destination node or no TTL information element is present in the received RELEASE, RELEASE COMPLETE or ADD PARTY REJECT message, does the IUT append a Crankback field indicator to the Saved Modified TTL information element followed by a Crankback field with the appropriate Crankback cause, Blocked Transit Type, and Blocked Transit Trace information element values from the received Crankback information element and use the resulting Saved Modified TTL as the Current TTL information element in the procedures of section 4.3.4.2?	M	SC4	4.3.4.1	Yes__No__
PEPT 57	If the received RELEASE, RELEASE COMPLETE OR ADD PARTY REJECT message contains a Crankback information element and if either - The Blocked transit type is "call or party has been blocked at the succeeding end of this interface", or - the IUT is a DTL originator, or - the IUT is an entry border node that generated DTLs for this call of equal or higher level than the crankback level, if the C flag in the Saved Original TTL information element is set to '1', and if the Trace status in the Saved modified TTL information element is not "trace in progress" and if either this is the trace	M	SC4	4.3.4.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	destination node or no TTL information element is present in the received RELEASE, RELEASE COMPLETE or ADD PARTY REJECT message, does the IUT use no Current TTL information element in the procedures of section 4.3.4.2,?				
PEPT 58	If the IUT is attempting alternate routing and there is a current TTL information element, does the IUT copy the Current TTL information element into the new SETUP/ADD PARTY message, and follow the procedures of section 4.3.1.2 (commencing with step 6) ?	M		4.3.4.2	Yes__No__
PEPT 59	If the IUT does not attempt alternate routing or if alternate routing fails for any reason, does the IUT apply the procedures in section 4.3.3 (if they have not yet been applied as already)?	M		4.3.4.2	Yes__No__
PEPT 60	If a RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message is received that does not contain a crankback information element, does the IUT, when acting as a trace destination node: • copy the Saved Modified Trace transit list information element into the call clearing message? • set the trace status field in the TTL information element inserted into the call clearing message to "trace completed normally" if the trace status field in TTL information element is set to "trace in progress"? • commence normal call clearing procedures?	M	SC 4	4.3.4.3	Yes__No__
PEPT 61	If a RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message is received over a PNNI interface that does contain a crankback information element, does the IUT, when acting as a trace destination node, follow the procedures of sections 4.3.4.1 or 4.3.4.3 ?	M	SC 4	4.3.4.3	Yes__No__
PEPT 62	If a RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message is received over a non PNNI interface that does contain a crankback information element, does the IUT: - add crankback received at trace destination node indicator to the Saved Modified TTL information element? - apply the procedures of section 4.3.4.2 with the current TTL information element set to Saved Modified TTL information element?	M	SC 4	4.3.4.3	Yes__No__
PEPT 63	Does the IUT when acting as a Trace source node remove the TTL information element and change the Trace status field to "trace in complete" if the field is set to "trace in	M		4.3.4.4	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	progress" before progressing a RELEASE, RELEASE COMPLETE, DROP PARTY or ADD PARTY REJECT message towards the calling party if the message contains the IE ?				
PEPT 64	If the received CONNECT/ADD PARTY ACKNOWLEDGE does not contain TTL information element or the IUT refuse to participate in path trace, does the IUT: ▪ include the Saved Modified TTL information element in the CONNECT/ADD PARTY/ ACKNOWLEDGE message, ▪ if the Trace status field in the message was set to 'trace in progress' ,set the Trace Status field to "trace incomplete"?	M		4.3.5.1	Yes__No__
PEPT 65	If the received CONNECT/ADD PARTY ACKNOWLEDGE contains a TTL information element and the IUT did not refuse to participate in path tracing, does the IUT that is not acting as a Trace source node propagate the TTL information element unmodified?	M		4.3.5.1	Yes__No__
PEPT 66	Does the IUT, when acting as the trace destination node, add the Saved Modified TTL information element to the CONNECT or ADD PARTY ACKNOWLEDGE message?	M		4.3.5.2	Yes__No__
PEPT 67	Does the IUT, when acting as the trace destination node, set the trace status field to "Trace completed normally" in the TTL information element within the CONNECT or ADD PARTY ACKNOWLEDGE, message if the trace status field was set to "trace in progress" ?	M		4.3.5.2	Yes__No__
PEPT 68	Does the IUT, when acting as the trace source node, remove the TTL information element before progressing a CONNECT or ADD PARTY ACKNOWLEDGE message towards the calling party and if the Trace status field is set to "trace in progress", then does the IUT change it to "trace incomplete"?	M		4.3.5.3	Yes__No
PEPT 69	If the length of the TTL information element exceeds the maximum length or the maximum message length is exceeded at any point during message processing (not necessarily during processing of the trace transit list information element but, for example, while adding Designated Transit List information elements), does the IUT set the Trace Status field to "trace has exceeded information element length limitations" or "trace has exceeded message length limitations" as appropriate?	M		4.3.7	Yes__No__
PEPT 70	If the length of the TTL information element exceeds the maximum length at any point during tracing or the maximum message length is exceeded at any point	M	SC 5	4.3.7	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	during message processing and the X flag is set to 1, does the IUT release the connection/party by sending a RELEASE, RELEASE COMPLETE, or ADD PARTY REJECT message with cause code in the Cause information element set to cause #31 "normal, unspecified" including the TTL information element?				
PEPT 71	If the length of the TTL information element exceeds the maximum length at any point during tracing or the maximum message length is exceeded at any point during message processing and the X flag is set to zero, does the IUT • save the TTL information element locally as Saved Modified Trace transit list and remove the TTL information element, save the current TTL as the Saved Original TTL, if the TTL information element has not yet been saved as the Saved Original TTL?	M	SC 5	4.3.7	Yes__No__
PEPT 72	Is the IUT when acting as the trace source node capable of setting the Ne, Rp, and I flags to '0' and '1'?	M	SC 21, SC 22, SC 23	4.3.1.1	Yes__No__
PEPT 73	If the Ne flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface, the IUT is the trace source node and the incoming interface is tagged by Ne-NSCs, does the IUT add the list of Ne-NSCs tagging the interface?	M	SC 21	4.3.1.2	Yes__No__
PEPT 74	If the Ne flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface, the IUT is not the trace source node, the I flag is set to 1, and the incoming interface is tagged by Ne-NSCs, does the IUT add the list of Ne-NSCs tagging the interface?	M	SC 21, SC 23	4.3.1.2	Yes__No__
PEPT 75	If the Rp flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface, the IUT is the trace source node, and incoming interface is tagged by Rp-NSCs, does the IUT add the list of Rp-NSCs tagging the interface?	M	SC 22	4.3.1.2	Yes__No__
PEPT 76	If the Rp flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface, the IUT is the trace source node, and the incoming interface is not tagged by any Rp-NSCs, does the IUT add the "Rp-NSC Bare" value?	M	SC 22	4.3.1.2	Yes__No__
PEPT 77	If the Rp flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface,	M	SC 22	4.3.1.2	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	the IUT is the trace source node, and the incoming interface is tagged by all Rp-NSCs, does the IUT add the "Rp-NSC Bare" value?				
PEPT 78	If the Rp flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface, the IUT is not the trace source node, the I flag is set to 1, and the incoming interface is tagged by Rp-NSCs, does the IUT add the list of Rp-NSCs tagging the interface?	M	SC 22, SC 23	4.3.1.2	Yes No
PEPT 79	If the Rp flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface, the IUT is not the trace source node, the I flag is set to 1, and the incoming interface is not tagged by any Rp-NSCs, does the IUT add the "Rp-NSC Bare" value?	M	SC 22, SC 23	4.3.1.2	Yes No
PEPT 80	If the Rp flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface, the IUT is not the trace source node, the I flag is set to 1, and the incoming interface is tagged by all Rp-NSCs, does the IUT add the "Rp-NSC Bare" value?	M	SC 22, SC 23	4.3.1.2	Yes No
PEPT 81	If the Ne flag is set to 1 and the outgoing interface is tagged by Ne-NSCs, does the IUT add the list of Ne-NSCs tagging the interface?	M	SC 21	4.3.1.2	Yes No
PEPT 82	If the Rp flag is set to 1 and the outgoing interface is tagged by Rp-NSCs, does the IUT add the list of Rp-NSCs tagging the interface?	M	SC 22	4.3.1.2	Yes No
PEPT 83	If the Rp flag is set to 1 and the outgoing interface is not tagged by Rp-NSCs, does the IUT add "Rp-NSC Bare"?	M	SC 22	4.3.1.2	Yes No
PEPT 84	If the Rp flag is set to 1 and the outgoing interface is tagged by all Rp-NSCs, does the IUT add "Rp-NSC Bare"?	M	SC 22	4.3.1.2	Yes No
PEPT 85	Is the IUT when acting as the trace source node capable of setting the L flag to '0' and '1'?	M	SC 24	4.3.1.1	Yes No
PEPT 86	If the L flag is set to 1, the IUT does not refuse to participate in path traces received over the incoming interface, and the incoming interface on which the SETUP or ADD PARTY message was received is an ATM-MPLS network interworking interface, does the IUT	M	SC 24	4.3.1.2	Yes No

Item	Item Description	Status	Condition for status	Reference	Support
	add the labels of the interworking LSPs to be used for the connection?				
PEPT 87	If the IUT is the trace destination node, the L flag is set to 1, and the trace destination interface is an ATM-MPLS network interworking interface, does the IUT add the labels of the interworking LSPs to be used for the connection?	M	SC 24	4.3.1.2	Yes__No__
Comments					

A.3.7 General Errors and Compatibility (GEC)

Item Number	Item Description	Status	Condition for status	Reference	Support
GEC 1	If the X flag is set to zero, does the IUT when acting as the trace source node set the information element instruction field flag in the trace transit list information element to “follow explicit instructions” and the information element IE action indicator to “discard information element and proceed”?	M		4.4	Yes__No__
GEC 2	If the X flag is set to 1, does the IUT when acting as the trace source node set the IE instruction field flag in the trace transit list information element to “follow explicit instructions” and the IE action indicator to “clear call”?	M		4.4	Yes__No__
Comments					

A.3.8 Supported Messages (Message structure) (MS)

Item	Item Description	Status	Condition for status	Reference	Support
MS 1	Does the IUT support the Trace transit list information element in the CONNECT message?	M		4.2.1	Yes__No__
MS 2	Does the IUT support the Trace transit list information element in the RELEASE message?	M		4.2.2	Yes__No__
MS 3	Does the IUT support the Trace transit list information element in the RELEASE COMPLETE message?	M		4.2.3	Yes__No__
MS 4	Does the IUT support the Trace transit list information element in the SETUP message?	M		4.2.4	Yes__No__
MS 5	Does the IUT support the Trace transit list information element in the ADD PARTY message?	M		4.2.5	Yes__No__
MS 6	Does the IUT support the Trace transit list information element in the ADD PARTY ACKNOWLEDGE message?	M		4.2.6	Yes__No__
MS 7	Does the IUT support the Trace transit list information element in the ADD PARTY REJECT message?	M		4.2.7	Yes__No__
MS 8	Does the IUT support the Trace transit list	M	MC2.3	4.2.9	Yes__No__

	information element in the CO-BI SETUP message?		from [~~GFS~~4]		
MS 9	Does the IUT support the Trace transit list information element in the Drop party message?	M		4.2.2	Yes__No__
Comments					

Annex B Protocol Implementation Conformance Statement (PICS) for PNNI 1.0 Connection Trace

[NORMATIVE]

B.1 Introduction

To evaluate conformance of a particular implementation, it is necessary to have a statement of which capabilities and options have been implemented. Such a statement is called a Protocol Implementation Conformance Statement (PICS).

B.1.1 Scope

This document provides the PICS proforma for the Addendum to PNNI 1.0 for the support of Path and Connection Trace, as specified in this document in compliance with the relevant requirements, and in accordance with the relevant guidelines, given in ISO/IEC 9646-2 [CTMI]. In most cases, statements contained in notes in the specification, which were intended as information, are not included in the PICS.

B.1.2 Normative References

[1] ISO/IEC 9646-1: 1994, Information technology – Open systems interconnection – Conformance testing methodology and framework – Part 1: General Concepts (See also ITU Recommendation X.290 (1995)).

[2] ISO/IEC 9646-2:1994, Information technology – Open systems interconnection – Conformance testing methodology and interconnection – Part 2: Abstract test suite specification (See also ITU telecommunication X.291 (1995)).

[3] PNNI Addendum for Path and Connection Trace V1.1, The ATM Forum Technical Committee, af-cs-0141.002, Octobert 2003

~~[4] Private Network to Network Interface V1.0, The ATM Forum Technical Committee, af-pnni-0055.000~~

~~[5] Private Network to Network Interface V 1.0, The ATM Forum Technical Committee, af-pnni-0081.000~~

[~~GFS~~4] PNNI Addendum on PNNI/B-QSIG Interworking and Generic Functional Protocol for Support of Supplementary Service, AF-CS-0102.000, October 1998.

[5] Policy Routing Version 1.0, The ATM Forum Technical Committee, af-cs-0195.000, April 2003

[6] ATM-MPLS Network Interworking Signalling Specification, Version 1.0, The ATM Forum Technical Committee, af-cs-0197.000, August 2003

B.1.3 Definitions

This following terms defined in ISO/IEC 9646-1[CTMF] are used in this document:

- A Protocol Implementation Conformance Statement (PICS) is a statement made by the supplier of an implementation or system, stating which capabilities have been implemented for a given protocol.
- A PICS proforma is a document, in the form of a questionnaire, designed by the protocol specifier or conformance test suite specifier, which when completed for an implementation or system becomes the PICS.

B.1.4 Acronyms

I.E.	Information Element
IUT	Implementation under test
M	Mandatory requirements (these are to be observed in all cases)
N/A	Not supported, not applicable, or the conditions for status are not met.
O	Optional (may be selected to suit the implementation, provided that any requirements applicable to the options are observed)
O.n	Optional, but support is required for either at least one or only one of the options in the group labelled with the same numeral "n".
PICS	Protocol Implementation Conformance Statement
SUT	System under test

B.1.5 Conformance

The supplier of a protocol implementation which is claimed to conform to the ATM Forum PNNI signalling Addendum for the support of Connection Trace is required to complete a copy of the PICS proforma provided in this document and is required to provide the information necessary to identify both the supplier and the implementation.

B.2 Identification of the Implementation

Implementation Under Test (IUT) Identification

IUT Name:____

IUT Version: ____

____

System Under Test (SUT) Identification

SUT Name:____

Hardware Configuration:
____

____

____

Operating System: ____

Product Supplier

Name:____

Address:____

____

Telephone Number: ____

Facsimile Number: ____

Email Address:____

Additional Information: ____

Client

Name:____

Address:
____

____

Telephone Number: ____

Facsimile Number: ____

Email Address:____

Additional Information: ____

PICS Contact Person

Name:____

Address: ____

____

Telephone Number: ____

Facsimile Number: ____

Email Address:____

Additional Information: ____

PICS/System Conformance Statement

Provide the relationship of the PICS with the System Conformance Statement for the system:
____
____
____

Identification of the protocol

This PICS proforma applies to the following:
The sections pertaining to Connection Trace in the *PNNI Addendum for Path and Connection Trace Version 1.0*.

B.3 PICS Proforma

B.3.1 Global statement of conformance

The implementation described in this PICS meets all of the mandatory requirements of the reference protocol.

[ ] YES
[ ] NO

Note: Answering "No" indicates non-conformance to the specified protocol. Non-supported mandatory capabilities are to be identified in the following tables, with an explanation by the implementor explaining why the implementation is non-conforming.

B.3.2 Instructions for Completing the PICS Proforma

The PICS Proforma is a fixed-format questionnaire. Answers to the questionnaire should be provided in the rightmost columns, either by simply indicating a restricted choice (such as Yes or No), or by entering a value or a set of range of values.

A supplier may also provide additional information, categorised as exceptional or supplementary information. These additional information should be provided as items labelled X.<i> for exceptional information, or S.<i> for supplemental information, respectively, for cross reference purposes, where <i> is any unambiguous identification for the item. The exception and supplementary information are not mandatory and the PICS is complete without such information. The presence of optional supplementary or exception information should not affect test execution, and will in no way affect interoperability verification. The column labelled 'Reference' gives a pointer to sections of the protocol specification for which the PICS Proforma is being written.

B.3.3 Major Capability (MC)

Item Number	Item Description	Status	Condition for status	Reference	Support
MC 1	Does the IUT support connection trace?	M		5	Yes__No__

B.3.4 Subsidiary Capabilities (SC)

Item Number	Item Description	Status	Condition for status	Reference	Support
SC 1	Does the IUT support connection trace for point-to-point connections?	M		5	Yes__No__
SC 2	Does the IUT support connection trace for point-to-multipoint connections?	M		5	Yes__No__
SC 3	Does the IUT support all trace status codes?	M		3.1	Yes__No__
SC 4	Does the IUT support VPI/VCI tracing?	M		3.1; 5.3	Yes__No__
SC 5	Does the IUT support generation of DLCIs?	O		3.1; 5.3	Yes__No__
SC 6	Is the IUT capable of recognising and interpreting the encoding of DLCIs?	M		3.1; 5.3	Yes__No__
SC 7	Does the IUT support call reference value tracing?	M		3.1; 5.3	Yes__No__
SC 8	Does the IUT support endpoint reference value tracing?	M		3.1; 5.3	Yes__No__
SC 9	Does the IUT support insertion of Vendor	O		3.1	Yes__No__

	Specific Information?				
SC 10	Is the IUT capable of recognising the presence and length of the Vendor Specific octet group?	M		3.1	Yes__No__
SC 11	Does the IUT support the tracing of Connection Oriented Bearer Independent (CO-BI) connections?	M	MC2.3 from [~~GFS~~4]	4.3.1	Yes__No__
SC 12	Can the IUT be configured to set the pass a long request bit to "pass along request" in the TRACE CONNECTION message-?	M		5.4	Yes__No__
SC 13	Can the IUT be configured to set the pass along request bit to "pass along request" in the TRACE CONNECTION ACKNOWLEDGE message-?	M		5.4	Yes__No__
SC 14	Can the IUT be configured to set the pass along request bit to "pass along request" in the Trace Transit List information element?	M		5.4	Yes__No__
SC 15	Does the IUT support the refusal of trace of a call that contains a TTL information element?	O		3.1	Yes__No__
SC 16	Does the IUT support Ne-NSC value tracing?	M	MCP1 from [5]	3.1; 5.3.1	Yes__No__
SC 17	Does the IUT support Rp -NSC value tracing?	M	MCP1 from [5]	3.1; 5.3.1	Yes__No__
SC 18	Does the IUT support incoming interface NSC value tracing?	M	MCP1 from [5]	3.1; 5.3.1	Yes__No__
SC 19	Does the IUT support interworking LSP label tracing?	M	MCP1 from [6]	3.1; 5.3.1	Yes__No__
Comments:					

B.3.5 Trace Transit List Format (TTLF)

Item	**Item Description**	**Status**	**Condition for status**	**Reference**	**Support**
TTLF 1	Is the TTL information element encoded according to the format described in Section 3.1?	M		3.1	Yes__No__
TTLF 2	Does the IUT support a maximum TTL information element length of at least 1466 octets?	M		4.2.1	Yes__No__
TTLF 3	Does the IUT support a maximum TTL information element length that is larger than 1466 octets?	O		4.2.1	Yes__No__
TTLF 4	If the IUT supports a TTL information element that is greater than 1466 octets, what is the maximum size of TTL information element supported?	M	TTLF 3	4.2.1	
TTLF 5	Does the IUT add at most one of each octet group into the TTL information element, in order of ascending octet group number, each time it processes a message? with the following exceptions: • Octet groups 18, 19, and 22, if present, appear after any octet groups 7 through 11 and before octet group 12. • Octet groups 20 and 21, if present, appear immediately following	M		3.1 Note 1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	octet group 12. • Octet group 22, if present, appears after octet group 19 and before octet group 12. • A node may add multiple instances of octet group 17. When octet group 17 is added , it must immediately follow either • Octet 6, • An octet group 12, unless octet group 12 is immediately followed by octet group 20 or 21, • An octet group 20, unless octet group 20 is immediately followed by octet group 21, • An octet group 21, • An octet group 16, or • Another octet group 17. A node attempting alternate routing may add instances of octet groups 12, 20, 21, 13, 14, 15, 16, and ~~through~~ 17, in the proper order, after an octet group 16 or octet 15. • The trace destination node may only add instances of octet groups 8 through 11 or 22 in order (according to the procedures specified in 4.3.1.2), after octet group 12, 20, 21, or one or more octet groups 16 that themselves follow octet group 12, 20, 21, in addition to any instances of octet groups 8 through 11 or 22 (according to the procedures specified in 4.3.1.2) that it adds prior to octet group 12 according to the normal sequence.				
TTLF 6	Is the IUT when acting as the trace source node capable of interpreting the encoding of the DLCI (octet group 9), even if it doesn't support Frame Relay?	M	SC 6	3.1; 4.3.1.2	Yes__No__
TTLF 7	Is the IUT capable of interpreting the encoding of the Vendor Specific Information (octet group 16)?	M	SC10	3.1; 4.3.1.2	Yes__No__
TTLF 8	Does the IUT support the addition of multiple NSCs each of the NSC lists (octet groups 18, 19, 20, 21)?	M	SC 16, SC 17, SC 18	3.1,5.3.1	Yes__No__
Comments					

B.3.6 Messages for Connection Trace (MCT)

Item Number	Item Description	Status	Condition for status	Reference	Support
MCT 1	Does the IUT support the TRACE CONNECTION message?	M		5.2.1	Yes__No__

MCT 2	Does the IUT support the TRACE CONNECTION ACKNOWLEDGE message?	M		5.2.2	Yes__No__

B.3.7 Procedures and Errors for Connection Trace (PECT)

Item	**Item Description**	**Status**	**Condition for status**	**Reference**	**Support**
PECT 1	Does the IUT when acting as the trace source node initiate the connection trace by creating TRACE CONNECTION message including a TTL information element to be sent on the outgoing interface in the appropriate direction?	M		5.3.1	Yes__No__
PECT 2	Does the IUT when originating a TRACE CONNECTION message set the C and X flags to zero in the TTL information element and ignore it during processing?	M		5.3.1	Yes__No__
PECT 3	After adding the Flags field to the TTL information element , does the IUT when acting as a trace source node add the Trace status field with the value set to "trace in progress"	M		5.3.1	Yes__No__
PECT 4	Does the IUT ignore the V flag when tracing Bearer Independent (CO-BI) calls?	M	SC11	5.3.1	Yes__No__
PECT 5	When the IUT is not the trace source node, does the IUT not perform any content validation on those portions of the received TTL information element after the trace status field?	M		5.3.1	Yes__No__
PECT 6	If a TRACE CONNECTION message is received over an incoming interface: - if the connection or party is in the null state , does the IUT treat the TRACE CONNECTION message as an unexpected message and there procedures of section 6.5.6.3.2/PNNI 1.0 shall apply? if the connection or part is in a clearing state, does the IUT discard the TRACE CONNECTION ACKNOWLEDGE message and no further action will be taken?	M		5.3.1	Yes__No__
PECT 7	If a TRACE CONNECTION message is received over an incoming interface, the Trace status in the TTL information element is "trace in progress", the connection/party is not in the Active /null or a call/party clearing state on the incoming interface , does the IUT • set the Trace status to "trace incomplete; • copy the TTL information element from the TRACE CONNECTION message to the TRACE CONNECTION ACKNOWLEDGE message; • send it on the incoming interface?	M		5.3.1	Yes__No__
PECT 8	When the IUT is acting as the trace source node and the direction of the trace is	M		5.3.1	Yes__No__

Item	**Item Description**	**Status**	**Condition for status**	**Reference**	**Support**
	incoming from the trace source node, does the IUT add the Trace source port identifier field, with the value either set to zero (if this is not a PNNI interface and no logical port ID is assigned to this interface)or identifying the originating logical port from which the path trace is initiated?				
PECT 9	If the V flag is set to '1', the trace direction is specified as the incoming direction from the trace source interface and the trace source interface is an ATM interface, does the IUT when acting as the trace source node add a VPI/VCI entry for the trace source interface?	M	SC 4	5.3.1	Yes__No__
PECT 10	If the V flag is set to 1, the trace direction is specified as the incoming direction from the trace source interface, the trace source interface is a FR interface and 1:1 mapping is used between the Frame Relay and ATM connection,, does the IUT when acting as the trace source node add a DLCI entry for the trace source interface?	M	SC 5	5.3.1	Yes__No__
PECT 11	If the V flag is set to 1 the trace direction is specified as the incoming direction from the trace source interface and the trace source interface is neither an ATM interface nor a FR interface(e.g. Circuit emulation service), does the IUT when acting as the trace source node add a VPI/VCI entry for the trace source interface?	O	SC4	5.3.1	Yes__No__
PECT 12	If the V flag is set to '1' and the IUT is not the trace source node, does the IUT add a VPI/VCI entry for the interface on which the TRACE CONNECTION message was received?	M	SC 4	5.3.1	Yes__No__
PECT 13	If the A flag is set to 1, the trace direction is specified as the incoming direction from the trace source interface, the connection leg on the trace source interface is switched and the trace source interface supports a Call reference that can be specified in 3 octets, does the IUT when acting as the trace source node add a Call Reference entry for the trace source interface?	M	SC 7	5.3.1	Yes__No__
PECT 14	If the A flag is set to 1 and the IUT is not the trace source node, does the IUT add a Call Reference entry for the interface on which the TRACE CONNECTION was received?	M	SC 7	5.3.1	Yes__No__
PECT 15	If the A flag is set to 1, does the IUT also include the endpoint reference value for point-to-multipoint connections?	M	SC 8	5.3.1	Yes__No__
PECT 16	If the IUT refuses to participate in connection traces received over the incoming interface, does the IUT add a Logical node/Logical port entry including its lowest level logical node ID and the logical port set to all zeros?	M		5.3.1	Yes__No__
PECT 17	If the IUT refuses to participate in connection traces received over the	M		5.3.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	incoming interface, does the IUT insert the Trace continuation refusal indicator into the TTL information element and set the Trace status to "trace incomplete "?				
PECT 18	If the IUT refuses to participate in connection traces received over the incoming interface, does the IUT copy the modified TTL information element from the TRACE CONNECTION message into the TRACE CONNECTION ACKNOWLEDGE message and return this message on the incoming interface over which the TRACE CONNECTION message was received?	M		5.3.1	Yes__No__
PECT 19	Does the IUT add a logical node/logical port entry into the TTL information element, where the logical node specifies the lowest level logical node and the logical port specifies either a) the outgoing interface on which the SETUP/ADD PARTY message will be progressed if the IUT is not the trace destination node? or b) zero or the next link that would be traversed by the connection/party if the IUT is the trace destination node?	M		5.3.1	Yes__No__
PECT 20	If the connection is not in the Active call state or the party is not in the Active party state on the outgoing interface, does the IUT • set the Trace status to "trace incomplete"; • copy the TTL information element from the TRACE CONNECTION message to a TRACE CONNECTION ACKNOWLEDGE message? • send the message on the incoming interface? • Set to zero the logical port id entry added in step 6, If the outgoing interface is not known ?	M		5.3.1	Yes__No__
PECT 21	If the IUT is the trace destination node, the V flag is set to 1 and the trace destination interface is an ATM interface, does the IUT add a VPI/VCI entry for the trace destination interface?	M	SC 4	5.3.1	Yes__No__
PECT 22	If the IUT is the trace destination node, the V flag is set to 1 and the trace destination interface is a FR interface and a 1:1 mapping is used between the Frame Relay and the ATM connection, does the IUT add a DLCI entry for the trace destination interface?	M	SC 5	5.3.1	Yes__No__
PECT 23	If the IUT is the trace destination node, the V flag is set to 1 and the trace destination interface is neither an ATM interface nor a Frame Relay interface, does the IUT add a VPI/VCI entry for the trace destination interface?	O	SC4	5.3.1	Yes__No__
PECT 24	If the IUT is the trace destination node, the A flag is set to 1, the connection leg on the trace destination interface is switched and	M	SC 7	5.3.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	the trace destination interface supports a Call reference that can be specified in 3 octets, does the IUT add a Call Reference entry for the trace destination interface?				
PECT 25	If the A flag is set to 1, does the IUT when acting as the trace destination node also add a endpoint reference entry for point-to-multipoint connections?	M	SC 8	5.3.1	Yes__No__
PECT 26	For point to point connections, does the IUT determine the outgoing interface from the local connection table using the call reference value?	M		5.3.1	Yes__No__
PECT 27	For point to multipoint connections, does the IUT determine the outgoing interface from the local connection table using the call reference and endpoint reference values?	M		5.3.1	Yes__No__
PECT 28	Does the IUT when acting as the trace destination node send a TRACE CONNECTION ACKNOWLEDGE message on the incoming interface over which the TRACE CONNECTION message was received?	M		5.3.1	Yes__No__
PECT 29	Does the IUT when acting as the trace destination node copy the content of the TTL information element from the TRACE CONNECTION message into the TRACE CONNECTION ACKNOWLEDGE message, after adding all appropriate fields and setting the Trace Status field to “trace completed normally”?	M		5.3.1	Yes__No__
PECT 30	During the procedures specified in section 5.3.1, if the IUT can not progress the TRACE CONNECTION message towards the trace destination node due to conditions not stated in section 5.3.1, does the IUT • copy the TTL information element from the TRACE CONNECTION message to a TRACE CONNECTION ACKNOWLEDGE message; • if the Trace status is “trace in progress” set the Trace status in the TTL information element to “trace incomplete” ; send the TRACE CONNECTION ACKNOWLEDGE message on the incoming interface?	M		5.3.1	Yes__No__
PECT 31	If the IUT is other than the trace source node, does the IUT progress the received TRACE CONNECTION ACKNOWLEDGE message over the same incoming interface as the TRACE CONNECTION message was received?	M		5.3.2	Yes__No__
PECT 32	When the trace source node receives a TRACE CONNECTION ACKNOWLEDGE message and the first logical node id of the TTL information element corresponds to the trace source node’s logi cal node id, Does the IUT save the Trace transit list information element and terminates the	M		5.3.2	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	progression of the message?				
PECT 33	When the trace source node receives a TRACE CONNECTION ACKNOWLEDGE message and the first logical node id of the TTL information element does not correspond to the trace source node's id, Does the IUT not save the Trace transit list information element and the node shall not act as the trace source node?	M		5.3.2	Yes__No__
PECT 34	If the length of the TTL information element exceeds the maximum length or the maximum message length is exceeded at any point during TRACE CONNECTION message processing (not necessarily during processing of the trace transit list information element, does the IUT set the Trace Status field to "trace has exceeded information element length limitations" or "trace has exceeded message length limitations" as appropriate?	M		5.3.3	Yes__No__
PECT 35	If the length of the TTL information element exceeds the maximum length at any point during tracing or the maximum message length is exceeded at any point during TRACE CONNECTION message processing ,does the IUT copy the TTL information element from the TRACE CONNECTION message to a TRACE CONNECTION ACKNOWLEDGE message and send the TRACE CONNECTION ACKNOWLEDGE message on the incoming interface?	M		5.3.3	Yes__No__
PECT 36	Is the IUT when acting as the trace source node capable of setting the Ne, Rp, and I flags to '0' and '1'?	M	SC 16, SC 17, SC 18	5.3.1	Yes__No__
PECT 37	If the Ne flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, the IUT is the trace source node, and the incoming interface is tagged by Ne-NSCs, does the IUT add the list of Ne-NSCs tagging the interface?	M	SC 16	5.3.1	Yes__No__
PECT 38	If the Ne flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, the IUT is not the trace source node, the I flag is set to 1, and the incoming interface is tagged by Ne-NSCs, does the IUT add the list of Ne-NSCs tagging the interface?	M	SC 16, SC 18	5.3.1	Yes__No__
PECT 39	If the Rp flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, the IUT is the trace source node, and the incoming interface is tagged by Rp-NSCs, does the IUT add	M	SC17	5.3.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	the list of Rp-NSCs tagging the interface?				
PECT 40	If the Rp flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, the IUT is the trace source node, and the incoming interface is not tagged by any Rp-NSCs, does the IUT add the "Rp-NSC Bare" value?	M	SC17	5.3.1	Yes__No__
PECT 41	If the Rp flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, the IUT is the trace source node, and the incoming interface is tagged by all Rp-NSCs, does the IUT add the "Rp-NSC Bare" value?	M	SC17	5.3.1	Yes__No__
PECT 42	If the Rp flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, the IUT is not the trace source node, the I flag is set to 1, and the incoming interface is tagged by Rp-NSCs, does the IUT add the list of Rp-NSCs tagging the interface?	M	SC17, SC18	5.3.1	Yes__No__
PECT 43	If the Rp flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, the IUT is not the trace source node, the I flag is set to 1, and the incoming interface is not tagged by any Rp-NSCs, does the IUT add the "Rp-NSC Bare" value?	M	SC17, SC18	5.3.1	Yes__No__
PECT 44	If the Rp flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, the IUT is not the trace source node, the I flag is set to 1, and the incoming interface is tagged by all Rp-NSCs, does the IUT add the "Rp-NSC Bare" value?	M	SC17, SC18	5.3.1	Yes__No__
PECT 45	If the Ne flag is set to 1 and the outgoing interface is tagged by Ne-NSCs, does the IUT add the list of Ne-NSCs tagging the interface?	M	SC 16	5.3.1	Yes__No__
PECT 46	If the Rp flag is set to 1 and the outgoing interface is tagged by Rp-NSCs, does the IUT add the list of Rp-NSCs tagging the interface?	M	SC17	5.3.1	Yes__No__
PECT 47	If the Rp flag is set to 1 and the outgoing interface is not tagged by Rp-NSCs, does the IUT add "Rp-NSC Bare"?	M	SC17	5.3.1	Yes__No__
PECT 48	If the Rp flag is set to 1 and the	M	SC17	5.3.1	Yes__No__

Item	Item Description	Status	Condition for status	Reference	Support
	outgoing interface is tagged by all Rp-NSCs, does the IUT add "Rp-NSC Bare"?				
PECT 49	Is the IUT when acting as the trace source node capable of setting the L flag to '0' and '1'?	M	SC 19	5.3.1	Yes__No__
PECT 50	If the L flag is set to 1, the IUT does not refuse to participate in connection traces received over the incoming interface, and the incoming interface on which the TRACE CONNECTION message was received is an ATM-MPLS network interworking interface, does the IUT add the labels of the interworking LSPs to be used for the connection?	M	SC 19	5.3.1	Yes__No__
PECT 51	If the IUT is the trace destination node, the L flag is set to 1, and the trace destination interface is an ATM-MPLS network interworking interface, does the IUT add the labels of the interworking LSPs to be used for the connection	M	SC 19	5.3.1	Yes__No__
Comments					

B.3.8 Compatibility of Connection Trace (CCT)

Item Number	Item Description	Status	Condition for status	Reference	Support
CCT 1	Does the IUT when acting as the trace source node set the Message instruction field flag in the TRACE CONNECTION message to "follow explicit instructions" and the Message action indicator to either "discard and ignore" or "discard and report status"?	M		5.4	Yes__No__
CCT 2	Does the IUT when acting as the trace destination node set the Message instruction field flag in the TRACE CONNECTION ACKNOWLEDGE message to "follow explicit instructions" and the Message action indicator to either "discard and ignore" or "discard and report status"?	M		5.4	Yes__No__
CCT 3	Does the IUT when acting as the trace source node set the IE instruction field flag in the TTL information element to "follow explicit instructions" and the IE action indicator to either "discard message, and ignore" or "discard message, and report status"?	M		5.4	Yes__No__
Comments					

Annex C Path and Connection Trace SNMP MIB

[NORMATIVE]

```
ATM-TRACE-MIB DEFINITIONS ::= BEGIN

IMPORTS
        MODULE-IDENTITY, OBJECT-TYPE, Integer32, Counter32,
        NOTIFICATION-TYPE, enterprises
                FROM SNMPv2-SMI
        TEXTUAL-CONVENTION, RowStatus, TimeStamp, TruthValue
                FROM SNMPv2-TC
        MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
                FROM SNMPv2-CONF
        PnniNodeId, PnniPortId, pnniIfEntry
                FROM PNNI-MIB
        AtmAddr, AtmConnCastType, AtmConnKind, AtmServiceCategory,
        AtmVcIdentifier, AtmVpIdentifier, AtmTrafficDescrParamIndex
                FROM ATM-TC-MIB
        InterfaceIndex, InterfaceIndexOrZero
                FROM IF-MIB
        NetworkEntityNetworkServiceCategory,
        ResourcePartitionNetworkServiceCategory,
        PolicyConstraintIndex
                FROM ATM-POLICY-CONSTRAINT-MIB
        MplsLabel
                FROM MPLS-TC-STD-MIB;


atmTraceMIB MODULE-IDENTITY
    LAST-UPDATED    "~~0002220000Z~~200402061200Z"
    ORGANIZATION    "The ATM Forum."
    CONTACT-INFO
        "The ATM Forum
        2570 West El Camino Real, Suite 304
        Mountain View, CA 94040-1313 USA
        Phone: +1 650-949-6700
        Fax:   +1 415-949-6705
        info@atmforum.com"
    DESCRIPTION
        "The MIB module for ATM path and connection trace."

    REVISION        "200402061200Z"
    DESCRIPTION
        "Addition of objects to support tracing of Policy Routing
         and ATM-MPLS Network Interworking related information."

    REVISION        "0002220000Z"
    DESCRIPTION
        "Initial version of the MIB for ATM path and
        connection trace."
    ::= { atmfTrace 1 }
```

```
atmForum        OBJECT IDENTIFIER ::= { enterprises 353 }

atmForumNetworkManagement  OBJECT IDENTIFIER ::= { atmForum 5 }

atmfSignalling   OBJECT IDENTIFIER ::= { atmForumNetworkManagement 9 }

atmfTrace       OBJECT IDENTIFIER ::= { atmfSignalling 2 }


atmTraceMIBObjects  OBJECT IDENTIFIER ::= { atmTraceMIB 1 }


-- Textual Conventions

CallReference ::= TEXTUAL-CONVENTION
    STATUS     current
    DESCRIPTION
        "The 24-bit Call Reference used by signalling
        to identify a connection.  The Call Reference is structured in
        two parts.  The most significant bit represents the Call
        Reference Flag and the 23 least significant bits represent the
        Call Reference Value.  For the same call, the Call Reference
        Value is identical on both sides of an interface while the Call
        Reference Flag is different.  For the side originating the Call
        Reference, the Call Reference Flag is set to '0' while it is
        set to '1' for the side not originating the Call Reference.

        The distinguished value zero indicates that no Call Reference
        value was returned in the trace transit list."
    REFERENCE
        "ITU-T Recommendation Q.2931 Section 4.3"
    SYNTAX     Integer32 (0..16777215)

AtmEndPointReference ::= TEXTUAL-CONVENTION
    STATUS     current
    DESCRIPTION
        "For point-to-multipoint SVCs and the switched connection legs
        of point-to-multipoint Soft PVCs, the 16-bit Endpoint Reference
        used by signalling to identify a leaf of the
        point-to-multipoint connection.  The Endpoint Reference is
        structured in two parts.  The most significant bit represents
        the Endpoint Reference Flag and the 15 least significant bits
        represent the Endpoint Reference Value.  For the same call and
        leaf, the Endpoint Reference Value is identical on both sides
        of an interface while the Endpoint Reference Flag is different.
        For the side originating the endpoint reference, the Endpoint
        Reference Flag is set to '0' while it is set to '1' for the
        side not originating the Endpoint Reference.

        For the permanent connection legs at the root of
        point-to-multipoint Soft PVCs, the value used to identify a
        leaf in the atmSoftPVccLeafReference object or the
        atmSoftPVpcLeafReference object from the ATM-SOFT-PVC-MIB.

        For the permanent connection legs at the leaf end of
        point-to-multipoint Soft PVCs, the value 1 shall be used
        (consistent with the values used for atmSoftPVccLeafReference
        and atmSoftPVpcLeafReference in the ATM-SOFT-PVC-MIB).

        The distinguished value -1 indicates that no Endpoint Reference
        value was returned in the Trace transit list."
    REFERENCE
        "ITU-T Recommendation Q.2971 Section 8.2.1,
        ATM Forum PNNI v1.0 Addendum (Soft PVC MIB), af-pnni-0066.000"
    SYNTAX     Integer32 (-1..65535)

AtmTraceRecordIndex ::= TEXTUAL-CONVENTION
    STATUS     current
    DESCRIPTION
        "The value of this object identifies a row in the
        atmTraceRecordTable.  The distinguished value zero signifies
        that no row has been identified."
    SYNTAX     Integer32 (0..2147483647)

AtmTraceOwnerString ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "255a"
    STATUS     current
    DESCRIPTION
        "This data type is used to model an administratively
        assigned name of the owner of a resource.  This
        information is taken from the NVT ASCII character set.
        It is suggested that this name contain one or more of
        the following: ASCII form of the manager station's
        transport address, management station name (e.g.,
        domain name), network management personnel's name,
        location, or phone number.  In some cases the agent
        itself will be the owner of an entry.  In these cases,
        this string shall be set to a string starting with
        'monitor'.

        SNMP access control is articulated entirely in terms
        of the contents of MIB views; access to a particular
        SNMP object instance depends only upon its presence
        or absence in a particular MIB view and never upon
        its value or the value of related object instances.
        Thus, objects of this type afford resolution of
        resource contention only among cooperating managers;
        they realize no access control function with respect
        to uncooperative parties."
    SYNTAX     OCTET STRING (SIZE(0..127))


-- This MIB contains six tables and a number of scalars.  The tables
-- are:
--   Trace Connection Table - trigger connection trace
--   Trace Path Test Table - trigger a test connection or party
--   Trace Path Filter Table - trigger path trace by filtering calls
--   Trace Filter Record Table - correlate records with filters
--   Trace Record Table - overall info about a connection or party
--   Trace Info Table - detailed trace info for a connection or party
--   Trace Interface Table - specify certain PNNI interfaces as
--                           trace destination interfaces
```

```
atmTraceBaseGroup OBJECT IDENTIFIER ::= {atmTraceMIBObjects 1}

atmTraceFilterControl OBJECT-TYPE
    SYNTAX     INTEGER {
                         enable(1),
                         disable(2)
                         }
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
        "This object enables or disables the path trace filtering
        feature in the ATM device.  When this object is modified
        from 'enable' to 'disable' the records in the
        atmTraceRecordTable are not removed but filtering is
        stopped in the device."
    DEFVAL     { disable }
    ::= { atmTraceBaseGroup 1 }

atmTraceMaxConcurrentRequests OBJECT-TYPE
    SYNTAX     Integer32 (0..65535)
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
        "The maximum number of concurrent active path or connection
        trace requests (i.e., connections or parties for which trace
        information gathering has been initiated, but for which no
        reply has been received yet) that are allowed by the agent.  A
        value of 0 for this object implies that there is no limit on
        the number of concurrent active requests."
    ::= { atmTraceBaseGroup 2 }

atmTraceAvailableRequests OBJECT-TYPE
    SYNTAX     Integer32 (0..65535)
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The number of new path or connection trace requests that can
        be initiated on the agent at this moment in time.  This is
        equal to the maximum number of concurrent active path or
        connection trace requests that are allowed by the agent (i.e.,
        atmTraceMaxConcurrentRequests), minus the current number of
        active path or connection trace requests."
    ::= { atmTraceBaseGroup 3 }

atmTraceTransitListMaximumSize OBJECT-TYPE
    SYNTAX     Integer32 (1466..65535)
    UNITS      "octets"
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
        "The maximum size in octets of the Trace transit list
        information element generated in any signalling message."
    DEFVAL     { 1466 }
    ::= { atmTraceBaseGroup 4 }


atmTraceConnGroup OBJECT IDENTIFIER ::= { atmTraceMIBObjects 2 }
```

```
atmTraceConnTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF AtmTraceConnEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "The table whose entries describe existing connections and
        parties to be traced or in the process of being traced."
    ::= { atmTraceConnGroup 1 }

atmTraceConnEntry OBJECT-TYPE
    SYNTAX     AtmTraceConnEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "Each entry in this table specifies an existing connection or
        party to be traced, in the process of being traced, or that
        has recently been traced.  The results of the connection trace
        are returned in the atmTraceRecordTable and the
        atmTraceInfoTable.

        A management station wishing to create an entry should first
        create the associated instance of the row status and row
        owner objects, using a value of atmTraceConnIndex that is not
        currently in use.  It must also, either in the same or in
        successive PDUs, create the associated instance of the objects
        that identify the connection to be traced.  It should also
        modify the default values for the other configuration objects
        if the defaults are not appropriate.

        Once the appropriate instances of all the configuration
        objects have been created, either by an explicit SNMP
        set request or by default, the row status should be set
        to active to initiate the request.  Note that this entire
        procedure may be initiated via a single set request which
        specifies a row status of createAndGo as well as specifies
        valid values for the non-defaulted configuration objects.

        After the connection trace completes, the management
        station should retrieve the values of the status objects of
        interest from the atmTraceRecordTable, and should then delete
        the entry.  In order to prevent old entries from clogging the
        table, entries will be aged out, but an entry will not be
        deleted within 5 minutes of the last activity."
    INDEX      { atmTraceConnIndex }
    ::= { atmTraceConnTable 1 }

AtmTraceConnEntry ::= SEQUENCE
    {
            atmTraceConnIndex             Integer32,
            atmTraceConnOwner             AtmTraceOwnerString,
            atmTraceConnTraceSourceIf     InterfaceIndex,
            atmTraceConnOrigConnType      INTEGER,
            atmTraceConnOrigVpi           AtmVpIdentifier,
            atmTraceConnOrigVci           AtmVcIdentifier,
            atmTraceConnEndPtRef          AtmEndPointReference,
            atmTraceConnCallRef           CallReference,
            atmTraceConnOrigDlci          Integer32,
            atmTraceConnOrigDirection     INTEGER,
            atmTraceConnTraceConnId       TruthValue,
            atmTraceConnTraceCallRef      TruthValue,
            atmTraceConnPassAlongRequest  TruthValue,
            atmTraceConnFailTimeout       Integer32,
            atmTraceConnAgeTimeout        Integer32,
            atmTraceConnRestart           INTEGER,
            atmTraceConnTrapOnCompletion  TruthValue,
            atmTraceConnRecordIndex       AtmTraceRecordIndex,
            atmTraceConnRowStatus         RowStatus,
            atmTraceConnTraceNeNsc        TruthValue,
            atmTraceConnTraceRpNsc        TruthValue,
            atmTraceConnTraceIncoming     TruthValue,
            atmTraceConnTraceLabels       TruthValue
    }

atmTraceConnIndex OBJECT-TYPE
    SYNTAX     Integer32
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An arbitrary integer uniquely identifying a connection
        trace request."
    ::= { atmTraceConnEntry 1 }

atmTraceConnOwner OBJECT-TYPE
    SYNTAX     AtmTraceOwnerString
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The entity that configured this entry."
    ::= { atmTraceConnEntry 2 }

atmTraceConnTraceSourceIf OBJECT-TYPE
    SYNTAX     InterfaceIndex
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The interface at which the connection trace is originated.
        This object must be specified."
    ::= { atmTraceConnEntry 3 }

atmTraceConnOrigConnType OBJECT-TYPE
    SYNTAX     INTEGER {
                        other(1),
                        atmVcc(2),
                        atmVpc(3),
                        atmCOBISigConn(4),
                        frameRelayVc(5)
                        }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The type of connection at the trace source interface."
    DEFVAL     { atmVcc }
    ::= { atmTraceConnEntry 4 }
```

```
atmTraceConnOrigVpi OBJECT-TYPE
    SYNTAX     AtmVpIdentifier
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates the VPI value of the existing connection on the
        trace source interface.

        This object only applies when atmTraceConnOrigConnType is set
        to 'atmVcc' or 'atmVpc'.  When this is the case, either this
        object or atmTraceConnCallRef must be specified.  When the
        atmTraceConnCallRef object is specified, this object may not
        be set."
    ::= { atmTraceConnEntry 5 }

atmTraceConnOrigVci  OBJECT-TYPE
    SYNTAX     AtmVcIdentifier
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates the VCI value of the existing connection on the
        trace source interface.

        This object only applies when atmTraceConnOrigConnType is set
        to 'atmVcc'.  When this is the case, either this
        object or atmTraceConnCallRef must be specified.  When the
        atmTraceConnCallRef object is specified, this object may not
        be set."
    ::= { atmTraceConnEntry 6 }

atmTraceConnEndPtRef  OBJECT-TYPE
    SYNTAX     AtmEndPointReference
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The Endpoint Reference value identifying a leaf of an
        existing point-to-multipoint connection.

        This object does not apply when the connection is not a point-
        to-multipoint connection."
    ::= { atmTraceConnEntry 7 }

atmTraceConnCallRef      OBJECT-TYPE
    SYNTAX     CallReference
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates the call reference value of the existing connection
        on the trace source interface.

        For CO-BI connections, the value of this object must be
        specified.  For other types of connections, either this object
        or the connection identifier objects (atmTraceConnOrigVpi,
        atmTraceConnOrigVpi and atmTraceConnOrigVci, or
        atmTraceConnOrigDlci, as appropriate for the connection type)
        must be specified.  When any of atmTraceConnOrigVpi,
        atmTraceConnOrigVci, or atmTraceConnOrigDlci are specified,
        this object may not be set."
    ::= { atmTraceConnEntry 8 }

atmTraceConnOrigDlci         OBJECT-TYPE
    SYNTAX     Integer32
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates the DLCI value of the existing connection on the
        trace source interface.

        This object only applies when atmTraceConnOrigConnType is set
        to 'frameRelayVc'.  When this is the case, either this
        object or atmTraceConnCallRef must be specified.  When the
        atmTraceConnCallRef object is specified, this object may not
        be set."
    ::= { atmTraceConnEntry 9 }

atmTraceConnOrigDirection    OBJECT-TYPE
    SYNTAX     INTEGER {
                         incoming(1),
                         outgoing(2)
                         }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the connection trace is to proceed in the
        incoming direction from the trace source interface, or in the
        outgoing direction from the trace source interface."
    DEFVAL     { incoming }
    ::= { atmTraceConnEntry 10 }

atmTraceConnTraceConnId      OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the connection trace shall include
        connection identifier (e.g. VPI/VCI, DLCI) information."
    DEFVAL     { false }
    ::= { atmTraceConnEntry 11 }

atmTraceConnTraceCallRef     OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the connection trace shall include
        call reference information, and endpoint reference information
        for point-to-multipoint connections."
    DEFVAL     { false }
    ::= { atmTraceConnEntry 12 }

atmTraceConnPassAlongRequest OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the 'pass along request' bit shall be set
        in the Trace transit list information element.  When this
        object is set to 'true' and systems that do not support
        connection trace are present in the network, gaps may occur
        between successive entries in the atmTraceInfoTable identifying
        logical nodes and logical ports traversed by this
        connection or party."
    DEFVAL     { true }
    ::= { atmTraceConnEntry 13 }

atmTraceConnFailTimeout      OBJECT-TYPE
    SYNTAX     Integer32 (0..100)
    UNITS      "seconds"
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The number of seconds left before the connection trace is
        declared to have failed.  After this timer expires the value
        of this object will be zero and the atmTraceConnRecordIndex
        will also remain at zero.  If the timer expires and
        atmTraceConnTrapOnCompletion is set to 'true', an
        atmTraceConnCompletion trap will be generated."
    DEFVAL     { 30 }
    ::= { atmTraceConnEntry 14 }

atmTraceConnAgeTimeout       OBJECT-TYPE
    SYNTAX     Integer32 (-1..2147483647)
    UNITS      "seconds"
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The number of seconds left for this entry to age out.
        On expiry of this timer the display records in the
        atmTraceRecordTable and atmTraceInfoTable corresponding to this
        entry are deleted, as well as the atmTraceConnEntry.

        When the management station modifies this object,
        the currently running timer, if any, is aborted and a timer is
        started with the new value of this object. The value '-1' will
        indicate an infinite timeout value. "
    DEFVAL     { 600 }
    ::= { atmTraceConnEntry 15 }

atmTraceConnRestart      OBJECT-TYPE
    SYNTAX     INTEGER {
                         restart(1),
                         noop(2)
                         }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "When the value is set to 'restart', the record for this
        connection trace is cleared and the connection trace is
        initiated.
        When the value is set to 'noop' no operation is performed.
        When read, the value 'noop' is returned."
    DEFVAL     { noop }
    ::= { atmTraceConnEntry 16 }

atmTraceConnTrapOnCompletion OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Specifies whether an atmTraceConnCompletion trap shall be
        issued on completion of the connection trace.  If such a trap
        is desired, it is the responsibility of the management entity
        to ensure that the SNMP administrative model is configured in
        such a way as to allow the trap to be delivered."
    DEFVAL     { false }
    ::= { atmTraceConnEntry 17 }

atmTraceConnRecordIndex OBJECT-TYPE
    SYNTAX     AtmTraceRecordIndex
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The value of this object identifies the row in the
        atmTraceRecordTable that was generated by this connection
        trace.  The distinguished value zero indicates that no
        reply has been received yet or that no reply was received
        before expiry of atmTraceConnFailTimeout, so no record has been
        generated."
    ::= { atmTraceConnEntry 18 }

atmTraceConnRowStatus OBJECT-TYPE
    SYNTAX     RowStatus
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Used to create and delete entries in this table.  When a row
        is activated, a connection trace is initiated."
    ::= { atmTraceConnEntry 19 }

atmTraceConnTraceNeNsc      OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the connection trace shall include
        Ne-NSCs supporting the connection ."
    DEFVAL     { false }
    ::= { atmTraceConnEntry 20 }

atmTraceConnTraceRpNsc      OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the connection trace shall include
        Rp-NSCs supporting the connection."
    DEFVAL      { false }
    ::= { atmTraceConnEntry 21 }

atmTraceConnTraceIncoming     OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS read-create
    STATUS      current
    DESCRIPTION
        "Indicates whether the connection trace shall record
        the NSCs supporting the connection at the incoming
        interface of the nodes.  If this value is set to true and
        the atmTraceConnTraceNeNsc object is also true, then the
        trace shall include the list of Ne-NSCs supporting the
        connection at the incoming interface of the nodes.
        If this value is set to true and the atmTraceConnTraceRpNsc
        object is also true, then the trace shall include the
        list of Rp-NSCs supporting the connection at the incoming
        interface of the nodes.
        If this value is set to false, then the trace shall not
        record the NSCs supporting the connection at the incoming
        interface of the nodes."
    DEFVAL      { false }
    ::= { atmTraceConnEntry 22 }

atmTraceConnTraceLabels     OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS read-create
    STATUS      current
    DESCRIPTION
        "Indicates whether the connection trace shall include
        interworking LSP labels, if applicable."
    DEFVAL      { false }
    ::= { atmTraceConnEntry 23 }

atmTracePathTestGroup OBJECT IDENTIFIER ::= { atmTraceMIBObjects 3 }

atmTracePathTestTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF AtmTracePathTestEntry
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
        "The table whose entries describe test connections and parties
        initiated to determine paths across the network.  Typically
        these connections and parties are cleared when the trace
        destination node is reached, but the entry may be configured
        to leave the connections up using the
        atmTracePathTestClearCallAtTDest object."
    ::= { atmTracePathTestGroup 1 }

atmTracePathTestEntry OBJECT-TYPE
    SYNTAX      AtmTracePathTestEntry
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
        "Each entry in this table specifies a test connection or test
        party that is initiated in order to determine a path across
        the network.  Typically the connection or party is cleared when
        the trace destination node is reached, but the entry may be
        configured to leave the connections up using the
        atmTracePathTestClearCallAtTDest object.  The results of the
        path trace are returned in the atmTraceFilterRecordTable, the
        atmTraceRecordTable, and the atmTraceInfoTable.

        A management station wishing to create an entry should first
        create the associated instance of the row status and row
        owner objects, using a value of atmTracePathTestIndex that is
        not currently in use.  It must also, either in the same or
        in successive PDUs, create the associated instance of the
        address objects.  It should also modify the default values
        for the other configuration objects if the defaults are not
        appropriate.

        Once the appropriate instance of all the configuration
        objects have been created, either by an explicit SNMP
        set request or by default, the row status should be set
        to active to initiate the request.  Note that this entire
        procedure may be initiated via a single set request which
        specifies a row status of createAndGo as well as specifies
        valid values for the non-defaulted configuration objects.

        After the test connection or party completes, the management
        station should retrieve the values of the status objects of
        interest from the atmTraceRecordTable, and should then delete
        the entry.  In order to prevent old entries from clogging the
        table, entries will be aged out, but an entry will not be
        deleted within 5 minutes of completing."
    INDEX      { atmTracePathTestIndex }
    ::= { atmTracePathTestTable 1 }

AtmTracePathTestEntry ::= SEQUENCE
    {
            atmTracePathTestIndex             Integer32,
            atmTracePathTestOwner             AtmTraceOwnerString,
            atmTracePathTestConnType          INTEGER,
            atmTracePathTestConnCastType      AtmConnCastType,
            atmTracePathTestTraceSourceIf     InterfaceIndex,
            atmTracePathTestP2MpNewConn       TruthValue,
            atmTracePathTestOrigVpi           AtmVpIdentifier,
            atmTracePathTestOrigVci           AtmVcIdentifier,
            atmTracePathTestCalledParty       AtmAddr,
            atmTracePathTestCallingParty      AtmAddr,
            atmTracePathTestRxTrafDescrIndex AtmTrafficDescrParamIndex,
            atmTracePathTestTxTrafDescrIndex AtmTrafficDescrParamIndex,
            atmTracePathTestClearCallAtTDest TruthValue,
            atmTracePathTestTraceCrankback   TruthValue,
            atmTracePathTestTraceConnId       TruthValue,
            atmTracePathTestTraceCallRef      TruthValue,
            atmTracePathTestPassAlongRequest TruthValue,
            atmTracePathTestAgeTimeout        Integer32,
            atmTracePathTestRestart           INTEGER,
            atmTracePathTestTrapOnCompletion TruthValue,
            atmTracePathTestRecordIndex       AtmTraceRecordIndex,
            atmTracePathTestRowStatus         RowStatus,
            atmTracePathTestTraceNeNsc      TruthValue,
            atmTracePathTestTraceRpNsc      TruthValue,
            atmTracePathTestTraceIncoming   TruthValue,
            atmTracePathTestPolicyConstraint PolicyConstraintIndex,
            atmTracePathTestTraceLabels     TruthValue
    }

atmTracePathTestIndex OBJECT-TYPE
    SYNTAX     Integer32
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An arbitrary integer uniquely identifying a test connection
        or test party."
    ::= { atmTracePathTestEntry 1 }

atmTracePathTestOwner OBJECT-TYPE
    SYNTAX     AtmTraceOwnerString
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The entity that configured this entry."
    ::= { atmTracePathTestEntry 2 }

atmTracePathTestConnType  OBJECT-TYPE
    SYNTAX     INTEGER {
                        other(1),
                        atmVcc(2),
                        atmVpc(3),
                        atmCOBISigConn(4)
                        }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the test connection is a virtual channel
        connection, a virtual path connection, or a connection-
        oriented bearer-independent signalling connection."
    DEFVAL     { atmVcc }
    ::= { atmTracePathTestEntry 3 }

atmTracePathTestConnCastType  OBJECT-TYPE
    SYNTAX     AtmConnCastType
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The connection topology type (e.g. point-to-point or
        point-to-multipoint) of the test connection or party."
    DEFVAL     { p2p }
    ::= { atmTracePathTestEntry 4 }

atmTracePathTestTraceSourceIf OBJECT-TYPE
    SYNTAX     InterfaceIndex
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The interface at which the test connection or test party is
        originated."
    ::= { atmTracePathTestEntry 5 }

atmTracePathTestP2MpNewConn  OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the test party for a point-to-multipoint
        connection is initiated on an existing branch (indicated by
        setting this object to 'false') or whether it is the first
        party of a new connection (indicated by setting this object
        to 'true').

        This object does not apply when atmTracePathTestConnCastType
        is set to 'p2p'."
    DEFVAL     { false }
    ::= { atmTracePathTestEntry 6 }

atmTracePathTestOrigVpi  OBJECT-TYPE
    SYNTAX     AtmVpIdentifier
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "When atmTracePathTestConnCastType is set to 'p2mpRoot' and
        atmTracePathTestP2MpNewConn is set to 'false' (i.e. path trace
        of a new party on an existing connection is requested),
        this object indicates the VPI value of the existing connection
        on the trace source interface.

        When atmTracePathTestClearCallAtTDest is set to 'true' and
        either
        - atmTracePathTestConnCastType is 'p2p', or
        - atmTracePathTestConnCastType is 'p2mpRoot' and
          atmTracePathTestP2MpNewConn is set to 'true',
        this object is not applicable.

        When atmTracePathTestClearCallAtTDest is set to 'false', this
        object indicates the VPI value to be used for the active
        connection.  Note that if the VPI/VCI can be assigned at the
        other side of the interface, there may be some possibility of
        VPI/VCI collision for SETUP messages received on this
        interface, when the other side of the interface does not know
        that this VPI/VCI value is being used."
    DEFVAL     { 0 }
    ::= { atmTracePathTestEntry 7 }

atmTracePathTestOrigVci  OBJECT-TYPE
    SYNTAX     AtmVcIdentifier
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "When atmTracePathTestConnCastType is set to 'p2mpRoot' and
        atmTracePathTestP2MpNewConn is set to 'false' (i.e. path trace
        of a new party on an existing connection is requested),
        this object indicates the VCI value of the existing connection
        on the trace source interface.
        When atmTracePathTestClearCallAtTDest is set to 'true' and
        either
        - atmTracePathTestConnCastType is 'p2p', or
        - atmTracePathTestConnCastType is 'p2mpRoot' and
          atmTracePathTestP2MpNewConn is set to 'true',
        this object is not applicable.

        When atmTracePathTestClearCallAtTDest is set to 'false', this
        object indicates the VCI value to be used for the active
        connection.  Note that if the VPI/VCI can be assigned at the
        other side of the interface, there may be some possibility of
        VPI/VCI collision for SETUP messages received on this
        interface, when the other side of the interface does not know
        that this VPI/VCI value is being used.

        If atmTracePathTestConnType is set to a value other than
        'atmVcc', this value is set to zero."
    DEFVAL     { 0 }
    ::= { atmTracePathTestEntry 8 }

atmTracePathTestCalledParty     OBJECT-TYPE
    SYNTAX     AtmAddr
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The called party number towards which the test connection or
        test party is to be initiated."
    ::= { atmTracePathTestEntry 9 }

atmTracePathTestCallingParty    OBJECT-TYPE
    SYNTAX     AtmAddr
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The calling party number used for the test connection or test
        party."
    DEFVAL     { "" }
    ::= { atmTracePathTestEntry 10 }

atmTracePathTestRxTrafDescrIndex  OBJECT-TYPE
    SYNTAX     AtmTrafficDescrParamIndex
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The value of this object identifies the row of the ATM
        Traffic Descriptor Table which applies to the receive
        direction of this test connection (from the point of view
        of the trace source interface).

        This object does not apply when the value of
        atmTracePathTestP2MpNewConn is 'false'."
    DEFVAL     { 0 }
    ::= { atmTracePathTestEntry 11 }

atmTracePathTestTxTrafDescrIndex  OBJECT-TYPE
    SYNTAX     AtmTrafficDescrParamIndex
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The value of this object identifies the row of the ATM
        Traffic Descriptor Table which applies to the transmit
        direction of this test connection (from the point of view
        of the trace source interface).

        This object does not apply when the value of
        atmTracePathTestP2MpNewConn is 'false'."
    DEFVAL     { 0 }
    ::= { atmTracePathTestEntry 12 }

atmTracePathTestClearCallAtTDest OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the test connection or party shall be
        cleared when the trace destination node is reached.  When this
        is set to 'false', the test connection/party shall be cleared
        when the entry is deleted using the atmTracePathTestRowStatus
        object."
    DEFVAL     { true }
    ::= { atmTracePathTestEntry 13 }

atmTracePathTestTraceCrankback   OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include crankback
        information.  When this is set to false, as a consequence of
        the signalling procedures for path trace, trace information
        will only be returned if the connection or party succeeds."
    DEFVAL     { false }
    ::= { atmTracePathTestEntry 14 }

atmTracePathTestTraceConnId      OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        connection identifier (e.g. VPI/VCI, DLCI) information."
    DEFVAL     { false }
    ::= { atmTracePathTestEntry 15 }

atmTracePathTestTraceCallRef     OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        call reference information, and endpoint reference information
        for point-to-multipoint connections."
    DEFVAL     { false }
    ::= { atmTracePathTestEntry 16 }
```

```
atmTracePathTestPassAlongRequest OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the 'pass along request' bit shall be set
        in the Trace transit list information element.  When this
        object is set to 'true' and systems that do not support path
        trace are present in the network, gaps may occur between
        successive entries in the atmTraceInfoTable identifying logical
        nodes and logical ports traversed by this connection or party.
        When this object is set to 'false', trace information might not
        be returned unless all systems along the path support the path
        trace functionality."
    DEFVAL     { true }
    ::= { atmTracePathTestEntry 17 }

atmTracePathTestAgeTimeout       OBJECT-TYPE
    SYNTAX     Integer32 (-1..2147483647)
    UNITS      "seconds"
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The number of seconds left for this entry to age out.
        On expiry of this timer the display records in the
        atmTraceRecordTable and the atmTraceInfoTable corresponding to
        this entry are deleted, as well as the atmTracePathTestEntry.

        When the management station modifies this object,
        the currently running timer, if any, is aborted and a timer is
        started with the new value of this object. The value '-1' will
        indicate an infinite timeout value. "
    DEFVAL     { 600 }
    ::= { atmTracePathTestEntry 18 }

atmTracePathTestRestart      OBJECT-TYPE
    SYNTAX     INTEGER {
                       restart(1),
                       noop(2)
                       }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "When the value is set to 'restart', the test record for this
        connection is cleared and the test connection or party is
        initiated.

        When the value is set to 'noop' no operation is performed.
        When read, the value 'noop' is returned."
    DEFVAL    { noop }
    ::= { atmTracePathTestEntry 19 }

atmTracePathTestTrapOnCompletion OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Specifies whether an atmTracePathTestCompletion trap shall be
        issued on completion of the path trace.  If such a trap
        is desired, it is the responsibility of the management entity
        to ensure that the SNMP administrative model is configured in
        such a way as to allow the trap to be delivered."
    DEFVAL     { false }
    ::= { atmTracePathTestEntry 20 }

atmTracePathTestRecordIndex OBJECT-TYPE
    SYNTAX     AtmTraceRecordIndex
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The value of this object identifies the row in the
        atmTraceRecordTable that was generated by this test
        connection or party.  The distinguished value zero indicates
        that no reply has been received yet, so no record has been
        generated."
    ::= { atmTracePathTestEntry 21 }

atmTracePathTestRowStatus  OBJECT-TYPE
    SYNTAX     RowStatus
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Used to create and delete entries in this table.  When a row
        is activated, a test connection or test party is initiated.
        When the row is deleted, the test connection or test party is
        cleared (if it has not already been cleared) and the
        corresponding entry in the atmTraceRecordTable is deleted."
    ::= { atmTracePathTestEntry 22 }

atmTracePathTestTraceNeNsc     OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        Ne-NSCs supporting the connection."
    DEFVAL     { false }
    ::= { atmTracePathTestEntry 23 }

atmTracePathTestTraceRpNsc     OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        Rp-NSCs supporting the connection."
    DEFVAL     { false }
    ::= { atmTracePathTestEntry 24 }

atmTracePathTestTraceIncoming  OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        the NSCs supporting the connection at the incoming
        interface of the nodes.  If this value is set to true and
        the atmTracePathtestTraceNeNsc object is also true, then the
        trace shall include the list of Ne-NSCs supporting the
        connection at the incoming interface of the nodes.
        If this value is set to true and the atmTracePathTestTraceRpNsc
        object is also true, then the trace shall include the
        list of Rp-NSCs supporting the connection at the incoming
        interface of the nodes.
        If this value is set to false, then the trace shall not
        record the NSCs supporting the connection at the incoming
        interface of the nodes."
    DEFVAL      { false }
    ::= { atmTracePathTestEntry 25 }

atmTracePathTestPolicyConstraint  OBJECT-TYPE
    SYNTAX      PolicyConstraintIndex
    MAX-ACCESS read-create
    STATUS      current
    DESCRIPTION
        "Defines the row of the policyConstraintTable
        that specifies the policy constraint to be used
        for the establishment of the test connection."
    DEFVAL      { 0 }
    ::= { atmTracePathTestEntry 26 }

atmTracePathTestTraceLabels     OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS read-create
    STATUS      current
    DESCRIPTION
        "Indicates whether the path trace shall include
        interworking LSP labels, if applicable."
    DEFVAL      { false }
    ::= { atmTracePathTestEntry 27 }

atmTraceFilterGroup OBJECT IDENTIFIER ::= { atmTraceMIBObjects 4 }

atmTraceFilterTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF AtmTraceFilterEntry
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
        "The table whose entries describe the filtering criteria
        for recording of path trace information."
    ::= { atmTraceFilterGroup 1 }

atmTraceFilterEntry OBJECT-TYPE
    SYNTAX      AtmTraceFilterEntry
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
        "Each entry in this table corresponds to a filtering criteria
        based on which path trace is initiated for connections and
        parties in the process of being established.  This selection
        criteria is applied against all connections and parties
        generated or detectedat this ATM device.  Only connections and
        parties being established which match against all of the
        entry's criteria are recorded in the atmTraceRecordTable.

        A new entry can be created by specifying a atmTraceFilterIndex
        value that is currently not being used and also using an
        appropriate value (createAndGo or createAndWait) for the
        atmTraceFilterRowStatus object.

        If a particular connection or party matches multiple entries in
        the atmTraceFilterTable then multiple entries will be created
        in the atmTraceRecordTable for each of the matched entries in
        the atmTraceFilterTable."
    INDEX      { atmTraceFilterIndex }
    ::= { atmTraceFilterTable 1 }

AtmTraceFilterEntry ::= SEQUENCE
    {
            atmTraceFilterIndex                Integer32,
            atmTraceFilterOwner                AtmTraceOwnerString,
            atmTraceFilterConnKind             BITS,
            atmTraceFilterConnCastType         BITS,
            atmTraceFilterServiceCategory      BITS,
            atmTraceFilterInIf                 InterfaceIndexOrZero,
            atmTraceFilterOutIf                InterfaceIndexOrZero,
            atmTraceFilterCallingPartyPrefix   AtmAddr,
            atmTraceFilterCallingPartyLength   Integer32,
            atmTraceFilterCalledPartyPrefix    AtmAddr,
            atmTraceFilterCalledPartyLength    Integer32,
            atmTraceFilterClearCallAtTDest     TruthValue,
            atmTraceFilterTraceCrankback       TruthValue,
            atmTraceFilterTraceConnId          TruthValue,
            atmTraceFilterTraceCallRef         TruthValue,
            atmTraceFilterPassAlongRequest     TruthValue,
            atmTraceFilterMaxRecords           Integer32,
            atmTraceFilterRecordCountDown      Integer32,
            atmTraceFilterStopTimeout          Integer32,
            atmTraceFilterAgeTimeout           Integer32,
            atmTraceFilterPurge                INTEGER,
            atmTraceFilterTrapEnable           TruthValue,
            atmTraceFilterNumMatches           Counter32,
            atmTraceFilterRowStatus            RowStatus,
            atmTraceFilterPolicy               TruthValue,
            atmTraceFilterTraceNeNsc           TruthValue,
            atmTraceFilterTraceRpNsc           TruthValue,
            atmTraceFilterTraceIncoming        TruthValue,
            atmTraceFilterTraceLabels          TruthValue
    }

atmTraceFilterIndex OBJECT-TYPE
    SYNTAX     Integer32 (1..50)
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An arbitrary integer uniquely identifying a filtering
        criteria."
    ::= { atmTraceFilterEntry 1 }

atmTraceFilterOwner OBJECT-TYPE
    SYNTAX     AtmTraceOwnerString
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The entity that configured this entry."
    ::= { atmTraceFilterEntry 2 }

atmTraceFilterConnKind  OBJECT-TYPE
    SYNTAX     BITS {
                      other(0),
                      svcAndSpvcNotInitiator(1),
                      spvcInitiator(2),
                      svpAndSpvpNotInitiator(3),
                      spvpInitiator(4),
                      atmCOBISigConn(5)
                      }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "This object enables the user to track the paths of switched
        virtual channel/path connections, soft permanent virtual
        channel/path connections initiated by this node, and soft
        permanent virtual channel/path connections initiated by
        other nodes."
    ::= { atmTraceFilterEntry 3 }

atmTraceFilterConnCastType OBJECT-TYPE
     SYNTAX     BITS {
                      p2p(0),
                      p2mp(1)
                      }
     MAX-ACCESS read-create
     STATUS     current
     DESCRIPTION
         "This object restricts the scope of the filter based on the
          type of topology of connections (point-to-point or
          point-to-multipoint)."
     ::= { atmTraceFilterEntry 4 }

atmTraceFilterServiceCategory OBJECT-TYPE
    SYNTAX     BITS {
                      cbr(0),
                      rtVbr(1),
                      nrtVbr(2),
                      abr(3),
                      ubr(4),
                      gfr(5),
                      other(6)
                      }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "This object restricts the scope of the filter to calls
        belonging to service categories represented by this object."
    ::= { atmTraceFilterEntry 5 }

atmTraceFilterInIf  OBJECT-TYPE
    SYNTAX     InterfaceIndexOrZero
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "This object restricts the scope of the filter to calls which
        enter the ATM device through the port represented by this
        object, or are initiated at this port (e.g. Soft PVCs).
        It has the value 0, or the ifIndex value of an ATM Interface.
        The value zero indicates that the scope of the filter is not
        restricted by the incoming port."
    DEFVAL     { 0 }
    ::= { atmTraceFilterEntry 6 }

atmTraceFilterOutIf OBJECT-TYPE
    SYNTAX     InterfaceIndexOrZero
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "This object restricts the scope of the filter to calls which
        exit the ATM device through the port represented by this
        object.  It has the value 0, or the ifIndex value of an ATM
        interface.  The value zero indicates that the scope of the
        filter is not restricted by the outgoing port."
    DEFVAL     { 0 }
    ::= { atmTraceFilterEntry 7 }

atmTraceFilterCallingPartyPrefix     OBJECT-TYPE
    SYNTAX     AtmAddr
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The combination of this object and the corresponding instance
        of atmTraceFilterCallingPartyLength is one selection criteria
        for this record.  To match this selection criteria, a
        connection setup must have a Calling Party Address which has
        an initial part (of length atmTraceFilterCalledPartyLength
        bits) equal in value to atmTraceFilterCallingParty.  When the
        default value for the object is retained then the call will
        match this filtering criteria for any calling address in the
        call, or if the calling party number is not present in the
        call.  The value must be padded with zeros from
        atmTraceFilterCallingPartyLength to the full length of the
        address (8 octets for E.164 numbers and 20 octets for AESAs)."
    DEFVAL     { "" }
    ::= { atmTraceFilterEntry 8 }

atmTraceFilterCallingPartyLength OBJECT-TYPE
    SYNTAX     Integer32 (1..160)
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "This object specifies the number of bits in
        atmTraceFilterCallingParty that shall be used when matching
        against the calling party of a new call setup."
    DEFVAL     { 152 }
    ::= { atmTraceFilterEntry 9 }

atmTraceFilterCalledPartyPrefix      OBJECT-TYPE
    SYNTAX     AtmAddr
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The combination of this object and the corresponding instance
        of atmTraceFilterCalledPartyLength is one selection criteria
        for this record.  To match this selection criteria, a
        connection setup must have a called party address which has
        an initial part (of length atmTraceFilterCalledPartyLength
        bits) equal in value to
        atmTraceFilterCalledParty. When the default value for
        the object is retained then the call will match this
        filtering criteria for any called address in the call.
        The value must be padded with zeros from
        atmTraceFilterCalledPartyLength to the full length of the
        address (8 octets for E.164 numbers and 20 octets for AESAs)."
    DEFVAL     {""}
    ::= { atmTraceFilterEntry 10 }

atmTraceFilterCalledPartyLength  OBJECT-TYPE
    SYNTAX     Integer32 (1..160)
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "This object specifies the number of bits in
        atmTraceFilterCalledParty that shall be used when matching
        against the called party of a new call setup."
    DEFVAL     { 152 }
    ::= { atmTraceFilterEntry 11 }

atmTraceFilterClearCallAtTDest OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the connection or party shall be cleared
        when the trace destination node is reached."
    DEFVAL     { false }
    ::= { atmTraceFilterEntry 12 }

atmTraceFilterTraceCrankback   OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        crankback information.  When this is set to false, as a
        consequence of the signalling procedures for path trace, trace
        information will only be returned if the connection or party
        succeeds."
    DEFVAL     { false }
    ::= { atmTraceFilterEntry 13 }
```

```
atmTraceFilterTraceConnId      OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        connection identifier (e.g. VPI/VCI, DLCI) information."
    DEFVAL     { false }
    ::= { atmTraceFilterEntry 14 }

atmTraceFilterTraceCallRef     OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        call reference information, and endpoint reference information
        for point-to-multipoint connections."
    DEFVAL     { false }
    ::= { atmTraceFilterEntry 15 }

atmTraceFilterPassAlongRequest OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the 'pass along request' bit shall be set
        in the Trace transit list information element.  When this
        object is set to 'true' and systems that do not support path
        trace are present in the network, gaps may occur between
        successive entries in the atmTraceInfoTable identifying logical
        nodes and logical ports traversed by this connection or party.
        When this object is set to 'false', trace information might not
        be returned unless all systems along the path support the path
        trace functionality."
    DEFVAL     { true }
    ::= { atmTraceFilterEntry 16 }

atmTraceFilterMaxRecords       OBJECT-TYPE
    SYNTAX     Integer32 (-1..214783647)
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The maximum number of entries that are desired in the
        atmTraceRecordTable on behalf of this filter.  The agent will
        not create more than this number of entries in the table, but
        may choose to further limit the number of entries for this
        filter in the atmTraceRecordTable for any reason including
        the lack of resources.  The agent will however dedicate
        resources for a minimum number of entries in the
        atmTraceRecordTable, to take care of temporary memory
        allocation failures in the system.  In case of memory
        allocation failures the agent will utilize these dedicated
        resources for the creation of the new entries.  If memory
        resource failures continue and the dedicated resources are
        exhausted then the records in the atmTraceRecordTable are
        pruned such that the oldest entries are removed to make way
```

```
          for the new entries.

          A value '-1' will indicate no upper limit for the number of
          records stored. The manager can set this object to -1 if
          overwriting of records is not desired. The new value for
          this object will take effect immediately.

          If this object is set to a value less than the number of
          entries that is currently present in the atmTraceRecordTable
          corresponding to this entry, then the oldest entries in the
          atmTraceRecordTable will be deleted so that their number
          equals the new value of this object.

          If the value of this object is changed from -1 to any other
          positive value then the entries will be pruned such that only
          the first n records collected for this entry are retained in
          the atmTraceRecordTable, n being the new value of this object."
    DEFVAL     { 20 }
    ::= { atmTraceFilterEntry 17 }

atmTraceFilterRecordCountDown      OBJECT-TYPE
    SYNTAX     Integer32 (-1..2147483647)
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The number of entries left to be collected in the
        atmTraceRecordTable before filtering is disabled for this
        entry. The display records in the atmTraceRecordTable
        corresponding to this entry are retained.  The management
        station can restart filtering for this entry by setting this
        object to a positive value (subject to atmTraceFilterRowStatus
        being 'active' and atmTraceFilterStopTimeout having a positive
        value or being set to '-1'). When the object is set to zero,
        filtering is stopped for this entry. When the management
        station modifies this object, the current value is replaced and
        the agent counts down from the new value of this object.

        The value '-1' indicates that filtering will not be
        automatically disabled based on the number of entries collected
        in the atmTraceRecordTable."
    DEFVAL     { -1 }
    ::= { atmTraceFilterEntry 18 }

atmTraceFilterStopTimeout       OBJECT-TYPE
    SYNTAX     Integer32 (-1..2147483647)
    UNITS      "seconds"
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The number of seconds left for this entry to collect records.
        On expiry of this timer filtering is disabled for this entry.
        The display records in the atmTraceRecordTable corresponding
        to this entry are retained. When the timer expires the object
        will have a value zero. The management station can restart
        filtering for this entry by setting this object to a positive
        value (subject to atmTraceFilterRowStatus being 'active' and
        atmTraceFilterRecordCountDown having a positive value or being
        set to '-1').  When the object is set to zero, filtering is
        stopped for this entry. When the management station modifies
        this object, the currently running timer, if any, is aborted
        and a timer is started with the new value of this object. The
        value '-1' will indicate an infinite timeout value."
    DEFVAL     { 600 }
    ::= { atmTraceFilterEntry 19 }

atmTraceFilterAgeTimeout       OBJECT-TYPE
    SYNTAX     Integer32 (-1..2147483647)
    UNITS      "seconds"
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The number of seconds left for this entry to age out.
        On expiry of this timer the display records in the
        atmTraceFilterRecordTable, atmTraceRecordTable and the
        atmTraceInfoTable corresponding to this entry are deleted,
        as well as the atmTraceFilterEntry.

        When the management station modifies this object,
        the currently running timer, if any, is aborted and a timer is
        started with the new value of this object. The value '-1' will
        indicate an infinite timeout value. "
    DEFVAL     { 600 }
    ::= { atmTraceFilterEntry 20 }

atmTraceFilterPurge OBJECT-TYPE
    SYNTAX     INTEGER {
                         purge(1),
                         noop(2)
                         }
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "The object provides a facility for the user to purge the
        records in the atmTraceRecordTable corresponding to this entry.
        When the value  is set to 'purge', the records in the
        atmTraceRecordTable corresponding to this entry are purged.
        When the value is set to 'noop' no operation is performed. When
        read, the value 'noop' is returned."
    DEFVAL     { noop }
    ::= { atmTraceFilterEntry 21 }

atmTraceFilterTrapEnable OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Specifies whether an atmTraceFilterTrap shall be issued the
        next time a record is added to the atmTraceRecordTable and the
        atmTraceFilterRecordTable corresponding to this filter.  This
        object automatically resets itself to 'false' each time a trap
        is generated for this filter.  This object must be reset to
        'true' before another atmTracePathFilter trap can be generated
        for this filter entry.
        If such a trap is desired, it is the responsibility of the
        management entity to ensure that the SNMP administrative model
        is configured in such a way as to allow the trap to be
        delivered."
    DEFVAL     { false }
    ::= { atmTraceFilterEntry 22 }

atmTraceFilterNumMatches OBJECT-TYPE
    SYNTAX     Counter32
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "A monotonically increasing counter to keep track of the
        number of calls or parties that matched this entry for
        the entire lifetime of this entry. "
    ::= { atmTraceFilterEntry 23 }

atmTraceFilterRowStatus  OBJECT-TYPE
    SYNTAX     RowStatus
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates the status of this row. Used according to the row
        installation and removal conventions. This object can be used
        to temporarily inactivate an entry in the table. When this
        object is set to a value 'notInService' filtering is stopped
        for the corresponding row. Any records that have been stored in
        the atmTraceRecordTable and the atmTraceFilterRecordTable
        corresponding to this entry will not be released. Any writeable
        objects in the row can be modified when the row is active. All
        values will take effect immediately. When this object is set to
        'destroy', all corresponding records in the
        atmTraceFilterRecordTable and the atmTraceRecordTable are
        deleted."
    ::= { atmTraceFilterEntry 24 }

atmTraceFilterPolicy     OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "This object restricts the scope of the filter to connection
        setups that include a Policy constraint information element."
    DEFVAL     { false }
    ::= { atmTraceFilterEntry 25 }

atmTraceFilterTraceNeNsc     OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-create
    STATUS     current
    DESCRIPTION
        "Indicates whether the path trace shall include
        Ne-NSCs supporting the connection."
    DEFVAL     { false }
    ::= { atmTraceFilterEntry 26 }

atmTraceFilterTraceRpNsc     OBJECT-TYPE
```

```
    SYNTAX      TruthValue
    MAX-ACCESS read-create
    STATUS      current
    DESCRIPTION
        "Indicates whether the path trace shall include
        Rp-NSCs supporting the connection."
    DEFVAL      { false }
    ::= { atmTraceFilterEntry 27 }

atmTraceFilterTraceIncoming  OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS read-create
    STATUS      current
    DESCRIPTION
        "Indicates whether the path trace shall include
        the NSCs supporting the connection at the incoming
        interface of the nodes.  If this value is set to true and
        the atmTraceFilterTraceNeNsc object is also true, then the
        trace shall include the list of Ne-NSCs supporting the
        connection at the incoming interface of the nodes.
        If this value is set to true and the atmTraceFilterTraceRpNsc
        object is also true, then the trace shall include the
        list of Rp-NSCs supporting the connection at the incoming
        interface of the nodes.
        If this value is set to false, then the trace shall not
        record the NSCs supporting the connection at the incoming
        interface of the nodes."
    DEFVAL      { false }
    ::= { atmTraceFilterEntry 28 }

atmTraceFilterTraceLabels  OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS read-create
    STATUS      current
    DESCRIPTION
        "Indicates whether the path trace shall include
        interworking LSP labels, if applicable."
    DEFVAL      { false }
    ::= { atmTraceFilterEntry 29 }

atmTraceFilterRecordTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF AtmTraceFilterRecordEntry
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
        "The table that lists which trace records have been
        returned for which trace filters.  This table also lists
        the connection characteristics for each connection record,
        other than those values returned in the Trace transit list
        information element."
    ::= { atmTraceFilterGroup 2 }

atmTraceFilterRecordEntry OBJECT-TYPE
    SYNTAX      AtmTraceFilterRecordEntry
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
```

```
        "An entry containing the index of a record associated with
        a given trace filter.  This table also lists some of the
        connection characteristics."
    INDEX      { atmTraceFilterIndex,
                 atmTraceFilterRecordIndex }
    ::= { atmTraceFilterRecordTable 1 }

AtmTraceFilterRecordEntry ::= SEQUENCE
    {
            atmTraceFilterRecordIndex            AtmTraceRecordIndex,
            atmTraceFilterRecordConnKind         AtmConnKind,
            atmTraceFilterRecordConnCastType     AtmConnCastType,
            atmTraceFilterRecordServiceCategory  AtmServiceCategory,
            atmTraceFilterRecordInIf             InterfaceIndex,
            atmTraceFilterRecordOutIf            InterfaceIndexOrZero,
            atmTraceFilterRecordCallingParty     AtmAddr,
            atmTraceFilterRecordCalledParty      AtmAddr
            }

atmTraceFilterRecordIndex OBJECT-TYPE
    SYNTAX     AtmTraceRecordIndex
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "The value of this object identifies a row in the
        atmTraceRecordTable that was generated by the trace filter
        identified by atmTraceFilterIndex."
    ::= { atmTraceFilterRecordEntry 1 }

atmTraceFilterRecordConnKind  OBJECT-TYPE
    SYNTAX     AtmConnKind
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "This object represents the use of call control (e.g.
        switched virtual connection or soft permanent virtual
        connection) of the connection or party on the incoming
        interface."
    ::= { atmTraceFilterRecordEntry 2 }

atmTraceFilterRecordConnCastType  OBJECT-TYPE
    SYNTAX     AtmConnCastType
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "This object represents the type of topology of the
        connection (point-to-point or point-to-multipoint) on
        the incoming interface."
    ::= { atmTraceFilterRecordEntry 3 }

atmTraceFilterRecordServiceCategory OBJECT-TYPE
    SYNTAX     AtmServiceCategory
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "This object represents the service category used by the call."
    ::= { atmTraceFilterRecordEntry 4 }
```

```
atmTraceFilterRecordInIf OBJECT-TYPE
    SYNTAX     InterfaceIndex
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The IfIndex of the incoming port on which this call was
        received by the ATM device."
    ::= { atmTraceFilterRecordEntry 5 }

atmTraceFilterRecordOutIf OBJECT-TYPE
    SYNTAX     InterfaceIndexOrZero
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The IfIndex of the outgoing port, if available, through which
        this call was routed to the network.  The distinguished value
        zero indicates that the call was rejected before any outgoing
        interface was chosen."
    ::= { atmTraceFilterRecordEntry 6 }

atmTraceFilterRecordCallingParty     OBJECT-TYPE
    SYNTAX       AtmAddr
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "Indicates the ATM adddress of the calling party in the
        connection or party."
    ::= { atmTraceFilterRecordEntry 7 }

atmTraceFilterRecordCalledParty      OBJECT-TYPE
    SYNTAX       AtmAddr
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "Indicates the ATM address of the called party in the
        connection or party."
    ::= { atmTraceFilterRecordEntry 8 }


atmTraceRecordGroup OBJECT IDENTIFIER ::= { atmTraceMIBObjects 5 }

atmTraceRecordTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF AtmTraceRecordEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "This table and the atmTraceInfoTable are used to display
        the path or connection trace results.  Trace
        information that is not repeated at each hop
        is shown in this table."
    ::= { atmTraceRecordGroup 1 }

atmTraceRecordEntry OBJECT-TYPE
    SYNTAX     AtmTraceRecordEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An entry representing a trace record for one new or existing
        connection or party."
    INDEX      { atmTraceRecordIndex }
    ::= { atmTraceRecordTable 1 }

AtmTraceRecordEntry ::=
    SEQUENCE {
          atmTraceRecordIndex                     AtmTraceRecordIndex,
          atmTraceRecordStatus                    INTEGER,
          atmTraceRecordCause                     Integer32,
          atmTraceRecordDiags                     OCTET STRING,
          atmTraceRecordTraceSourcePortId         PnniPortId,
          atmTraceRecordTraceSourceDlci           Integer32,
          atmTraceRecordTraceDestVpi              AtmVpIdentifier,
          atmTraceRecordTraceDestVci              AtmVcIdentifier,
          atmTraceRecordTraceDestCallRef          CallReference,
          atmTraceRecordTraceDestEndPtRef         AtmEndPointReference,
          atmTraceRecordTraceDestDlci             Integer32,
          atmTraceRecordTimeStamp                 TimeStamp,
          atmTraceRecordTraceDestReceiveLabel     MplsLabel,
          atmTraceRecordTraceDestTransmitLabel    MplsLabel
    }

atmTraceRecordIndex OBJECT-TYPE
    SYNTAX     AtmTraceRecordIndex
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An arbitrary integer used to distinguish between multiple
        trace records. "
    ::= { atmTraceRecordEntry 1 }

atmTraceRecordStatus OBJECT-TYPE
    SYNTAX    INTEGER {
                        traceInProgress(1),
                        traceCompletedNormally(2),
                        traceIncomplete(3),
                        traceExceededIELengthLimitations(4),
                        traceExceededMessageLengthLimitations(5),
                        traceLackResource(6)
                        }
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The returned trace status for this connection or party."
    ::= { atmTraceRecordEntry 2 }

atmTraceRecordCause      OBJECT-TYPE
    SYNTAX     Integer32 (0..255)
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "This object identifies the reason for the call failure.  When
        the call succeeds, the distinguished value zero is returned.
        When a PNNI Crankback information element is included in the
        last call clearing message, this object contains the crankback
        cause.  In all other cases, the values are the same as the
        cause code values defined for the Cause information element."
    REFERENCE
        "ATM Forum's UNI3.0/3.1 Specification. "
    ::= { atmTraceRecordEntry 3 }

atmTraceRecordDiags      OBJECT-TYPE
    SYNTAX     OCTET STRING(SIZE(0..17))
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "This object contains the contents of the diagnostics fields
        from the Cause information element.  When the value of
        atmTraceRecordCause is 49, 'Quality of Service unavailable',
        the diagnostics are taken from the PNNI Crankback information
        element instead of the Cause information element."
    REFERENCE
        "ATM Forum's UNI3.0/3.1 Specification. "
    ::= { atmTraceRecordEntry 4 }

atmTraceRecordTraceSourcePortId   OBJECT-TYPE
    SYNTAX     PnniPortId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The PNNI logical port ID identifying the trace source
        interface.  The distinguished value zero indicates that no
        trace source port ID was returned in the Trace transit list."
    ::= { atmTraceRecordEntry 5 }

atmTraceRecordTraceSourceDlci     OBJECT-TYPE
    SYNTAX     Integer32
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The DLCI used on the trace source interface.  The
        distinguished value zero indicates that no DLCI was included in
        the Trace transit list for the trace source interface."
    ::= { atmTraceRecordEntry 6 }

atmTraceRecordTraceDestVpi      OBJECT-TYPE
    SYNTAX     AtmVpIdentifier
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The VPI used on the preceding side of the trace destination
        interface.  The value zero is returned if no VPI was
        included in the Trace transit list for the trace destination
        interface."
    ::= { atmTraceRecordEntry 7 }

atmTraceRecordTraceDestVci      OBJECT-TYPE
    SYNTAX     AtmVcIdentifier
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The VCI used on the trace destination interface.  The
        distinguished value zero indicates that no VCI was included in
        the Trace transit list for the trace destination interface."
    ::= { atmTraceRecordEntry 8 }

atmTraceRecordTraceDestCallRef OBJECT-TYPE
    SYNTAX     CallReference
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The call reference used on the trace destination interface."
    ::= { atmTraceRecordEntry 9 }

atmTraceRecordTraceDestEndPtRef  OBJECT-TYPE
    SYNTAX     AtmEndPointReference
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The endpoint reference used on the trace destination
        interface."
     ::= { atmTraceRecordEntry 10 }

atmTraceRecordTraceDestDlci   OBJECT-TYPE
    SYNTAX     Integer32
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The DLCI used on the trace destination interface.  The
        distinguished value zero indicates that no DLCI was included in
        the Trace transit list for the trace destination interface."
    ::= { atmTraceRecordEntry 11 }

atmTraceRecordTimeStamp          OBJECT-TYPE
    SYNTAX     TimeStamp
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The time at which this record entry was created."
    ::= { atmTraceRecordEntry 12 }

atmTraceRecordTraceDestReceiveLabel      OBJECT-TYPE
    SYNTAX     MplsLabel
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The label for the interworking LSP used for packets
        transmitted in the direction of the tracing message
        (either SETUP, ADD PARTY, or TRACE CONNECTION) on the
        trace destination interface.
        The value zero is returned if no labels were
        included in the Trace transit list for the trace destination
        interface."
    ::= { atmTraceRecordEntry 13 }

atmTraceRecordTraceDestTransmitLabel      OBJECT-TYPE
    SYNTAX     MplsLabel
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The label for the interworking LSP used for packets
        transmitted in the opposite direction to that of
        the tracing message (either SETUP, ADD PARTY, or
        TRACE CONNECTION) on the trace destination interface.
        The value zero is returned if no labels were
        included in the Trace transit list for the trace destination
        interface."
    ::= { atmTraceRecordEntry 14 }

atmTraceInfoTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF AtmTraceInfoEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "The table in which the detailed trace information (i.e.,
        logical nodes, logical ports, VPI/VCIs, and Call/Endpoint
        References) of traced connections or parties are recorded."
    ::= { atmTraceRecordGroup 2}

atmTraceInfoEntry OBJECT-TYPE
    SYNTAX     AtmTraceInfoEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "Trace information for one hop of an existing or new connection
        or party.  This lists the nodes and ports traversed by the
        connection or party.  VPI/VCIs and Call/Endpoint References
        may also be included in this entry.  Each entry contains
        trace information added by one node.  If there are gaps in the
        Trace transit list due to the use of the Pass along request
        flag and the presence of nodes that do not support trace, the
        gaps will be between successive entries in this table.  Since
        the entries in this table are linked to the entries of the
        atmTraceRecordTable, the entries are added and removed from the
        table as and when the corresponding entries in the
        atmTraceRecordTable are added and removed."
    INDEX      { atmTraceRecordIndex,
                 atmTraceInfoSequenceIndex }
    ::= { atmTraceInfoTable 1 }

AtmTraceInfoEntry ::=
    SEQUENCE {
            atmTraceInfoSequenceIndex         Integer32,
            atmTraceInfoNodeId                PnniNodeId,
            atmTraceInfoOutgoingPortId        PnniPortId,
            atmTraceInfoIncomingVpi           AtmVpIdentifier,
            atmTraceInfoIncomingVci           AtmVcIdentifier,
            atmTraceInfoIncomingCallRef       CallReference,
            atmTraceInfoIncomingEndPtRef      AtmEndPointReference,
            atmTraceInfoRefusalIndicator      TruthValue,
            atmTraceInfoCrankBackRcvdAtDest   TruthValue,
            atmTraceInfoCrankBackGap          TruthValue,
            atmTraceInfoCrankBackIndicator    TruthValue,
            atmTraceInfoCrankBackBlockedTransitType  INTEGER,
            atmTraceInfoCrankBackBlockedTransitInfo  OCTET STRING,
            atmTraceInfoCrankBackCause        Integer32,
            atmTraceInfoReceiveLabel          MplsLabel,
            atmTraceInfoTransmitLabel         MplsLabel
    }

atmTraceInfoSequenceIndex OBJECT-TYPE
    SYNTAX     Integer32 (1..200)
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An index into the list of logical nodes / logical ports
        traversed by the connection or party.  The logical nodes and
        logical ports are given in order, as specified by this index."
    ::= { atmTraceInfoEntry 1 }

atmTraceInfoNodeId OBJECT-TYPE
    SYNTAX     PnniNodeId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The node ID of a logical node traversed by the connection
        or party."
    ::= { atmTraceInfoEntry 2 }

atmTraceInfoOutgoingPortId OBJECT-TYPE
    SYNTAX     PnniPortId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The port ID of the logical node identified in
        atmTraceInfoNodeId that identifies the logical port
        used to progress this connection or party towards the
        called party."
    ::= { atmTraceInfoEntry 3 }

atmTraceInfoIncomingVpi  OBJECT-TYPE
    SYNTAX     AtmVpIdentifier
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The VPI used on the succeeding side of the incoming interface
        of the node identified by atmTraceInfoNodeId.  The value zero
        is returned if no VPI was included in the Trace transit list.
        If there are no gaps in the Trace transit list, this is the
        VPI used on the other side of the interface identified by the
        atmTraceInfoNodeId and atmTraceInfoPortId under the previous
        atmTraceInfoSequenceIndex."
    ::= { atmTraceInfoEntry 4 }

atmTraceInfoIncomingVci  OBJECT-TYPE
    SYNTAX     AtmVcIdentifier
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The VCI used on the incoming interface of the node identified
        by atmTraceInfoNodeId.  The distinguished value zero indicates
        that no VCI was included in the Trace transit list.
        If there are no gaps in the Trace transit list, this is the
        VCI used on the interface identified by the atmTraceInfoNodeId
        and atmTraceInfoPortId under the previous
        atmTraceInfoSequenceIndex."
    ::= { atmTraceInfoEntry 5 }

atmTraceInfoIncomingCallRef  OBJECT-TYPE
    SYNTAX     CallReference
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The Call Reference used on the incoming interface of the node
        identified by atmTraceInfoNodeId.
        If there are no gaps in the Trace transit list, this is the
        call reference used on the interface identified by the
        atmTraceInfoNodeId and atmTraceInfoPortId under the previous
        atmTraceInfoSequenceIndex."
    ::= { atmTraceInfoEntry 6 }

atmTraceInfoIncomingEndPtRef  OBJECT-TYPE
    SYNTAX     AtmEndPointReference
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The Endpoint Reference used on the incoming interface of the
        node identified by atmTraceInfoNodeId.
        If there are no gaps in the Trace transit list, this is the
        endpoint reference used on the interface identified by the
        atmTraceInfoNodeId and atmTraceInfoPortId under the previous
        atmTraceInfoSequenceIndex."
    ::= { atmTraceInfoEntry 7 }

atmTraceInfoRefusalIndicator  OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "Indicates whether the node identified by the
        atmTraceInfoNodeId refused to participate in this trace."
    ::= { atmTraceInfoEntry 8 }

atmTraceInfoCrankBackRcvdAtDest  OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "Indicates whether a crankback was received at the node
        identified by the atmTraceInfoNodeId, when that node is the
        trace destination node and the trace destination interface
        is not a PNNI interface."
    ::= { atmTraceInfoEntry 9 }

atmTraceInfoCrankBackGap  OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "Indicates that the trace was propagated beyond the node
        identified by the atmTraceInfoNodeId, but was cranked back,
        and no trace information was returned by the node initiating
        crankback."
    ::= { atmTraceInfoEntry 10 }

atmTraceInfoCrankBackIndicator     OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "Indicates whether crankback information (octet group 16 of
        the Trace transit list information element) is present after
        the node identified by the atmTraceInfoNodeId, but before the
        next node identified in the Trace transit list information
        element."
    ::= { atmTraceInfoEntry 11 }

atmTraceInfoCrankBackBlockedTransitType   OBJECT-TYPE
    SYNTAX     INTEGER  {
                          blockedIncomingLink(1),
                          blockedNode(2),
                          blockedOutgoingLink(3)
                          }
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "This object identifies the type of blockage in case of a
        blocked call at the node identified by the atmTraceInfoNodeId.
        This object does not apply if the value of
        atmTraceInfoCrankBackIndicator is 'false'."
    ::= { atmTraceInfoEntry 12 }

atmTraceInfoCrankBackBlockedTransitInfo   OBJECT-TYPE
    SYNTAX     OCTET STRING
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "This object does not apply if the value of
        atmTraceInfoCrankBackIndicator is 'false'.

        When the value of atmTraceInfoCrankBackIndicator is 'true',
        this object includes the contents of the Blocked Transit
        Trace Information field from the Trace transit list
        Information element."
    REFERENCE
        "PNNI Addendum for Path and Connection Trace Version 1.0,
        Section 3.1"
    ::= { atmTraceInfoEntry 13 }

atmTraceInfoCrankBackCause   OBJECT-TYPE
    SYNTAX     Integer32
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "This object returns the PNNI crankback cause.  This object
        does not apply if atmTraceInfoCrankBackIndicator is set to
        'false'."
    ::= { atmTraceInfoEntry 14 }

atmTraceInfoReceiveLabel  OBJECT-TYPE
    SYNTAX     MplsLabel
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The label for the interworking LSP used for packets
        transmitted in the direction of the tracing message
        (either SETUP, ADD PARTY, or TRACE CONNECTION).
        If there are no gaps in the Trace transit list, this
        is from the atmTraceInfoNodeId
        under the previous atmTraceInfoSequenceIndex towards the
        atmTraceInfoNodeId under the current
        atmTraceInfoSequenceIndex."
    ::= { atmTraceInfoEntry 15 }

atmTraceInfoTransmitLabel  OBJECT-TYPE
    SYNTAX     MplsLabel
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "The label for the interworking LSP used for packets
        transmitted in the opposite direction to that of the
        the tracing message (either SETUP, ADD PARTY, or
        TRACE CONNECTION).
        If there are no gaps in the Trace transit list, this
        is from the atmTraceInfoNodeId
        under the current atmTraceInfoSequenceIndex towards the
        atmTraceInfoNodeId under the previous
        atmTraceInfoSequenceIndex."
    ::= { atmTraceInfoEntry 16 }

atmTraceInfoNeNscTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF AtmTraceInfoNeNscEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "The table in which records the Ne-NSCs which tag the
        network entity over which the connection was established
        on the interfaces of the node."
    ::= { atmTraceRecordGroup 3}

atmTraceInfoNeNscEntry OBJECT-TYPE
    SYNTAX     AtmTraceInfoNeNscEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "One of the Ne-NSCs which tag the network entity
        over which the connection was established
        on the interface of the node."
    INDEX      { atmTraceRecordIndex,
                 atmTraceInfoSequenceIndex,
                 atmTraceInfoNeNscInterface,
                 atmTraceInfoNeNscIndex }
    ::= { atmTraceInfoNeNscTable 1 }
```

```
AtmTraceInfoNeNscEntry ::=
    SEQUENCE {
           atmTraceInfoNeNscInterface     INTEGER,
           atmTraceInfoNeNscIndex         Integer32,
     atmTraceInfoNeNsc NetworkEntityNetworkServiceCategory
     }

atmTraceInfoNeNscInterface OBJECT-TYPE
    SYNTAX     INTEGER {
                        incoming(1),
                        outgoing(2)
                        }
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An index into the list of NSCs, used to identify the
         NSCs tagging resources on the incoming or outgoing
         interface"
    ::= { atmTraceInfoNeNscEntry 1 }

atmTraceInfoNeNscIndex OBJECT-TYPE
    SYNTAX     Integer32 (1..40)
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An index into the list of Ne-NSCs which tag the network
        entity over which the connection was established on the
        interface of the node.  The order of the Ne-NSCs
        is not important"
    ::= { atmTraceInfoNeNscEntry 2 }

atmTraceInfoNeNsc OBJECT-TYPE
    SYNTAX     NetworkEntityNetworkServiceCategory
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "One of the Ne-NSCs which tag the network entity over
        which the connection was established on the incoming
        interface of the node. "
    ::= { atmTraceInfoNeNscEntry 3 }

atmTraceInfoRpNscTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF AtmTraceInfoRpNscEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "The table in which records the Rp-NSCs which tag the
        resource in which the connection was established
        on the interfaces of the node."
    ::= { atmTraceRecordGroup 4}

atmTraceInfoRpNscEntry OBJECT-TYPE
    SYNTAX     AtmTraceInfoRpNscEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "One of the Rp-NSCs which tag the resource in
```

```
         which the connection was established on the
         interfaces of the node."
    INDEX      { atmTraceRecordIndex,
                 atmTraceInfoSequenceIndex,
                 atmTraceInfoRpNscInterface,
                 atmTraceInfoRpNscSequenceIndex }
    ::= { atmTraceInfoRpNscTable 1 }

AtmTraceInfoRpNscEntry ::=
    SEQUENCE {
           atmTraceInfoRpNscInterface     INTEGER,
           atmTraceInfoRpNscSequenceIndex Integer32,
     atmTraceInfoRpNsc ResourcePartitionNetworkServiceCategory
     }

atmTraceInfoRpNscInterface OBJECT-TYPE
    SYNTAX     INTEGER {
                         incoming(1),
                         outgoing(2)
                         }
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An index into the list of NSCs, used to identify the
         NSCs tagging resources on the incoming or outgoing
         interface"
    ::= { atmTraceInfoRpNscEntry 1 }

atmTraceInfoRpNscSequenceIndex OBJECT-TYPE
    SYNTAX     Integer32 (1..40)
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An index into the list of Rp-NSCs which tag the resource
        in which the connection was established on the
        interface of the node.  The order of the Rp-NSCs
        is not important"
    ::= { atmTraceInfoRpNscEntry 2 }

atmTraceInfoRpNsc OBJECT-TYPE
    SYNTAX     ResourcePartitionNetworkServiceCategory
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "One of the Rp-NSCs which tag the resource in
        which the connection was established on the
        interface of the node. "
    ::= { atmTraceInfoRpNscEntry 3 }

atmTraceIfGroup OBJECT IDENTIFIER ::= { atmTraceMIBObjects 6 }

atmTraceIfTable OBJECT-TYPE
    SYNTAX     SEQUENCE OF AtmTraceIfEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
```

```
        "This table is used to specify trace-related properties of a
        PNNI interface (e.g. whether a PNNI interface allows tracing
        over that interface)."
::= { atmTraceIfGroup 1 }

atmTraceIfEntry OBJECT-TYPE
    SYNTAX     AtmTraceIfEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An entry representing the trace-related properties of a
        PNNI interface."
    AUGMENTS   { pnniIfEntry }
    ::= { atmTraceIfTable 1 }

AtmTraceIfEntry ::=
    SEQUENCE {
          atmTraceIfTraceBoundary            TruthValue
    }

atmTraceIfTraceBoundary  OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
        "When this is a PNNI interface, indicates whether path
        and connection trace will be terminated or refused for
        outgoing or incoming, respectively, connections or parties on
        this interface.

        This object has no effect when this is not a PNNI interface."
    DEFVAL     { false }
    ::= { atmTraceIfEntry 1 }


-- Path and Connection Trace Traps

atmTraceMIBTrapsPrefix  OBJECT IDENTIFIER ::= { atmTraceMIB 2 }
atmTraceMIBTraps  OBJECT IDENTIFIER ::= { atmTraceMIBTrapsPrefix 0 }

atmTraceConnCompletion  NOTIFICATION-TYPE
    OBJECTS    {
                 atmTraceConnRecordIndex
                 }
    STATUS     current
    DESCRIPTION
        "An atmTraceConnCompletion trap is sent when enabled and either
        a TRACE CONNECTION ACKNOWLEDGE message is received at the trace
        source node, or after atmTraceConnFailTimeout has passed
        without any response (i.e., the connection trace fails)."
    ::= { atmTraceMIBTraps 1 }

atmTracePathTestCompletion  NOTIFICATION-TYPE
    OBJECTS    {
                 atmTracePathTestRecordIndex
                 }
    STATUS     current
    DESCRIPTION
        "An atmTracePathTestCompletion trap is sent when enabled and
        the test connection or test party becomes active on the
        trace source interface, or is cleared across the trace source
        interface."
    ::= { atmTraceMIBTraps 2 }

atmTracePathFilterTrap  NOTIFICATION-TYPE
    OBJECTS    {
                 atmTraceFilterRecordConnKind
                 }
    STATUS     current
    DESCRIPTION
        "An atmTracePathFilter trap is sent when the trap is enabled
        and a record is added to the atmTraceRecordTable and the
        atmTraceFilterRecordTable corresponding to this filter.  The
        atmTraceFilterTrapEnable object must be reset to 'true' before
        another atmTracePathFilter trap can be generated by the agent
        for this filter entry."
    ::= { atmTraceMIBTraps 3 }


-- conformance information

atmTraceMIBConformance
        OBJECT IDENTIFIER ::= { atmTraceMIB 3 }

atmTraceMIBCompliances
        OBJECT IDENTIFIER ::= { atmTraceMIBConformance 1 }

atmTraceMIBGroups
        OBJECT IDENTIFIER ::= { atmTraceMIBConformance 2 }

-- compliance statements

atmTraceMIBCompliance2 MODULE-COMPLIANCE
    STATUS     current
    DESCRIPTION
        "The compliance statement for entities which implement the
        PNNI Addendum for Path and Connection Trace Version 1.~~0~~1.

        Groups of objects required to support certain functionality
        are identified by the suffix MandatoryGroup.

        Groups of optional objects are identified by the suffix
        OptionalGroup."
    MODULE     -- this module
    MANDATORY-GROUPS
        { atmTraceMIBMandatoryGroup
          }

    GROUP atmTraceConnAndPathFilterMandatoryGroup
    DESCRIPTION
        "Required if connection trace or path trace using
        filtering of new connection and party establishment messages
        is supported."
    GROUP atmTracePathMandatoryGroup
    DESCRIPTION
        "Required if path trace is supported."

    GROUP atmTraceConnMandatoryGroup
    DESCRIPTION
        "Required if connection trace is supported."

    GROUP atmTracePathTestMandatoryGroup
    DESCRIPTION
        "Required if path trace using test connections and parties is
        supported."

    GROUP atmTracePathFilterMandatoryGroup
    DESCRIPTION
        "Required if path trace is supported using filtering of
        new connection and party establishment messages."

    GROUP atmTraceConnAndPathFilterPolicyMandatoryGroup
    DESCRIPTION
        "Required if connection trace or path trace using
        filtering of new connection and party establishment messages
        and policy tracing is supported."

    GROUP atmTraceConnPolicyMandatoryGroup
    DESCRIPTION
        "Required if connection trace and policy
        tracing is supported."

    GROUP atmTracePathTestPolicyMandatoryGroup
    DESCRIPTION
        "Required if path trace using test connections and
        parties and policy tracing is supported."

    GROUP atmTracePathFilterPolicyMandatoryGroup
    DESCRIPTION
        "Required if path trace using filtering of
        new connection and party establishment messages and
        policy tracing is supported."

    GROUP atmTraceConnAndPathFilterMplsMandatoryGroup
    DESCRIPTION
        "Required if connection trace or path trace using
        filtering of new connection and party establishment messages
        and interworking LSP label tracing is supported."

    GROUP atmTraceConnMplsMandatoryGroup
    DESCRIPTION
        "Required if connection trace and interworking
        LSP label tracing is supported."

    GROUP atmTracePathTestMplsMandatoryGroup
    DESCRIPTION
        "Required if path trace using test connections and
        parties and interworking LSP label tracing is
        supported."
    GROUP atmTracePathFilterMplsMandatoryGroup
    DESCRIPTION
        "Required if path trace using filtering of
        new connection and party establishment messages and
        interworking LSP label tracing is supported."

    OBJECT atmTraceTransitListMaximumSize
    MIN-ACCESS read-only
    DESCRIPTION
        "Maximum size of the Trace transit list information element
        larger than 1466 octets is optional."

    OBJECT atmTraceConnOrigConnType
    SYNTAX INTEGER { atmVcc(2) }
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to trace connections other than ATM VCCs
        (e.g. ATM VPCs, bearer-independent ATM connections,
        frame relay connections) is optional."

    OBJECT atmTraceConnOrigDirection
    SYNTAX INTEGER { incoming(1) }
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to trace connections and parties starting from the
        outgoing interface of a device is optional."

    OBJECT atmTracePathTestConnType
    SYNTAX INTEGER { atmVcc(2) }
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to generate test connections for path trace other
        than ATM VCCs (e.g. ATM VPCs, bearer-independent ATM
        connections) is optional."

    OBJECT atmTracePathTestClearCallAtTDest
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to generate test connections and parties that
        remain active after the path trace is completed is optional."

    OBJECT atmTraceFilterClearCallAtTDest
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to indicate call clearing at the trace destination
        node for calls that match a certain filter at the trace source
        node is optional."

    ::= { atmTraceMIBCompliances ~~1~~ 2 }


-- units of conformance

atmTraceMIBMandatoryGroup  OBJECT-GROUP
    OBJECTS {
              atmTraceMaxConcurrentRequests,
              atmTraceAvailableRequests,
                atmTraceTransitListMaximumSize,
                atmTraceRecordStatus,
                atmTraceRecordTraceSourcePortId,
                atmTraceRecordTimeStamp,
                atmTraceInfoNodeId,
                atmTraceInfoOutgoingPortId,
                atmTraceInfoRefusalIndicator
                }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when path or connection
        trace is supported."
    ::= { atmTraceMIBGroups 1 }

atmTraceMIBOptionalGroup  OBJECT-GROUP
    OBJECTS {
                atmTraceRecordTraceSourceDlci
                }
    STATUS    current
    DESCRIPTION
        "A collection of optional objects used for path and connection
        trace."
    ::= { atmTraceMIBGroups 2 }

atmTraceConnAndPathFilterMandatoryGroup  OBJECT-GROUP
    OBJECTS {
                atmTraceRecordTraceDestVpi,
                atmTraceRecordTraceDestVci,
                atmTraceRecordTraceDestCallRef,
                atmTraceRecordTraceDestEndPtRef,
                atmTraceRecordTraceDestDlci,
                atmTraceInfoIncomingVpi,
                atmTraceInfoIncomingVci,
                atmTraceInfoIncomingCallRef,
                atmTraceInfoIncomingEndPtRef
                }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when supporting connection
        trace or path trace using filtering of new connection and party
        establishment messages."
    ::= { atmTraceMIBGroups 3 }

atmTracePathMandatoryGroup  OBJECT-GROUP
    OBJECTS {
                atmTraceRecordCause,
                atmTraceRecordDiags,
                atmTraceInfoCrankBackRcvdAtDest,
                atmTraceInfoCrankBackGap,
                atmTraceInfoCrankBackIndicator,
                atmTraceInfoCrankBackBlockedTransitType,
                atmTraceInfoCrankBackBlockedTransitInfo,
                atmTraceInfoCrankBackCause
                }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when supporting path trace."
    ::= { atmTraceMIBGroups 4 }

atmTraceConnMandatoryGroup  OBJECT-GROUP
    OBJECTS {
                atmTraceConnOwner,
                atmTraceConnTraceSourceIf,
                atmTraceConnOrigConnType,
                atmTraceConnOrigVpi,
                atmTraceConnOrigVci,
                atmTraceConnEndPtRef,
                atmTraceConnOrigDirection,
                atmTraceConnTraceConnId,
                atmTraceConnTraceCallRef,
                atmTraceConnPassAlongRequest,
                atmTraceConnFailTimeout,
                atmTraceConnAgeTimeout,
                atmTraceConnRestart,
                atmTraceConnRecordIndex,
                atmTraceConnRowStatus             }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when connection trace is
        supported."
    ::= { atmTraceMIBGroups 5 }

atmTraceConnOptionalGroup  OBJECT-GROUP
    OBJECTS {
                atmTraceConnCallRef,
                atmTraceConnOrigDlci,
                atmTraceConnTrapOnCompletion
                }
    STATUS    current
    DESCRIPTION
        "A collection of optional objects used for connection trace."
    ::= { atmTraceMIBGroups 6 }

atmTracePathTestMandatoryGroup  OBJECT-GROUP
    OBJECTS {
                atmTracePathTestOwner,
                atmTracePathTestConnType,
                atmTracePathTestConnCastType,
                atmTracePathTestTraceSourceIf,
                atmTracePathTestP2MpNewConn,
                atmTracePathTestOrigVpi,
                atmTracePathTestOrigVci,
                atmTracePathTestCalledParty,
                atmTracePathTestTxTrafDescrIndex,
                atmTracePathTestRxTrafDescrIndex,
                atmTracePathTestClearCallAtTDest,
                atmTracePathTestTraceCrankback,
                atmTracePathTestPassAlongRequest,
                atmTracePathTestAgeTimeout,
                atmTracePathTestRestart,
                atmTracePathTestRecordIndex,
                atmTracePathTestRowStatus
                }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when path trace using test
        connections and test parties is supported."
    ::= { atmTraceMIBGroups 7 }

atmTracePathTestOptionalGroup  OBJECT-GROUP
    OBJECTS {
              atmTracePathTestCallingParty,
              atmTracePathTestTraceConnId,
              atmTracePathTestTraceCallRef,
              atmTracePathTestTrapOnCompletion
              }
    STATUS    current
    DESCRIPTION
        "A collection of optional objects used for path trace using
        test connections and test parties."
    ::= { atmTraceMIBGroups 8 }

atmTracePathFilterMandatoryGroup  OBJECT-GROUP
    OBJECTS {
              atmTraceFilterControl,
              atmTraceFilterOwner,
              atmTraceFilterConnKind,
              atmTraceFilterInIf,
              atmTraceFilterCalledPartyPrefix,
              atmTraceFilterCalledPartyLength,
              atmTraceFilterClearCallAtTDest,
              atmTraceFilterTraceCrankback,
              atmTraceFilterTraceConnId,
              atmTraceFilterTraceCallRef,
              atmTraceFilterPassAlongRequest,
              atmTraceFilterMaxRecords,
              atmTraceFilterStopTimeout,
              atmTraceFilterAgeTimeout,
              atmTraceFilterPurge,
              atmTraceFilterNumMatches,
              atmTraceFilterRowStatus,
              atmTraceFilterRecordConnKind,
              atmTraceFilterRecordConnCastType,
              atmTraceFilterRecordServiceCategory,
              atmTraceFilterRecordInIf,
              atmTraceFilterRecordOutIf,
              atmTraceFilterRecordCallingParty,
              atmTraceFilterRecordCalledParty
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when path trace is supported
        using filtering of new connection and party establishment
        messages."
    ::= { atmTraceMIBGroups 9 }

atmTracePathFilterOptionalGroup  OBJECT-GROUP
    OBJECTS {
              atmTraceFilterConnCastType,
              atmTraceFilterServiceCategory,
              atmTraceFilterOutIf,
              atmTraceFilterCallingPartyPrefix,
              atmTraceFilterCallingPartyLength,
              atmTraceFilterRecordCountDown,
              atmTraceFilterTrapEnable
              }
    STATUS    current
    DESCRIPTION
        "A collection of optional objects used for path trace using
        filtering of new connection and party establishment
        messages."
    ::= { atmTraceMIBGroups 10 }

atmTraceIfOptionalGroup  OBJECT-GROUP
    OBJECTS {
              atmTraceIfTraceBoundary
              }
    STATUS    current
    DESCRIPTION
        "A collection of optional objects used to configure PNNI
        interfaces to refuse incoming and terminate outgoing
        path and connection traces."
    ::= { atmTraceMIBGroups 11 }

atmTraceNotificationOptionalGroup  NOTIFICATION-GROUP
    NOTIFICATIONS {
                    atmTraceConnCompletion,
                    atmTracePathTestCompletion,
                    atmTracePathFilterTrap
                    }
    STATUS    current
    DESCRIPTION
        "A collection of optional notifications used for path and
        connection trace."
    ::= { atmTraceMIBGroups 12 }

atmTraceConnAndPathFilterPolicyMandatoryGroup  OBJECT-GROUP
    OBJECTS {
              atmTraceInfoNeNsc,
              atmTraceInfoRpNsc
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when supporting connection
        trace or path trace using filtering of new connection and party
        establishment messages and policy tracing."
    ::= { atmTraceMIBGroups 13 }

atmTraceConnPolicyMandatoryGroup  OBJECT-GROUP
    OBJECTS {
              atmTraceConnTraceNeNsc,
              atmTraceConnTraceRpNsc,
              atmTraceConnTraceIncoming
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when connection trace and
        policy tracing is supported."
    ::= { atmTraceMIBGroups 14 }

atmTracePathTestPolicyMandatoryGroup  OBJECT-GROUP
    OBJECTS {
              atmTracePathTestTraceNeNsc,
              atmTracePathTestTraceRpNsc,
              atmTracePathTestTraceIncoming
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when path trace using test
        connections and test parties and policy tracing are supported."
    ::= { atmTraceMIBGroups 15 }

atmTracePathFilterPolicyMandatoryGroup2  OBJECT-GROUP
    OBJECTS {
              atmTraceFilterPolicy,
              atmTraceFilterTraceNeNsc,
              atmTraceFilterTraceRpNsc,
              atmTraceFilterTraceIncoming
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when path trace
        using filtering of new connection and party establishment
        messages and policy tracing is supported."
    ::= { atmTraceMIBGroups 16 }


atmTraceConnAndPathFilterMplsMandatoryGroup  OBJECT-GROUP
    OBJECTS {
              atmTraceRecordTraceDestReceiveLabel,
              atmTraceRecordTraceDestTransmitLabel,
              atmTraceInfoReceiveLabel,
              atmTraceInfoTransmitLabel
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when supporting connection
        trace or path trace using filtering of new connection and party
        establishment messages and interworking LSP label tracing."
    ::= { atmTraceMIBGroups 17 }

atmTraceConnMplsMandatoryGroup  OBJECT-GROUP
    OBJECTS {
              atmTraceConnTraceLabels
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when connection trace and
        interworking LSP label tracing is supported."
    ::= { atmTraceMIBGroups 18 }

atmTracePathTestMplsMandatoryGroup2  OBJECT-GROUP
    OBJECTS {
              atmTracePathTestTraceLabels
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when path trace using test
        connections and test parties and interworking LSP label
        tracing are supported."
    ::= { atmTraceMIBGroups 19 }

atmTracePathFilterMplsMandatoryGroup2  OBJECT-GROUP
    OBJECTS {
              atmTraceFilterTraceLabels
              }
    STATUS    current
    DESCRIPTION
        "A collection of objects required when path trace
        using filtering of new connection and party establishment
        messages and interworking LSP label tracing is supported."
    ::= { atmTraceMIBGroups 20 }


-- deprecated definitions - compliance statements

atmTraceMIBCompliance MODULE-COMPLIANCE
    STATUS     deprecated
    DESCRIPTION
        "The compliance statement for entities which implement the
        PNNI Addendum for Path and Connection Trace Version 1.0.

        Groups of objects required to support certain functionality
        are identified by the suffix MandatoryGroup.

        Groups of optional objects are identified by the suffix
        OptionalGroup."
    MODULE     -- this module
    MANDATORY-GROUPS
        { atmTraceMIBMandatoryGroup
          }

    GROUP atmTraceConnAndPathFilterMandatoryGroup
    DESCRIPTION
        "Required if connection trace or path trace using
        filtering of new connection and party establishment messages
        is supported."

    GROUP atmTracePathMandatoryGroup
    DESCRIPTION
        "Required if path trace is supported."

    GROUP atmTraceConnMandatoryGroup
    DESCRIPTION
        "Required if connection trace is supported."

    GROUP atmTracePathTestMandatoryGroup
    DESCRIPTION
        "Required if path trace using test connections and parties is
        supported."

    GROUP atmTracePathFilterMandatoryGroup
    DESCRIPTION
        "Required if path trace is supported using filtering of
        new connection and party establishment messages."

    OBJECT atmTraceTransitListMaximumSize
    MIN-ACCESS read-only
    DESCRIPTION
        "Maximum size of the Trace transit list information element
        larger than 1466 octets is optional."

    OBJECT atmTraceConnOrigConnType
    SYNTAX INTEGER { atmVcc(2) }
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to trace connections other than ATM VCCs
        (e.g. ATM VPCs, bearer-independent ATM connections,
        frame relay connections) is optional."

    OBJECT atmTraceConnOrigDirection
    SYNTAX INTEGER { incoming(1) }
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to trace connections and parties starting from the
        outgoing interface of a device is optional."

    OBJECT atmTracePathTestConnType
    SYNTAX INTEGER { atmVcc(2) }
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to generate test connections for path trace other
        than ATM VCCs (e.g. ATM VPCs, bearer-independent ATM
        connections) is optional."

    OBJECT atmTracePathTestClearCallAtTDest
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to generate test connections and parties that
        remain active after the path trace is completed is optional."

    OBJECT atmTraceFilterClearCallAtTDest
    MIN-ACCESS read-only
    DESCRIPTION
        "The ability to indicate call clearing at the trace destination
        node for calls that match a certain filter at the trace source
        node is optional."

    ::= { atmTraceMIBCompliances 1 }


END
```