Network Working Group
Request for Comments: 3057
Category: Standards Track

K. Morneault
Cisco Systems
S. Rengasami
M. Kalla
Telcordia Technologies
G. Sidebottom
Nortel Networks

February 2001

ISDN Q.921-User Adaptation Layer

Status of this Memo

Copyright Notice

Abstract

This document defines a protocol for backhauling of ISDN Q.921 User messages over IP using the Stream Control Transmission Protocol (SCTP). This protocol would be used between a Signaling Gateway (SG) and Media Gateway Controller (MGC). It is assumed that the SG receives ISDN signaling over a standard ISDN interface.

Table of Contents

1. Introduction

In this document, the term Q.921-User refers to an upper layer which uses the services of Q.921, not the user side of ISDN interface [1]. Examples of the upper layer would be Q.931 and QSIG.

This section describes the need for ISDN Q.921-User Adaptation (IUA) layer protocol as well as how this protocol shall be implemented.

1.1 Scope

There is a need for Switched Circuit Network (SCN) signaling protocol delivery from an ISDN Signaling Gateway (SG) to a Media Gateway Controller (MGC) as described in the Framework Architecture for

Signaling Transport [4]. The delivery mechanism SHOULD meet the following criteria:

* Support for transport of the Q.921 / Q.931 boundary primitives
* Support for communication between Layer Management modules on SG and MGC
* Support for management of active associations between SG and MGC

This document supports both ISDN Primary Rate Access (PRA) as well as Basic Rate Access (BRA) including the support for both point-to-point and point-to-multipoint modes of communication. This support includes Facility Associated Signaling (FAS), Non-Facility Associated Signaling (NFAS) and NFAS with backup D channel. QSIG adaptation layer requirements do not differ from Q.931 adaptation layer, hence; the procedures described in this document are also applicable for a QSIG adaptation layer. For simplicity, only Q.931 will be mentioned in the rest of this document.

1.2 Terminology

Interface - For the purposes of this document an interface supports the relevant ISDN signaling channel. This signaling channel MAY be a 16 kbps D channel for an ISDN BRA as well as 64 kbps primary or backup D channel for an ISDN PRA. For QSIG, the signaling channel is a Qc channel.

Q.921-User - Any protocol normally using the services of the ISDN Q.921 (e.g., Q.931, QSIG, etc.).

Backhaul - A SG terminates the lower layers of an SCN protocol and backhauls the upper layer(s) to MGC for call processing. For the purposes of this document the SG terminates Q.921 and backhauls Q.931 to MGC.

Association - An association refers to a SCTP association. The association will provide the transport for the delivery of Q.921-User protocol data units and IUA adaptation layer peer messages.

Stream - A stream refers to an SCTP stream; a uni-directional logical channel established from one SCTP endpoint to another associated SCTP endpoint, within which all user messages are delivered in-sequence except for those submitted to the un-ordered delivery service.

Interface Identifier - The Interface Identifier identifies the physical interface at the SG for which the signaling messages are sent/received. The format of the Interface Identifier parameter can be text or integer, the values of which are assigned according to network operator policy. The values used are of local significance only, coordinated between the SG and ASP. Significance is not implied across SGs served by an AS.

Application Server (AS) - A logical entity serving a specific application instance. An example of an Application Server is a MGC handling the Q.931 and call processing for D channels terminated by the Signaling Gateways. Practically speaking, an AS is modeled at the SG as an ordered list of one or more related Application Server Processes (e.g., primary, secondary, tertiary).

Application Server Process (ASP) - A process instance of an Application Server. Examples of Application Server Processes are primary or backup MGC instances.

Fail-over - The capability to re-route signaling traffic as required between related ASPs in the event of failure or unavailability of the currently used ASP (e.g., from primary MGC to back-up MGC). Fail-over also applies upon the return to service of a previously unavailable process.

Layer Management - Layer Management is a nodal function that handles the inputs and outputs between the IUA layer and a local management entity.

Network Byte Order - Most significant byte first, a.k.a Big Endian.

Host - The computing platform that the ASP process is running on.

1.3 IUA Overview

The architecture that has been defined [4] for SCN signaling transport over IP uses multiple components, including an IP transport protocol, a signaling common transport protocol and an adaptation module to support the services expected by a particular SCN signaling protocol from its underlying protocol layer.

This document defines an adaptation module that is suitable for the transport of ISDN Q.921-User (e.g., Q.931) messages.

1.3.1 Example - SG to MGC

In a Signaling Gateway, it is expected that the ISDN signaling is received over a standard ISDN network termination. The SG then provides interworking of transport functions with IP Signaling Transport, in order to transport the Q.931 signaling messages to the MGC where the peer Q.931 protocol layer exists, as shown below:

```
******   ISDN       ******      IP      *******
* EP *---------------* SG *--------------* MGC *
******              ******              *******

+-----+                                 +-----+
|Q.931|              (NIF)              |Q.931|
+-----+           +----------+          +-----+
|     |           |     | IUA|          | IUA |
|     |           |     +----+          +-----+
|Q.921|           |Q.921|SCTP|          |SCTP |
|     |           |     +----+          +-----+
|     |           |     | IP |          | IP  |
+-----+           +-----+----+          +-----+

NIF  - Nodal Interworking Function
EP   - ISDN End Point
SCTP - Stream Control Transmission Protocol (Refer to [3])
IUA  - ISDN User Adaptation Layer Protocol
```

It is recommended that the IUA use the services of the Stream Control Transmission Protocol (SCTP) as the underlying reliable common signaling transport protocol. The use of SCTP provides the following features:

- explicit packet-oriented delivery (not stream-oriented)
- sequenced delivery of user messages within multiple streams, with an option for order-of-arrival delivery of individual user messages,
- optional multiplexing of user messages into SCTP datagrams,
- network-level fault tolerance through support of multi-homing at either or both ends of an association,
- resistance to flooding and masquerade attacks, and
- data segmentation to conform to discovered path MTU size

There are scenarios without redundancy requirements and scenarios in which redundancy is supported below the transport layer. In these cases, the SCTP functions above MAY NOT be a requirement and TCP can be used as the underlying common transport protocol.

1.3.2 Support for the management of SCTP associations between the SG and ASPs

The IUA layer at the SG maintains the availability state of all dynamically registered remote ASPs, in order to manage the SCTP Associations and the traffic between the SG and ASPs. As well, the active/inactive state of remote ASP(s) are also maintained. Active ASPs are those currently receiving traffic from the SG.

The IUA layer MAY be instructed by local management to establish an SCTP association to a peer IUA node. This can be achieved using the M-SCTP ESTABLISH primitive to request, indicate and confirm the establishment of an SCTP association with a peer IUA node.

The IUA layer MAY also need to inform local management of the status of the underlying SCTP associations using the M-SCTP STATUS request and indication primitive. For example, the IUA MAY inform local management of the reason for the release of an SCTP association, determined either locally within the IUA layer or by a primitive from the SCTP.

1.3.3 Signaling Network Architecture

A Signaling Gateway is used to support the transport of Q.921-User signaling traffic to one or more distributed ASPs (e.g., MGCs). Clearly, the IUA protocol is not designed to meet the performance and reliability requirements for such transport by itself. However, the conjunction of distributed architecture and redundant networks does allow for a sufficiently reliable transport of signaling traffic over IP. The IUA protocol is flexible enough to allow its operation and management in a variety of physical configurations, enabling Network Operators to meet their performance and reliability requirements.

To meet the ISDN signaling reliability and performance requirements for carrier grade networks, Network Operators SHOULD ensure that there is no single point of failure provisioned in the end-to-end network architecture between an ISDN node and an IP ASP.

Depending of course on the reliability of the SG and ASP functional elements, this can typically be met by the provision of redundant QOS-bounded IP network paths for SCTP Associations between SCTP End Points, and redundant Hosts, and redundant SGs. The distribution of ASPs within the available Hosts is also important. For a particular Application Server, the related ASPs SHOULD be distributed over at least two Hosts.

An example logical network architecture relevant to carrier-grade operation in the IP network domain is shown in Figure 1 below:

```
                                                          Host1
     ********                                         **************
     *      *_________________________________________*  ********  *
     *      *                                _________*  * ASP1 *  *
     *  SG1 *   SCTP Associations           |         *  ********  *
     *      *_______________________        |         *            *
     ********                       |       |         **************
                                    |       |
     ********                       |       |
     *      *_______________________________|
     *      *                       |
     *  SG2 *    SCTP Associations  |
     *      *____________           |
     *      *            |          |                     Host2
     ********            |          |                 **************
                         |          |_________________*  ********  *
                         |____________________________*  * ASP1 *  *
                                                      *  ********  *
                                                      *            *
                                                      **************
                                                              .
                                                              .
                                                              .
```

Figure 2 - Logical Model Example

For carrier grade networks, the failure or isolation of a particular ASP SHOULD NOT cause stable calls to be dropped. This implies that ASPs need, in some cases, to share the call state or be able to pass the call state between each other. However, this sharing or communication of call state information is outside the scope of this document.

1.3.4 ASP Fail-over Model and Terminology

The IUA layer supports ASP fail-over functions in order to support a high availability of call processing capability. All Q.921-User messages incoming to an SG are assigned to a unique Application Server, based on the Interface Identifier of the message.

The Application Server is, in practical terms, a list of all ASPs configured to process Q.921-User messages from certain Interface Identifiers. One or more ASPs in the list are normally active (i.e., handling traffic) while any others MAY be unavailable or inactive, to be possibly used in the event of failure or unavailability of the active ASP(s).

The fail-over model supports an n+k redundancy model, where n ASP(s) are the minimum number of redundant ASPs required to handle traffic and k ASPs are available to take over for a failed or unavailable ASP. Note that 1+1 active/standby redundancy is a subset of this model. A simplex 1+0 model is also supported as a subset, with no ASP redundancy.

To avoid a single point of failure, it is recommended that a minimum of two ASPs be in the list, resident in separate hosts and therefore available over different SCTP Associations. For example, in the network shown in Figure 2, all messages from a particular D Channel (Interface Identifier) could be sent to ASP1 in Host1 or ASP1 in Host2. The AS list at SG1 might look like the following:

```
   Interface Identifier(s) - Application Server #1
       ASP1/Host1  - State=Up, Active
       ASP1/Host2  - State=Up, Inactive
```

In this 1+1 redundancy case, ASP1 in Host1 would be sent any incoming message for the Interface Identifiers registered. ASP1 in Host2 would normally be brought to the active state upon failure of, or loss of connectivity to, ASP1/Host1. In this example, both ASPs are Up, meaning that the related SCTP association and far-end IUA peer is ready.

The AS List at SG1 might also be set up in load-share mode as shown below:

```
   Interface Identifier(s) - Application Server #1
       ASP1/Host1 - State=Up, Active
       ASP1/Host2 - State=Up, Active
```

In this case, both the ASPs would be sent a portion of the traffic.

In the process of fail-over, it is recommended that in the case of ASPs supporting call processing, stable calls do not get released. It is possible that calls in transition MAY fail, although measures of communication between the ASPs involved can be used to mitigate this problem. For example, the two ASPs MAY share call state via shared memory, or MAY use an ASP to ASP protocol to pass call state information. The ASP to ASP protocol is outside the scope of this document.

1.3.5 Client/Server Model

It is recommended that the SG and ASP be able to support both client and server operation. The peer endpoints using IUA SHOULD be configured so that one always takes on the role of client and the other the role of server for initiating SCTP associations. The default orientation would be for the SG to take on the role of server while the ASP is the client. In this case, ASPs SHOULD initiate the SCTP association to the SG.

The SCTP (and UDP/TCP) Registered User Port Number Assignment for IUA is 9900.

1.4 Services Provided by the IUA Layer

1.4.1 Support for transport of Q.921/Q.931 boundary primitives

In the backhaul scenario, the Q.921/Q.931 boundary primitives are exposed. IUA layer needs to support all of the primitives of this boundary to successfully backhaul Q.931.

This includes the following primitives [1]:

DL-ESTABLISH

The DL-ESTABLISH primitives are used to request, indicate and confirm the outcome of the procedures for establishing multiple frame operation.

DL-RELEASE

DL-RELEASE primitives are used to request, indicate, and confirm the outcome of the procedures for terminating a previously established multiple frame operation, or for reporting an unsuccessful establishment attempt.

DL-DATA

The DL-DATA primitives are used to request and indicate layer 3 (Q.931) messages which are to be transmitted, or have been received, by the Q.921 layer using the acknowledged information transfer service.

DL-UNIT DATA

The DL-UNIT DATA primitives are used to request and indicate layer 3 (Q.931) messages which are to be transmitted, by the Q.921 layer using the unacknowledged information transfer service.

1.4.2 Support for communication between Layer Management modules on SG and MGC

It is envisioned that the IUA layer needs to provide some services that will facilitate communication between Layer Management modules on the SG and MGC. These primitives are pointed out in [2], which are shown below:

M-TEI STATUS

The M-TEI STATUS primitives are used to request, confirm and indicate the status (assigned/unassigned) of a TEI.

M-ERROR

The M-ERROR primitive is used to indicate an error with a received IUA message (e.g., interface identifier value is not known to the SG).

1.4.3 Support for management of active associations between SG and MGC

A set of primitives between the IUA layer and the Layer Management are defined below to help the Layer Management manage the SCTP association(s) between the SG and MGC. The IUA layer can be instructed by the Layer Management to establish an SCTP association to a peer IUA node. This procedure can be achieved using the M-SCTP ESTABLISH primitive.

M-SCTP ESTABLISH

The M-SCTP ESTABLISH primitives are used to request, indicate, and confirm the establishment of an SCTP association to a peer IUA node.

M-SCTP RELEASE

The M-SCTP RELEASE primitives are used to request, indicate, and confirm the release of an SCTP association to a peer IUA node.

The IUA layer MAY also need to inform the status of the SCTP associations to the Layer Management. This can be achieved using the M-SCTP STATUS primitive.

M-SCTP STATUS

The M-SCTP STATUS primitives are used to request and indicate the status of the underlying SCTP association(s).

The Layer Management MAY need to inform the IUA layer of an AS/ASP status (i.e., failure, active, etc.), so that messages can be exchanged between IUA layer peers to stop traffic to the local IUA user. This can be achieved using the M-ASP STATUS primitive.

M-ASP STATUS

The ASP status is stored inside IUA layer on both the SG and MGC sides. The M-ASP STATUS primitive can be used by Layer Management to request the status of the Application Server Process from the IUA layer. This primitive can also be used to indicate the status of the Application Server Process.

M-ASP-UP

The M-ASP-UP primitive can be used by Layer Management to send a ASP Up message for the Application Server Process. It can also be used to generate an ASP Up Acknowledgement.

M-ASP-DOWN

The M-ASP-DOWN primitive can be used by Layer Management to send a ASP Down message for the Application Server Process. It can also be used to generate an ASP Down Acknowledgement.

M-ASP-ACTIVE

The M-ASP-UP primitive can be used by Layer Management to send a ASP Active message for the Application Server Process. It can also be used to generate an ASP Active Acknowledgement.

M-ASP-INACTIVE

The M-ASP-UP primitive can be used by Layer Management to send a ASP Inactive message for the Application Server Process. It can also be used to generate an ASP Inactive Acknowledgement.

M-AS STATUS

The M-AS STATUS primitive can be used by Layer Management to request the status of the Application Server. This primitive can also be used to indicate the status of the Application Server.

1.5 Functions Implemented by the IUA Layer

1.5.1 Mapping

The IUA layer MUST maintain a map of the Interface Identifier to a physical interface on the Signaling Gateway. A physical interface would be a T1 line, E1 line, etc., and could include the TDM timeslot. In addition, for a given interface the SG MUST be able to identify the associated signaling channel. IUA layers on both SG and MGC MAY maintain the status of TEIs and SAPIs.

The SG maps an Interface Identifier to an SCTP association/stream only when an ASP sends an ASP Active message for a particular Interface Identifier. It MUST be noted, however, that this mapping is dynamic and could change at any time due to a change of ASP state. This mapping could even temporarily be invalid, for example during failover of one ASP to another. Therefore, the SG MUST maintain the states of AS/ASP and reference them during the routing of an messages to an AS/ASP.

One example of the logical view of relationship between D channel, Interface Identifier, AS and ASP in the SG is shown below:

```
           /----------------------------------------------------+
          /   /----------------------------------------------------|--+
         /   /                                                     v  |
        /   /    +----+             act+-----+    +-------+ -+--+-|+--+-
D chan1-------->|IID |-+           +-->| ASP |--->| Assoc |       v
          /      +----+ |  +----+  |   +-----+    +-------+ -+--+--+--+-
         /              +->| AS |--+                        Streams
        /        +----+ |  +----+   stb+-----+
D chan2-------->|IID |-+              | ASP |
                 +----+               +-----+
```

where IID = Interface Identifier

Note that an ASP can be in more than one AS.

1.5.2 Status of ASPs

The IUA layer on the SG MUST maintain the state of the ASPs it is supporting. The state of an ASP changes because of reception of peer-to-peer messages (ASPM messages as described in Section 3.3.2) or reception of indications from the local SCTP association. ASP state transition procedures are described in Section 4.3.1.

At a SG, an Application Server list MAY contain active and inactive ASPs to support ASP load-sharing and fail-over procedures. When, for example, both a primary and a back-up ASP are available, IUA peer protocol is required to control which ASP is currently active. The ordered list of ASPs within a logical Application Server is kept updated in the SG to reflect the active Application Server Process(es).

Also the IUA layer MAY need to inform the local management of the change in status of an ASP or AS. This can be achieved using the M-ASP STATUS or M-AS STATUS primitives.

1.5.3 SCTP Stream Management

SCTP allows a user specified number of streams to be opened during the initialization. It is the responsibility of the IUA layer to ensure proper management of these streams. Because of the unidirectional nature of streams, an IUA layer is not aware of the stream number to Interface Identifier mapping of its peer IUA layer. Instead, the Interface Identifier is in the IUA message header.

The use of SCTP streams within IUA is recommended in order to minimize transmission and buffering delay, therefore improving the overall performance and reliability of the signaling elements. It is recommended that a separate SCTP stream is used for each D channel.

1.5.4 Seamless Network Management Interworking

The IUA layer on the SG SHOULD pass an indication of unavailability of the IUA-User (Q.931) to the local Layer Management, if the currently active ASP moves from the ACTIVE state. The Layer Management could instruct Q.921 to take some action, if it deems appropriate.

Likewise, if an SCTP association fails, the IUA layer on both the SG and ASP sides MAY generate Release primitives to take the data links out-of-service.

1.5.5 Congestion Management

If the IUA layer becomes congested (implementation dependent), it MAY stop reading from the SCTP association to flow control from the peer IUA.

1.6 Definition of IUA Boundaries

1.6.1 Definition of IUA/Q.921 boundary

DL-ESTABLISH
DL-RELEASE
DL-DATA
DL-UNIT DATA

1.6.2 Definition of IUA/Q.931 boundary

DL-ESTABLISH
DL-RELEASE
DL-DATA
DL-UNIT DATA

1.6.3 Definition of SCTP/IUA Boundary

An example of the upper layer primitives provided by SCTP are available in Reference [3] section 10.

1.6.4 Definition of IUA/Layer-Management Boundary

M-SCTP ESTABLISH request
Direction: LM -> IUA
Purpose: LM requests ASP to establish an SCTP association with an SG.

M-STCP ESTABLISH confirm
Direction: IUA -> LM
Purpose: ASP confirms to LM that it has established an SCTP association with an SG.

M-SCTP ESTABLISH indication
Direction: IUA -> LM
Purpose: SG informs LM that an ASP has established an SCTP association.

M-SCTP RELEASE request
Direction: LM -> IUA
Purpose: LM requests ASP to release an SCTP association with SG.

M-SCTP RELEASE confirm
Direction: IUA -> LM
Purpose: ASP confirms to LM that it has released SCTP association with SG.

M-SCTP RELEASE indication
Direction: IUA -> LM
Purpose: SG informs LM that ASP has released an SCTP association.

M-SCTP STATUS request
Direction: LM -> IUA
Purpose: LM requests IUA to report status of SCTP association.

M-SCTP STATUS indication
Direction: IUA -> LM
Purpose: IUA reports status of SCTP association.

M-ASP STATUS request
Direction: LM -> IUA
Purpose: LM requests SG to report status of remote ASP.

M-ASP STATUS indication
Direction: IUA -> LM
Purpose: SG reports status of remote ASP.

M-AS-STATUS request
Direction: LM -> IUA
Purpose: LM requests SG to report status of AS.

M-AS-STATUS indication
Direction: IUA -> LM
Purpose: SG reports status of AS.

M-NOTIFY indication
Direction: IUA -> LM
Purpose: ASP reports that it has received a NOTIFY message from its peer.

M-ERROR indication
Direction: IUA -> LM
Purpose: ASP or SG reports that it has received an ERROR message from its peer.

M-ASP-UP request
Direction: LM -> IUA
Purpose: LM requests ASP to start its operation and send an ASP UP message to the SG.

M-ASP-UP confirm
Direction: IUA -> LM
Purpose: ASP reports that is has received an ASP UP Acknowledgement message from the SG.

M-ASP-DOWN request
Direction: LM -> IUA
Purpose: LM requests ASP to stop its operation and send an ASP DOWN message to the SG.

M-ASP-DOWN confirm
Direction: IUA -> LM
Purpose: ASP reports that is has received an ASP DOWN Acknowledgement message from the SG.

M-ASP-ACTIVE request
Direction: LM -> IUA
Purpose: LM requests ASP to send an ASP ACTIVE message to the SG.

M-ASP-ACTIVE confirm
Direction: IUA -> LM
Purpose: ASP reports that is has received an ASP ACTIVE Acknowledgement message from the SG.

M-ASP-INACTIVE request
Direction: LM -> IUA
Purpose: LM requests ASP to send an ASP INACTIVE message to the SG.

M-ASP-INACTIVE confirm
Direction: IUA -> LM
Purpose: ASP reports that is has received an ASP INACTIVE Acknowledgement message from the SG.

M-TEI STATUS request
Direction: LM -> IUA
Purpose: LM requests ASP to send a TEI status request to the SG.

M-TEI STATUS indication
Direction: IUA -> LM
Purpose: ASP reports that is has received a TEI status indication from the SG.

M-TEI STATUS confirm
Direction: IUA -> LM
Purpose: ASP reports that is has received a TEI status confirm from the SG.

2.0 Conventions

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, NOT RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [RFC2119].

3.0 Protocol Elements

This section describes the format of various messages used in this protocol.

3.1 Common Message Header

The protocol messages for Q.921-User Adaptation require a message header which contains the adaptation layer version, the message type, and message length.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Version    |   Reserved    | Message Class | Message Type  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Message Length                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3 Common Header Format

All fields in an IUA message MUST be transmitted in the network byte order, unless otherwise stated.

3.1.1 Version

The version field contains the version of the IUA adaptation layer. The supported versions are the following:

Value	Version
1	Release 1.0

3.1.2 Message Classes and Types

The following List contains the valid Message Classes:

Message Class: 8 bits (unsigned integer)

0	Management (MGMT) Message [IUA/M2UA/M3UA/SUA]
1	Transfer Messages [M3UA]
2	SS7 Signalling Network Management (SSNM) Messages [M3UA/SUA]
3	ASP State Maintenance (ASPSM) Messages [IUA/M2UA/M3UA/SUA]
4	ASP Traffic Maintenance (ASPTM) Messages [IUA/M2UA/M3UA/SUA]
5	Q.921/Q.931 Boundary Primitives Transport (QPTM) Messages [IUA]
6	MTP2 User Adaptation (MAUP) Messages [M2UA]
7	Connectionless Messages [SUA]

```
     8      Connection-Oriented Messages [SUA]
  9 to 127  Reserved by the IETF
128 to 255  Reserved for IETF-Defined Message Class extensions
```

The following list contains the message names for the defined messages.

Q.921/Q.931 Boundary Primitives Transport (QPTM) Messages

```
     0        Reserved
     1        Data Request Message
     2        Data Indication Message
     3        Unit Data Request Message
     4        Unit Data Indication Message
     5        Establish Request
     6        Establish Confirm
     7        Establish Indication
     8        Release Request
     9        Release Confirm
    10        Release Indication
  11 to 127   Reserved by the IETF
 128 to 255   Reserved for IETF-Defined QPTM extensions
```

Application Server Process State Maintenance (ASPSM) messages

```
     0        Reserved
     1        ASP Up (UP)
     2        ASP Down (DOWN)
     3        Heartbeat (BEAT)
     4        ASP Up Ack (UP ACK)
     5        ASP Down Ack (DOWN ACK)
     6        Heatbeat Ack (BEAT ACK)
   7 to 127   Reserved by the IETF
 128 to 255   Reserved for IETF-Defined ASPSM extensions
```

Application Server Process Traffic Maintenance (ASPTM) messages

```
     0        Reserved
     1        ASP Active (ACTIVE)
     2        ASP Inactive (INACTIVE)
     3        ASP Active Ack (ACTIVE ACK)
     4        ASP Inactive Ack (INACTIVE ACK)
   5 to 127   Reserved by the IETF
 128 to 255   Reserved for IETF-Defined ASPTM extensions
```

Management (MGMT) Messages

0	Error (ERR)
1	Notify (NTFY)
2	TEI Status Request
3	TEI Status Confirm
4	TEI Status Indication
5 to 127	Reserved by the IETF
128 to 255	Reserved for IETF-Defined MGMT extensions

3.1.3 Reserved

The Reserved field is 8-bits. It SHOULD be set to all ’0’s and ignored by the receiver.

3.1.4 Message Length

The Message length defines the length of the message in octets, including the Common header.

3.1.5 Variable-Length Parameter Format

IUA messages consist of a Common Header followed by zero or more variable-length parameters, as defined by the message type. The variable-length parameters contained in a message are defined in a Tag-Length-Value format as shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Parameter Tag        |       Parameter Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
\                                                               \
/                       Parameter Value                         /
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Mandatory parameters MUST be placed before optional parameters in a message.

Parameter Tag: 16 bits (unsigned integer)

The Tag field is a 16 bit identifier of the type of parameter. It takes a value of 0 to 65534.

The value of 65535 is reserved for IETF-defined extensions. Values other than those defined in specific parameter description are reserved for use by the IETF.

Parameter Length: 16 bits (unsigned integer)

The Parameter Length field contains the size of the parameter in bytes, including the Parameter Tag, Parameter Length, and Parameter Value fields. The Parameter Length does not include any padding bytes.

Parameter Value: variable-length

The Parameter Value field contains the actual information to be transferred in the parameter.

The total length of a parameter (including Tag, Parameter Length and Value fields) MUST be a multiple of 4 bytes. If the length of the parameter is not a multiple of 4 bytes, the sender pads the Parameter at the end (i.e., after the Parameter Value field) with all zero bytes. The length of the padding is NOT included in the parameter length field. A sender SHOULD NEVER pad with more than 3 bytes. The receiver MUST ignore the padding bytes.

3.2 IUA Message Header

In addition to the common message header, there will be a specific message header for QPTM and the TEI Status MGMT messages. The IUA message header will immediately follow the Common header in these messages.

This message header will contain the Interface Identifier and Data Link Connection Identifier (DLCI). The Interface Identifier identifies the physical interface terminating the signaling channel at the SG for which the signaling messages are sent/received. The format of the Interface Identifier parameter can be text or integer. The Interface Identifiers are assigned according to network operator policy. The integer values used are of local significance only, coordinated between the SG and ASP.

The integer formatted Interface Identifier MUST be supported. The text formatted Interface Identifier MAY optionally be supported.

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |          Tag (0x1)            |             Length            |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                Interface Identifier (integer)                 |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |          Tag (0x5)            |             Length=8          |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |           DLCI                |              Spare            |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4 IUA Message Header (Integer-based Interface Identifier)

The Tag value for the Integer-based Interface Identifier is 0x1. The length is always set to a value of 8.

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |          Tag (0x3)            |             Length            |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

 |                  Interface Identifier (text)                  |

 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |          Tag (0x5)            |             Length=8          |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |           DLCI                |             Spare             |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5 IUA Message Header (Text-based Interface Identifier)

The Tag value for the Text-based Interface Identifier is 0x3. The length is variable.

The DLCI format is shown below in Figure 6.

```
    0     1     2     3     4     5     6     7
 +-----+-----+-----+-----+-----+-----+-----+-----+
 |  0  | SPR |      SAPI                         |
 +-----------------------------------------------+
 |  1  |            TEI                          |
 +-----------------------------------------------+
```

Figure 6 DLCI Format

SPR: Spare 2nd bit in octet 1, (1 bit)

SAPI: Service Access Point Identifier, 3rd through 8th bits in octet 1 (6 bits)

TEI: Terminal Endpoint Identifier, 2nd through 8th bits in octet 2 (7 bits)

The DLCI field (including the SAPI and TEI) is coded in accordance with Q.921.

3.3 IUA Messages

The following section defines the messages and parameter contents. The IUA messages will use the common message header (Figure 3) and the IUA message header (Figure 4 and Figure 5).

3.3.1 Q.921/Q.931 Boundary Primitives Transport (QPTM) Messages

3.3.1.1 Establish Messages (Request, Confirm, Indication)

The Establish Messages are used to establish a data link on the signaling channel or to confirm that a data link on the signaling channel has been established. The MGC controls the state of the D channel. When the MGC desires the D channel to be in-service, it will send the Establish Request message.

When the MGC sends an IUA Establish Request message, the MGC MAY start a timer. This timer would be stopped upon receipt of an IUA Establish Confirm or Establish Indication. If the timer expires, the MGC would re-send the IUA Establish Request message and restart the timer. In other words, the MGC MAY continue to request the establishment of the data link on periodic basis until the desired state is achieved or take some other action (notify the Management Layer).

When the SG receives an IUA Establish Request from the MGC, the SG shall send the Q.921 Establish Request primitive to the its Q.921 entity. In addition, the SG shall map any response received from the Q.921 entity to the appropriate message to the MGC. For example, if the Q.921 entity responds with a Q.921 Establish Confirm primitive, the IUA layer shall map this to an IUA Establish Confirm message. As another example, if the IUA Layer receives a Q.921 Release Confirm or Release Indication as an apparent response to the Q.921 Establish Request primitive, the IUA Layer shall map these to the corresponding IUA Release Confirm or Release Indication messages.

The Establish messages contain the common message header followed by IUA message header. It does not contain any additional parameters.

3.3.1.2 Release Messages (Request, Indication, Confirmation)

The Release Request message is used to release the data link on the signaling channel. The Release Confirm and Indication messages are used to indicate that the data link on the signaling channel has been released.

If a response to the Release Request message is not received, the MGC MAY resend the Release Request message. If no response is received, the MGC can consider the data link as being released. In this case, signaling traffic on that D channel is not expected from the SG and signaling traffic will not be sent to the SG for that D channel.

The Release messages contain the common message header followed by IUA message header. The Release confirm message is in response to a Release Request message and it does not contain any additional parameters. The Release Request and Indication messages contain the following parameter:

REASON

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0xf)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Reason                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The valid values for Reason are shown in the following table.

Define	Value	Description
RELEASE_MGMT	0x0	Management layer generated release.
RELEASE_PHYS	0x1	Physical layer alarm generated release.
RELEASE_DM	0x2	Specific to a request. Indicates Layer 2 SHOULD release and deny all requests from far end to establish a data link on the signaling channel (i.e., if SABME is received send a DM)
RELEASE_OTHER	0x3	Other reasons

Note: Only RELEASE_MGMT, RELEASE_DM and RELEASE_OTHER are valid reason codes for a Release Request message.

3.3.1.3 Data Messages (Request, Indication)

The Data message contains an ISDN Q.921-User Protocol Data Unit (PDU) corresponding to acknowledged information transfer service.

The Data messages contain the common message header followed by IUA message header. The Data message contains the following parameters:

PROTOCOL DATA

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0xe)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                         Protocol Data                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The protocol data contains upper layer signaling message e.g. Q.931, QSIG.

3.3.1.4 Unit Data Messages (Request, Indication)

The Unit Data message contains an ISDN Q.921-User Protocol Data Unit (PDU) corresponding to unacknowledged information transfer service.

The Unit Data messages contain the common message header followed by IUA message header. The Unit Data message contains the following parameters

PROTOCOL DATA

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0xe)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                         Protocol Data                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.3.2 Application Server Process Maintenance (ASPM) Messages

The ASPM messages will only use the common message header.

3.3.2.1 ASP Up (ASPUP)

The ASP Up (ASPUP) message is sent by an ASP to indicate to an SG that it is ready to receive traffic or maintenance messages.

The ASPUP message contains the following parameters:

Info String (optional)

The format for ASPUP Message parameters is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0x4)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                          INFO String*                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The optional INFO String parameter can carry any meaningful 8-bit ASCII character string along with the message. Length of the INFO String parameter is from 0 to 255 characters. No procedures are presently identified for its use but the INFO String MAY be used for debugging purposes.

3.3.2.2 ASP Up Ack

The ASP Up Ack message is used to acknowledge an ASP Up message received from a remote IUA peer.

The ASPUP Ack message contains the following parameters:

INFO String (optional)

The format for ASPUP Ack Message parameters is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0x4)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                          INFO String*                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format and description of the optional Info String parameter is the same as for the ASP Up message (See Section 3.3.3.1).

3.3.2.3 ASP Down (ASPDN)

The ASP Down (ASPDN) message is sent by an ASP to indicate to an SG that it is NOT ready to receive traffic or maintenance messages.

The ASPDN message contains the following parameters:

Reason
INFO String (Optional)

The format for the ASPDN message parameters is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0xa)           |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Reason                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0x4)           |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                         INFO String*                          |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format and description of the optional Info String parameter is the same as for the ASP Up message (See Section 3.3.3.1.).

The Reason parameter indicates the reason that the remote IUA adaptation layer is unavailable. The valid values for Reason are shown in the following table.

Value	Description
0x1	Management Inhibit

If a ASP is removed from Management Inhibit, the ASP will send an ASP Up message.

3.3.2.4 ASP Down Ack

The ASP Down Ack message is used to acknowledge an ASP Down message received from a remote IUA peer.

The ASP Down Ack message contains the following parameters:

Reason
INFO String (Optional)

The format for the ASP Down Ack message parameters is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0xa)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Reason                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0x4)            |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                         INFO String*                          |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format and description of the optional Info String parameter is the same as for the ASP Up message (See Section 3.3.2.1.).

The format of the Reason parameter is the same as for the ASP Down message (See Section 3.3.2.3).

3.3.2.5 ASP Active (ASPAC)

The ASPAC message is sent by an ASP to indicate to an SG that it is Active and ready to be used.

The ASPAC message contains the following parameters

- Traffic Mode Type (Mandatory)
- Interface Identifier (Optional)
  - Combination of integer and integer ranges, OR
  - string (text formatted)
- INFO String (Optional)

The format for the ASPAC message using integer formatted Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0xb)           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Traffic Mode Type                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Tag (0x1=integer)         |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                      Interface Identifiers*                   |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Tag (0x8=integer range)    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier Start1*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier Stop1*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier Start2*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier Stop2*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        .                                                         .
        .                                                         .
        .                                                         .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier StartN*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier StopN*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|              Additional Interface Identifiers                 |
|                    of Tag Type 0x1 or 0x8                     |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Tag (0x4)             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                          INFO String*                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format for the ASPAC message using text formatted (string) Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0xb)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Traffic Mode Type                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Tag (0x3=string)         |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                      Interface Identifier*                    |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|               Additional Interface Identifiers                |
|                       of Tag Type 0x3                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Tag (0x4)             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                           INFO String*                        |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Traffic Mode Type parameter identifies the traffic mode of operation of the ASP within an AS. The valid values for Type are shown in the following table:

Value	Description
0x1	Over-ride
0x2	Load-share

Within a particular Interface Identifier, only one Traffic Mode Type can be used. The Over-ride value indicates that the ASP is operating in Over-ride mode, where the ASP takes over all traffic in an Application Server (i.e., primary/back-up operation), over-riding any currently active ASPs in the AS. In Load-share mode, the ASP will share in the traffic distribution with any other currently active ASPs.

The optional Interface Identifiers parameter contains a list of Interface Identifier integers (Type 0x1 or Type 0x8) or text strings (Type 0x3) indexing the Application Server traffic that the sending ASP is configured/registered to receive. If integer formatted

Interface Identifiers are being used, the ASP can also send ranges of Interface Identifiers (Type 0x8). Interface Identifier types Integer (0x1) and Integer Range (0x8) are allowed in the same message. Text formatted Interface Identifiers (0x3) cannot be used with either Integer (0x1) or Integer Range (0x8) types.

If no Interface Identifiers are included, the message is for all provisioned Interface Identifiers within the AS(s) in which the ASP is provisioned. If only a subset of Interface Identifiers are included, the ASP is noted as Active for all the Interface Identifiers provisioned for that AS.

Note: If the optional Interface Identifier parameter is present, the integer formatted Interface Identifier MUST be supported, while the text formatted Interface Identifier MAY be supported.

The format and description of the optional Info String parameter is the same as for the ASP Up message (See Section 3.3.2.1.).

An SG that receives an ASPAC with an incorrect Traffic Mode Type for a particular Interface Identifier will respond with an Error Message (Cause: Unsupported Traffic Handling Mode).

3.3.2.6 ASP Active Ack

The ASPAC Ack message is used to acknowledge an ASP-Active message received from a remote IUA peer.

The ASPAC Ack message contains the following parameters:

- Traffic Mode Type (Mandatory)
- Interface Identifier (Optional)
  - Combination of integer and integer ranges, OR
  - string (text formatted)
- INFO String (Optional)

The format for the ASPAC Ack message with Integer-formatted Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0xb)           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Traffic Mode Type                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Tag (0x1=integer)         |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                     Interface Identifiers*                    |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Tag (0x8=integer range)    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier Start1*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier Stop1*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier Start2*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier Stop2*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        .                                                           .
        .                                                           .
        .                                                           .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier StartN*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier StopN*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|              Additional Interface Identifiers                 |
|                    of Tag Type 0x1 or 0x8                     |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Tag (0x4)             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                          INFO String*                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format for the ASP Active Ack message using text formatted (string) Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0xb)           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Traffic Mode Type                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Tag (0x3=string)        |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                      Interface Identifier*                    |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|               Additional Interface Identifiers                |
|                         of Tag Type 0x3                       |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Tag (0x4)             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                           INFO String*                        |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the Traffic Mode Type and Interface Identifier parameters is the same as for the ASP Active message (See Section 3.3.2.5).

The format and description of the optional Info String parameter is the same as for the ASP Up message (See Section 3.3.2.1.).

3.3.2.7 ASP Inactive (ASPIA)

The ASPIA message is sent by an ASP to indicate to an SG that it is no longer an active ASP to be used from within a list of ASPs. The SG will respond with an ASPIA Ack message and either discard incoming messages or buffer for a timed period and then discard.

The ASPIA message contains the following parameters

Traffic Mode Type (Mandatory)
Interface Identifiers (Optional)
- Combination of integer and integer ranges, OR
- string (text formatted)

INFO String (Optional)

The format for the ASP Inactive message parameters using Integer formatted Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0xb)           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Traffic Mode Type                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Tag (0x1=integer)         |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                     Interface Identifiers*                    |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Tag (0x8=integer range)    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier Start1*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier Stop1*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier Start2*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier Stop2*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        .                                                           .
        .                                                           .
        .                                                           .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier StartN*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier StopN*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|              Additional Interface Identifiers                 |
|                    of Tag Type 0x1 or 0x8                     |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0x4)           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                          INFO String*                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format for the ASP Inactive message using text formatted (string) Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0xb)           |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Traffic Mode Type                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Tag (0x3=string)        |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                      Interface Identifier*                    |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|               Additional Interface Identifiers                |
|                      of Tag Type 0x3                          |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Tag (0x4)             |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                         INFO String*                          |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Traffic Mode Type parameter identifies the traffic mode of operation of the ASP within an AS. The valid values for Traffic Mode Type are shown in the following table:

Value	Description
0x1	Over-ride
0x2	Load-share

The format and description of the optional Interface Identifiers and Info String parameters is the same as for the ASP Active message (See Section 3.3.2.3.).

The optional Interface Identifiers parameter contains a list of Interface Identifier integers or text strings indexing the Application Server traffic that the sending ASP is configured/registered to receive, but does not want to receive at this time.

3.3.2.8 ASP Inactive Ack

The ASP Inactive (ASPIA) Ack message is used to acknowledge an ASP Inactive message received from a remote IUA peer.

The ASPIA Ack message contains the following parameters:

Traffic Mode Type (Mandatory)
Interface Identifiers (Optional)
- Combination of integer and integer ranges, OR
- string (text formatted)

INFO String (Optional)

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |           Tag (0xb)           |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                      Traffic Mode Type                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Tag (0x1=integer)         |            Length             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    |                     Interface Identifiers*                    |

    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Tag (0x8=integer range)    |            Length             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                  Interface Identifier Start1*                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                   Interface Identifier Stop1*                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                  Interface Identifier Start2*                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                   Interface Identifier Stop2*                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           .                                                           .
           .                                                           .
           .                                                           .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                  Interface Identifier StartN*                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                   Interface Identifier StopN*                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    |               Additional Interface Identifiers                |
    |                     of Tag Type 0x1 or 0x8                    |

    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Tag (0x4)             |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    |                          INFO String*                         |

    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format for the ASP Inactive Ack message using text formatted (string) Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0xb)           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Traffic Mode Type                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Tag (0x3=string)        |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                      Interface Identifier*                    |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|               Additional Interface Identifiers                |
|                        of Tag Type 0x3                        |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Tag (0x4)             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                          INFO String*                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the Traffic Mode Type and Interface Identifier parameters is the same as for the ASP Inactive message (See Section 3.3.2.7).

The format and description of the optional Info String parameter is the same as for the ASP Up message (See Section 3.3.2.1).

3.3.2.9 Heartbeat (BEAT)

The Heartbeat message is optionally used to ensure that the IUA peers are still available to each other. It is recommended for use when the IUA runs over a transport layer other than the SCTP, which has its own heartbeat.

The BEAT message contains the following parameters:

    Heartbeat Data   Optional

The format for the BEAT message is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Tag = 9            |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
\                                                               \
|                       Heartbeat Data *                        |
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Heartbeat Data parameter contents are defined by the sending node. The Heartbeat Data could include, for example, a Heartbeat Sequence Number and, or Timestamp. The receiver of a Heartbeat message does not process this field as it is only of significance to the sender. The receiver MUST respond with a Heartbeat Ack message.

3.3.2.10 Heartbeat Ack (BEAT-Ack)

The Heartbeat Ack message is sent in response to a received Heartbeat message. It includes all the parameters of the received Heartbeat message, without any change.

3.3.3 Layer Management (MGMT) Messages

3.3.3.1 Error (ERR)

The Error message is used to notify a peer of an error event associated with an incoming message. For example, the message type might be unexpected given the current state, or a parameter value might be invalid.

The Error message will only have the common message header. The Error message contains the following parameters:

    Error Code
    Diagnostic Information (optional)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0xc)           |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Error Code                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Tag (0x7)           |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                     Diagnostic Information*                   |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Error Code parameter indicates the reason for the Error Message. The Error parameter value can be one of the following values:

Invalid Version	0x01
Invalid Interface Identifier	0x02
Unsupported Message Class	0x03
Unsupported Message Type	0x04
Unsupported Traffic Handling Mode	0x05
Unexpected Message	0x06
Protocol Error	0x07
Unsupported Interface Identifier Type	0x08
Invalid Stream Identifier	0x09
Unassigned TEI	0x0a
Unrecognized SAPI	0x0b
Invalid TEI, SAPI combination	0x0c

The "Invalid Version" error would be sent if a message was received with an invalid or unsupported version. The Error message would contain the supported version in the Common header. The Error message could optionally provide the supported version in the Diagnostic Information area.

The "Invalid Interface Identifier" error would be sent by a SG if an ASP sends a message with an invalid (unconfigured) Interface Identifier value.

The "Unsupported Traffic Handling Mode" error would be sent by a SG if an ASP sends an ASP Active with an unsupported Traffic Handling Mode. An example would be a case in which the SG did not support load-sharing.

The "Unexpected Message" error would be sent by an ASP if it received a QPTM message from an SG while it was in the Inactive state (the ASP could optionally drop the message and not send an Error). It would also be sent by an ASP if it received a defined and recognized message that the SG is not expected to send (e.g., if the MGC receives an IUA Establish Request message).

The "Protocol Error" error would be sent for any protocol anomaly (i.e., a bogus message).

The "Invalid Stream Identifier" error would be sent if a message was received on an unexpected SCTP stream (i.e., a MGMT message was received on a stream other than "0").

The "Unsupported Interface Identifier Type" error would be sent by a SG if an ASP sends a Text formatted Interface Identifier and the SG only supports Integer formatted Interface Identifiers. When the ASP receives this error, it will need to resend its message with an Integer formatted Interface Identifier.

The "Unsupported Message Type" error would be sent if a message with an unexpected or unsupported Message Type is received.

The "Unsupported Message Class" error would be sent if a message with an unexpected or unsupported Message Class is received.

The "Unassigned TEI" error may be used when the SG receives an IUA message that includes a TEI which has not been assigned or recognized for use on the indicated ISDN D-channel.

The "Unrecognized SAPI" error would handle the case of using a SAPI that is not recognized by the SG. The "Invalid TEI, SAPI combination" error identify errors where the TEI is assigned and the the SAPI is recognized, but the combination is not valid for the interface (e.g., on a BRI the MGC tries to send Q.921 Management messages via IUA when Layer Management at the SG SHOULD be performing this function).

The optional Diagnostic information can be any information germane to the error condition, to assist in identification of the error condition. To enhance debugging, the Diagnostic information could contain the first 40 bytes of the offending message.

3.3.3.2 Notify (NTFY)

The Notify message used to provide an autonomous indication of IUA events to an IUA peer.

The Notify message will only use the common message header. The Notify message contains the following parameters:

Status Type
Status Identification
Interface Identifiers (Optional)
INFO String (Optional)

The format for the Notify message with Integer-formatted Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0xd)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Status Type            |    Status Identification      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Tag (0x1=integer)         |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                     Interface Identifiers*                    |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Tag (0x8=integer range)    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier Start1*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier Stop1*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier Start2*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier Stop2*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        .                                                           .
        .                                                           .
        .                                                           .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Interface Identifier StartN*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Interface Identifier StopN*                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|              Additional Interface Identifiers                 |
|                    of Tag Type 0x1 or 0x8                     |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Tag (0x4)             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                         INFO String*                          |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format for the Notify message with Text-formatted Interface Identifiers is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0xd)            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Status Type            |    Status Identification      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Tag (0x3=string)        |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                     Interface Identifier*                     |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|              Additional Interface Identifiers                 |
|                        of Tag Type 0x3                        |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Tag (0x4)             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

|                          INFO String*                         |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Status Type parameter identifies the type of the Notify message. The following are the valid Status Type values:

Value	Description
0x1	Application Server state change (AS_State_Change)
0x2	Other

The Status Identification parameter contains more detailed information for the notification, based on the value of the Status Type. If the Status Type is AS_State_Change the following Status Identification values are used:

Value	Description
1	Application Server Down (AS_Down)
2	Application Server Inactive (AS_Inactive)
3	Application Server Active (AS_Active)
4	Application Server Pending (AS_Pending)

These notifications are sent from an SG to an ASP upon a change in status of a particular Application Server. The value reflects the new state of the Application Server.

If the Status Type is Other, then the following Status Information values are defined:

Value	Description
1	Insufficient ASP resources active in AS
2	Alternate ASP Active

These notifications are not based on the SG reporting the state change of an ASP or AS. In the Insufficient ASP Resources case, the SG is indicating to an "Inactive" ASP(s) in the AS that another ASP is required in order to handle the load of the AS (Load-sharing mode). For the Alternate ASP Active case, an ASP is informed when an alternate ASP transitions to the ASP-Active state in Over-ride mode.

The format and description of the optional Interface Identifiers and Info String parameters is the same as for the ASP Active message (See Section 3.3.2.3.).

3.3.3.3 TEI Status Messages (Request, Confirm and Indication)

The TEI Status messages are exchanged between IUA layer peers to request, confirm and indicate the status of a particular TEI.

The TEI Status messages contain the common message header followed by IUA message header. The TEI Status Request message does not contain any additional parameters.

In the integrated ISDN Layer 2/3 model (e.g., in traditional ISDN switches), it is assumed that the Layer Management for the Q.921 Layer and the Q.931 layer are co-located. When backhauling ISDN, this assumption is not necessarily valid. The TEI status messages allow the two Layer Management entities to communicate the status of the TEI. In addition, knowing that a TEI is in service allows the ASP to request the SG to establish the datalink to the terminal (via the IUA Establish message) for signaling if the ASP wants to be in control of data link establishment. Another use of the TEI status procedure is where the Layer Management at the ASP can prepare for send/receive signaling to/from a given TEI and confirm/verify the establishment of a datalink to that TEI. For example, if a datalink is established for a TEI that the ASP did not know was assigned, the ASP can check to see whether it was assigned or whether there was an error in the signaling message. Also, knowing that a TEI is out of service, the ASP need not request the SG to establish a datalink to that TEI.

The TEI Status Indication, and Confirm messages contain the following parameter:

STATUS

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Tag (0x10)           |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Status                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The valid values for Status are shown in the following table.

Define	Value	Description
ASSIGNED	0x0	TEI is considered assigned by Q.921
UNASSIGNED	0x1	TEI is considered unassigned by Q.921

4.0 Procedures

The IUA layer needs to respond to various primitives it receives from other layers as well as messages it receives from the peer IUA layer. This section describes various procedures involved in response to these events.

4.1 Procedures to support service in section 1.4.1

These procedures achieve the IUA layer's "Transport of Q.921/Q.931 boundary" service.

4.1.1 Q.921 or Q.931 primitives procedures

On receiving these primitives from the local layer, the IUA layer will send the corresponding QPTM message (Data, Unit Data, Establish, Release) to its peer. While doing so, the IUA layer needs to fill various fields of the common and specific headers correctly. In addition the message needs to be sent on the SCTP stream that corresponds to the D channel (Interface Identifier).

4.1.2 QPTM message procedures

On receiving QPTM messages from a peer IUA layer, the IUA layer on an SG or MGC needs to invoke the corresponding layer primitives (DL-ESTABLISH, DL-DATA, DL-UNIT DATA, DL-RELEASE) to the local Q.921 or Q.931 layer.

4.2 Procedures to support service in section 1.4.2

These procedures achieve the IUA layer's "Support for Communication between Layer Managements" service.

4.2.1 Layer Management primitives procedures

On receiving these primitives from the local Layer Management, the IUA layer will provide the appropriate response primitive across the internal local Layer Management interface.

An M-SCTP ESTABLISH request from Layer Management will initiate the establishment of an SCTP association. An M-SCTP ESTABLISH confirm will be sent to Layer Management when the initiated association set-up is complete. An M-SCTP ESTABLISH indication is sent to Layer Management upon successful completion of an incoming SCTP association set-up from a peer IUA node

An M-SCTP RELEASE request from Layer Management will initiate the tear-down of an SCTP association. An M-SCTP RELEASE confirm will be sent by Layer Management when the association teardown is complete. An M-SCTP RELEASE indication is sent to Layer Management upon successful tear-down of an SCTP association initiated by a peer IUA.

M-SCTP STATUS request and indication support a Layer Management query of the local status of a particular SCTP association.

M-NOTIFY indication and M-ERROR indication indicate to Layer Management the notification or error information contained in a received IUA Notify or Error message respectively. These indications can also be generated based on local IUA events.

M-ASP STATUS request/indication and M-AS-STATUS request/indication support a Layer Management query of the local status of a particular ASP or AS. No IUA peer protocol is invoked.

M-ASP-UP request, M-ASP-DOWN request, M-ASP-INACTIVE request and M-ASP-ACTIVE request allow Layer Management at an ASP to initiate state changes. These requests result in outgoing IUA ASP UP, ASP DOWN, ASP INACTIVE and ASP ACTIVE messages.

M-ASP-UP confirmation, M-ASP-DOWN confirmation, M-ASP-INACTIVE confirmation and M-ASP-ACTIVE confirmation indicate to Layer Management that the previous request has been confirmed.

Upon receipt of a M-TEI Status primitive from Layer Management, the IUA will send the corresponding MGMT message (TEI Status) to its peer. While doing so, the IUA layer needs to fill various fields of the common and specific headers correctly.

All MGMT messages are sent on a sequenced stream to ensure ordering. SCTP stream '0' SHOULD be used.

4.2.2 Receipt of IUA Peer Management messages

Upon receipt of IUA Management messages, the IUA layer MUST invoke the corresponding Layer Management primitive indications (e.g., M-AS Status ind., M-ASP Status ind., M-ERROR ind., M-TEI STATUS...) to the local layer management.

M-NOTIFY indication and M-ERROR indication indicate to Layer Management the notification or error information contained in a received IUA Notify or Error message. These indications can also be generated based on local IUA events.

All MGMT messages are sent on a sequenced stream to ensure ordering. SCTP stream '0' SHOULD be used.

4.3 Procedures to support service in section 1.4.3

These procedures achieve the IUA layer's "Support for management of active associations between SG and MGC" service.

4.3.1 AS and ASP State Maintenance

The IUA layer on the SG needs to maintain the states of each ASP as well as the state of the AS.

4.3.1.1 ASP States

The state of the each ASP, in each AS that it is configured, is maintained in the IUA layer on the SG. The state of an ASP changes due to the following type of events:

* Reception of messages from peer IUA layer at that ASP
* Reception of some messages from the peer IUA layer at other ASPs in the AS
* Reception of indications from SCTP layer

The ASP state transition diagram is shown in Figure 7. The possible states of an ASP are the following:

ASP-DOWN: Application Server Process is unavailable and/or the related SCTP association is down. Initially, all ASPs will be in this state. An ASP in this state SHOULD NOT be sent any IUA messages.

ASP-INACTIVE: The remote IUA peer at the ASP is available (and the related SCTP association is up) but application traffic is stopped. In this state the ASP can be sent any non-QPTM IUA messages (except for TEI Status messages).

ASP-ACTIVE: The remote IUA peer at the ASP is available and application traffic is active.

Figure 7 ASP State Transition Diagram

```
                                    +-------------+
             +----------------------|             |
             |   Alternate  +-------| ASP-ACTIVE  |
             |       ASP    |       +-------------+
             |    Takeover  |           ^     |
             |              |    ASP    |     | ASP
             |              |    Active |     | Inactive
             |              |           |     v
             |              |       +-------------+
             |              |       |             |
             |              +------>|  ASP-INACT  |
             |                      +-------------+
             |                          ^    |
   ASP Down/ |                     ASP  |    | ASP Down /
   SCTP CDI  |                     Up   |    | SCTP CDI
             |                          |    v
             |                      +-------------+
             +--------------------->|             |
                                    |  ASP-DOWN   |
                                    +-------------+
```

SCTP CDI: The local SCTP layer's Communication Down Indication to the Upper Layer Protocol (IUA) on an SG. The local SCTP will send this indication when it detects the loss of connectivity to the ASP's peer SCTP layer. SCTP CDI is understood as either a SHUTDOWN COMPLETE notification and COMMUNICATION LOST notification from the SCTP.

4.3.1.2 AS States

The state of the AS is maintained in the IUA layer on the SG.

The state of an AS changes due to events. These events include the following:

* ASP state transitions
* Recovery timer triggers

The possible states of an AS are the following:

AS-DOWN: The Application Server is unavailable. This state implies that all related ASPs are in the ASP-DOWN state for this AS. Initially the AS will be in this state.

AS-INACTIVE: The Application Server is available but no application traffic is active (i.e., one or more related ASPs are in the ASP-INACTIVE state, but none in the ASP-ACTIVE state). The recovery timer T(r) is not running or has expired.

AS-ACTIVE: The Application Server is available and application traffic is active. This state implies that at least one ASP is in the ASP-ACTIVE state.

AS-PENDING: An active ASP has transitioned from active to inactive or down and it was the last remaining active ASP in the AS. A recovery timer T(r) will be started and all incoming SCN messages will be queued by the SG. If an ASP becomes active before T(r) expires, the AS will move to AS-ACTIVE state and all the queued messages will be sent to the active ASP.

If T(r) expires before an ASP becomes active, the SG stops queuing messages and discards all previously queued messages. The AS will move to AS-INACTIVE if at least one ASP is in ASP-INACTIVE state, otherwise it will move to AS-DOWN state.

                 Figure 8 AS State Transition Diagram

```
        +----------+  one ASP trans ACTIVE   +-------------+
        |          |------------------------>|             |
        | AS-INACT |                         |  AS-ACTIVE  |
        |          |                         |             |
        |          |<                        |             |
        +----------+ \                       +-------------+
           ^   |      \ Tr Trigger                ^    |
           |   |       \ at least one             |    |
           |   |        \ ASP in UP               |    |
           |   |         \                        |    |
           |   |          \                       |    |
           |   |           \                      |    |
   one ASP |   |            \            one ASP  |    | Last ACTIVE ASP
   trans   |   | all ASP     \------\    trans to |    | trans to INACT
   to      |   | trans to            \   ACTIVE   |    | or DOWN
   INACT   |   | DOWN                 \           |    | (start Tr timer)
           |   |                       \          |    |
           |   |                        \         |    |
           |   |                         \        |    |
           |   v                          \       |    v
        +----------+                       \ +-------------+
        |          |                        -|             |
        | AS-DOWN  |                         | AS-PENDING  |
        |          |                         |  (queueing) |
        |          |<------------------------|             |
        +----------+    Tr Expiry and no     +-------------+
                       ASP in INACTIVE state
```

      Tr = Recovery Timer

4.3.2 ASPM procedures for primitives

Before the establishment of an SCTP association the ASP state at both the SG and ASP is assumed to be "Down".

As the ASP is responsible for initiating the setup of an SCTP association to an SG, the IUA layer at an ASP receives an M-SCTP ESTABLISH request primitive from the Layer Management, the IUA layer will try to establish an SCTP association with the remote IUA peer at an SG. Upon reception of an eventual SCTP-Communication Up confirm primitive from the SCTP, the IUA layer will invoke the primitive M-SCTP ESTABLISH confirm to the Layer Management.

At the SG, the IUA layer will receive an SCTP Communication Up indication primitive from the SCTP. The IUA layer will then invoke the primitive M-SCTP ESTABLISH indication to the Layer Management.

Once the SCTP association is established and assuming that the local IUA-User is ready, the local ASP IUA Application Server Process Maintenance (ASPM) function will initiate the ASPM procedures, using the ASP Up/-Down/-Active/-Inactive messages to convey the ASP state to the SG - see Section 4.3.3.

The Layer Management and the IUA layer on SG can communicate the status of the application server using the M-AS STATUS primitives. The Layer Management and the IUA layer on both the SG and ASP can communicate the status of an SCTP association using the M-SCTP STATUS primitives.

If the Layer Management on SG or ASP wants to bring down an SCTP association for management reasons, they would send M-SCTP RELEASE request primitive to the local IUA layer. The IUA layer would release the SCTP association and upon receiving the SCTP Communication Down indication from the underlying SCTP layer, it would inform the local Layer Management using M-SCTP RELEASE confirm primitive.

If the IUA layer receives an SCTP-Communication Down indication from the underlying SCTP layer, it will inform the Layer Management by invoking the M-SCTP RELEASE indication primitive. The state of the ASP will be moved to "Down" at both the SG and ASP.

At an ASP, the Layer Management MAY try to reestablish the SCTP association using M-SCTP ESTABLISH request primitive.

4.3.3 ASPM procedures for peer-to-peer messages

All ASPM messages are sent on a sequenced stream to ensure ordering. SCTP stream ’0’ SHOULD be used.

4.3.3.1 ASP Up

After an ASP has successfully established an SCTP association to an SG, the SG waits for the ASP to send an ASP Up message, indicating that the ASP IUA peer is available. The ASP is always the initiator of the ASP Up exchange.

When an ASP Up message is received at an SG and internally the remote ASP is not considered locked-out for local management reasons, the SG marks the remote ASP as "Inactive". The SG responds with an ASP Up Ack message in acknowledgement. The SG sends an ASP-Up Ack message in response to a received ASP Up message even if the ASP is already marked as "Inactive" at the SG.

If for any local reason the SG cannot respond with an ASP Up, the SG responds to a ASP Up with a with an ASP-Down Ack message with Reason "Management Blocking".

At the ASP, the ASP Up Ack message received from the SG is not acknowledged by the ASP. If the ASP does not receive a response from the SG, or an ASP Down Ack is received, the ASP MAY resend ASP Up messages every 2 seconds until it receives a ASP Up Ack message from the SG. The ASP MAY decide to reduce the frequency (say to every 5 seconds) if an ASP Up Ack is not received after a few tries.

The ASP MUST wait for the ASP Up Ack message from the SG before sending any ASP traffic control messages (ASPAC or ASPIA) or Data messages or it will risk message loss. If the SG receives QPTM, ASP Active or ASP Inactive messages before an ASP Up is received, the SG SHOULD discard these messages.

4.3.3.2 ASP Down

The ASP will send an ASP Down to an SG when the ASP is to be removed from the list of ASPs in all Application Servers that it is a member and no longer receive any IUA traffic or management messages.

Whether the ASP is permanently removed from an AS is a function of configuration management.

The SG marks the ASP as "Down" and returns an ASP Down Ack message to the ASP if one of the following events occur:

- to acknowledge an ASP Down message from an ASP,
- to reply to ASPM messages from an ASP which is locked out for management reasons.

The SG sends an ASP Down Ack message in response to a received ASP Down message from the ASP even if the ASP is already marked as "Down" at the SG.

If the ASP does not receive a response from the SG, the ASP MAY send ASP Down messages every 2 seconds until it receives an ASP Down Ack message from the SG or the SCTP association goes down. The ASP MAY decide to reduce the frequency (say to every 5 seconds) if an ASP Down Ack is not received after a few tries.

4.3.3.3 IUA Version Control

If a ASP Up message with an unsupported version is received, the receiving end responds with an Error message, indicating the version the receiving node supports and notifies Layer Management.

This is useful when protocol version upgrades are being performed in a network. A node upgraded to a newer version SHOULD support the older versions used on other nodes it is communicating with. Because ASPs initiate the ASP Up procedure it is assumed that the Error message would normally come from the SG.

4.3.3.4 ASP Active

Any time after the ASP has received a ASP Up Ack from the SG, the ASP sends an ASP-Active (ASPAC) to the SG indicating that the ASP is ready to start processing traffic. In the case where an ASP is configured/registered to process the traffic for more than one Application Server across an SCTP association, the ASPAC contains one or more Interface Identifiers to indicate for which Application Servers the ASPAC applies.

When an ASP Active (ASPAC) message is received, the SG responds to the ASP with a ASPAC Ack message acknowledging that the ASPAC was received and starts sending traffic for the associated Application Server(s) to that ASP.

The ASP MUST wait for the ASP-Active Ack message from the SG before sending any Data messages or it will risk message loss. If the SG receives QPTM messages before an ASP Active is received, the SG SHOULD discard these messages.

There are two modes of Application Server traffic handling in the SG IUA - Over-ride and Load-sharing. The Type parameter in the ASPAC message indicates the mode used in a particular Application Server. If the SG determines that the mode indicates in an ASPAC is incompatible with the traffic handling mode currently used in the AS, the SG responds with an Error message indicating Unsupported Traffic Handling Mode.

In the case of an Over-ride mode AS, reception of an ASPAC message at an SG causes the redirection of all traffic for the AS to the ASP that sent the ASPAC. The SG responds to the ASPAC with an ASP-Active Ack message to the ASP. Any previously active ASP in the AS is now considered Inactive and will no longer receive traffic from the SG within the AS. The SG sends a Notify (Alternate ASP-Active) to the previously active ASP in the AS, after stopping all traffic to that ASP.

In the case of a load-share mode AS, reception of an ASPAC message at an SG causes the direction of traffic to the ASP sending the ASPAC, in addition to all the other ASPs that are currently active in the AS. The algorithm at the SG for load-sharing traffic within an AS to all the active ASPs is implementation dependent. The algorithm could, for example be round-robin or based on information in the Data message, such as Interface Identifier, depending on the requirements of the application and the call state handling assumptions of the collection of ASPs in the AS. The SG responds to the ASPAC with a ASP-Active Ack message to the ASP.

4.3.3.5 ASP Inactive

When an ASP wishes to withdraw from receiving traffic within an AS, the ASP sends an ASP Inactive (ASPIA) to the SG. In the case where an ASP is configured/registered to process the traffic for more than one Application Server across an SCTP association, the ASPIA contains one or more Interface Identifiers to indicate for which Application Servers the ASPIA applies.

There are two modes of Application Server traffic handling in the SG IUA when withdrawing an ASP from service - Over-ride and Load-sharing. The Type parameter in the ASPIA message indicates the mode used in a particular Application Server. If the SG determines that the mode indicates in an ASPAC is incompatible with the traffic handling mode currently used in the AS, the SG responds with an Error message indicating Unsupported Traffic Handling Mode.

In the case of an Over-ride mode AS, where normally another ASP has already taken over the traffic within the AS with an Over-ride ASPAC, the ASP which sends the ASPIA is already considered by the SG to be "Inactive". An ASPIA Ack message is sent to the ASP, after ensuring that all traffic is stopped to the ASP.

In the case of a Load-share mode AS, the SG moves the ASP to the "Inactive" state and the AS traffic is re-allocated across the remaining "active" ASPs per the load-sharing algorithm currently used within the AS. An ASPIA Ack message is sent to the ASP after all traffic is halted to the ASP. A NTFY (Insufficient ASPs) MAY be sent to all inactive ASPs, if required.

If no other ASPs are Active in the Application Server, the SG sends a NTFY (AS-Pending) to all inactive ASPs of the AS and either discards all incoming messages for the AS or starts buffering the incoming messages for T(r)seconds, after which messages will be discarded. T(r) is configurable by the network operator. If the SG receives an ASPAC from an ASP in the AS before expiry of T(r), the buffered traffic is directed to the ASP and the timer is cancelled. If T(r) expires, the AS is moved to the "Inactive" state.

4.3.3.6 Notify

A Notify message reflecting a change in the AS state is sent to all ASPs in the AS, except those in the "Down" state, with appropriate Status Identification.

In the case where a Notify (AS-Pending) message is sent by an SG that now has no ASPs active to service the traffic, or a NTFY (Insufficient ASPs) is sent in the Load-share mode, the Notify does not explicitly force the ASP(s) receiving the message to become active. The ASPs remain in control of what (and when) action is taken.

4.3.3.7 Heartbeat

The optional Heartbeat procedures MAY be used when operating over transport layers that do not have their own heartbeat mechanism for detecting loss of the transport association (i.e., other than the SCTP).

After receiving an ASP Up Ack message from the SG in response to an ASP Up message, the ASP MAY optionally send Beat messages periodically, subject to a provisionable timer T(beat). The SG IUA, upon receiving a BEAT message from the ASP, responds with a BEAT ACK message. If no BEAT message (or any other IUA message) is received from the SG within the timer 2*T(beat), the SG will consider the remote IUA as "Down". The SG will also send an ASP Down Ack message to the ASP.

At the ASP, if no BEAT ACK message (or any other IUA message) is received from the SG within 2*T(beat), the SG is considered unavailable. Transmission of BEAT messages is stopped and ASP Up procedures are used to re-establish communication with the SG IUA peer.

The BEAT message MAY optionally contain an opaque Heartbeat Data parameter that MUST be echoed back unchanged in the related Beat Ack message. The ASP upon examining the contents of the returned BEAT Ack message MAY choose to consider the remote ASP as unavailable. The contents/format of the Heartbeat Data parameter is implementation-dependent and only of local interest to the original sender. The contents MAY be used, for example, to support a Heartbeat sequence algorithm (to detect missing Heartbeats), and/or a timestamp mechanism (to evaluate delays).

Note: Heartbeat related events are not shown in Figure 4 "ASP state transition diagram".

5.0 Examples

5.1 Establishment of Association and Traffic between SGs and ASPs

5.1.1 Single ASP in an Application Server (1+0 sparing)

This scenario shows the example IUA message flows for the establishment of traffic between an SG and an ASP, where only one ASP is configured within an AS (no backup). It is assumed that the SCTP association is already set-up.

```
                SG                        ASP1
                 |
                 |<----------ASP Up----------|
                 |--------ASP Up Ack------->|
                 |                           |
                 |<-------ASP Active--------|
                 |------ASP Active Ack----->|
                 |                           |
```

5.1.2 Two ASPs in Application Server (1+1 sparing)

This scenario shows the example IUA message flows for the establishment of traffic between an SG and two ASPs in the same Application Server, where ASP1 is configured to be Active and ASP2 a standby in the event of communication failure or the withdrawal from service of ASP1. ASP2 MAY act as a hot, warm, or cold standby depending on the extent to which ASP1 and ASP2 share call state or can communicate call state under failure/withdrawal events. The example message flow is the same whether the ASP-Active messages are Over-ride or Load-share mode although typically this example would use an Over-ride mode.

```
          SG                        ASP1                         ASP2
           |                          |                            |
           |<---------ASP Up----------|                            |
           |-------ASP Up Ack------->|                            |
           |                          |                            |
           |<----------------------------ASP Up----------------|
           |----------------------------ASP Up Ack------------->|
           |                          |                            |
           |                          |                            |
           |<-------ASP Active-------|                            |
           |-----ASP Active Ack----->|                            |
           |                          |                            |
```

5.1.3 Two ASPs in an Application Server (1+1 sparing, load-sharing case)

This scenario shows a similar case to Section 5.1.2 but where the two ASPs are brought to active and load-share the traffic load. In this case, one ASP is sufficient to handle the total traffic load.

```
      SG                        ASP1                       ASP2
       |                          |                          |
       |<---------ASP Up---------|                          |
       |--------ASP Up Ack------>|                          |
       |                          |                          |
       |<------------------------------ASP Up----------------|
       |-----------------------------ASP Up Ack------------->|
       |                          |                          |
       |                          |                          |
       |<--ASP Active (Ldshr)----|                          |
       |----ASP Active Ack------>|                          |
       |                          |                          |
       |<----------------------------ASP Active (Ldshr)-----|
       |-----------------------------ASP Active Ack--------->|
       |                          |                          |
```

5.1.4 Three ASPs in an Application Server (n+k sparing, load-sharing case)

This scenario shows the example IUA message flows for the establishment of traffic between an SG and three ASPs in the same Application Server, where two of the ASPs are brought to active and share the load. In this case, a minimum of two ASPs are required to handle the total traffic load (2+1 sparing).

```
      SG                  ASP1                ASP2                ASP3
       |                    |                   |                   |
       |<------ASP Up-------|                   |                   |
       |-----ASP Up Ack---->|                   |                   |
       |                    |                   |                   |
       |<--------------------------ASP Up-------|                   |
       |------------------------ASPUp Ack)----->|                   |
       |                    |                   |                   |
       |<---------------------------------------------ASP Up--------|
       |-------------------------------------------ASP Up Ack------>|
       |                    |                   |                   |
       |                    |                   |                   |
       |<-ASP Act (Ldshr)---|                   |                   |
       |----ASP Act Ack---->|                   |                   |
       |                    |                   |                   |
       |<---------------------ASP Act (Ldshr)---|                   |
       |----------------------ASP Act Ack------>|                   |
       |                    |                   |                   |
```

5.2 ASP Traffic Fail-over Examples

5.2.1 (1+1 Sparing, withdrawal of ASP, Back-up Over-ride)

The following example shows a case in which an ASP withdraws from service:

```
          SG                        ASP1                        ASP2
           |                          |                           |
           |<-----ASP Inactive-------|                           |
           |----ASP Inactive Ack---->|                           |
           |-------------------NTFY(AS-Pending) ----------------->|
           |                          |                           |
           |<----------------------------- ASP Active-----------|
           |----------------------------ASP Active Ack)------->|
           |                                                      |
```

In this case, the SG notifies ASP2 that the AS has moved to the Down state. The SG could have also (optionally) sent a Notify message when the AS moved to the Pending state.

Note: If the SG detects loss of the IUA peer (IUA heartbeat loss or detection of SCTP failure), the initial SG-ASP1 ASP Inactive message exchange would not occur.

5.2.2 (1+1 Sparing, Back-up Over-ride)

The following example shows a case in which ASP2 wishes to over-ride ASP1 and take over the traffic:

```
        SG                       ASP1                       ASP2
         |                         |                           |
         |<-------------------------------ASP Active----------|
         |----------------------------ASP Active Ack--------->|
         |----NTFY( Alt ASP-Act)-->|
         |                         |                           |
```

In this case, the SG notifies ASP1 that an alternative ASP has overridden it.

5.2.3 (n+k Sparing, Load-sharing case, withdrawal of ASP)

Following on from the example in Section 5.1.4, and ASP1 withdraws from service

```
   SG                   ASP1                ASP2                ASP3
    |                    |                   |                   |
    |<----ASP Inact------|                   |                   |
    |---ASP Inact Ack--->|                   |                   |
    |                    |                   |                   |
    |---------------------------------NTFY(Ins. ASPs)----------->|
    |                    |                   |                   |
    |<------------------------------------------ASP Act (Ldshr)---|
    |-------------------------------------------ASP Act (Ack)--->|
    |                    |                   |                   |
```

In this case, the SG has knowledge of the minimum ASP resources required (implementation dependent) for example if the SG knows that n+k = 2+1 for a load-share AS and n currently equals 1.

Note: If the SG detects loss of the ASP1 IUA peer (IUA heartbeat loss or detection of SCTP failure), the first SG-ASP1 ASP Inactive message exchange would not occur.

5.3 Q.921/Q.931 primitives backhaul Examples

When the IUA layer on the ASP has a QPTM message to send to the SG, it will do the following:

- Determine the correct SG

- Find the SCTP association to the chosen SG

- Determine the correct stream in the SCTP association based on the D channel

- Fill in the QPTM message, fill in IUA Message Header, fill in Common Header

- Send the QPTM message to the remote IUA peer in the SG, over the SCTP association

When the IUA layer on the SG has a QPTM message to send to the ASP, it will do the following:

- Determine the AS for the Interface Identifier

- Determine the Active ASP (SCTP association) within the AS

- Determine the correct stream in the SCTP association based on the D channel

- Fill in the QPTM message, fill in IUA Message Header, fill in Common Header

- Send the QPTM message to the remote IUA peer in the ASP, over the SCTP association

An example of the message flows for establishing a data link on a signaling channel, passing PDUs and releasing a data link on a signaling channel is shown below. An active association between MGC and SG is established (Section 5.1) prior to the following message flows.

```
         SG                              ASP

                     <----------- Establish Request
 Establish Confirm  ---------->

                     <----------- Data Request
    Data Indication ----------->
                     <----------- Data Request
    Data Indication ----------->
                     <----------- Data Request
                     <----------- Data Request
    Data Indication ----------->

                     <----------- Release Request (RELEASE_MGMT)
   Release Confirm  ---------->
```

An example of the message flows for a failed attempt to establish a data link on the signaling channel is shown below. In this case, the gateway has a problem with its physical connection (e.g., Red Alarm), so it cannot establish a data link on the signaling channel.

```
          SG                              ASP

                       <----------- Establish Request (ESTABLISH_START)
      Release Indication ---------->
      (RELEASE_PHYS)
```

5.4 Layer Management Communication Examples

An example of the message flows for communication between Layer Management modules between SG and ASP is shown below. An active association between ASP and SG is established (Section 5.1) prior to the following message flows.

```
                SG                        ASP

                       <----------- Data Request
        Error Indication ---------->
         (INVALID_TEI)

                       <----------- TEI Status Request
      TEI Status Confirm ---------->
           (Unassigned)
```

6.0 Security

IUA is designed to carry signaling messages for telephony services. As such, IUA MUST involve the security needs of several parties the end users of the services; the network providers and the applications involved. Additional requirements MAY come from local regulation. While having some overlapping security needs, any security solution SHOULD fulfill all of the different parties' needs.

6.1 Threats

There is no quick fix, one-size-fits-all solution for security. As a transport protocol, IUA has the following security objectives:

* Availability of reliable and timely user data transport.
* Integrity of user data transport.
* Confidentiality of user data.

IUA runs on top of SCTP. SCTP [3] provides certain transport related security features, such as

* Blind Denial of Service Attacks
* Flooding
* Masquerade
* Improper Monopolization of Services

When IUA is running in professionally managed corporate or service provider network, it is reasonable to expect that this network includes an appropriate security policy framework. The "Site Security Handbook" [5] SHOULD be consulted for guidance.

When the network in which IUA runs in involves more than one party, it MAY NOT be reasonable to expect that all parties have implemented security in a sufficient manner. In such a case, it is recommended that IPSEC is used to ensure confidentiality of user payload. Consult [6] for more information on configuring IPSEC services.

6.2 Protecting Confidentiality

Particularly for mobile users, the requirement for confidentiality MAY include the masking of IP addresses and ports. In this case application level encryption is not sufficient; IPSEC ESP SHOULD be used instead. Regardless of which level performs the encryption, the IPSEC ISAKMP service SHOULD be used for key management.

7.0 IANA Considerations

7.1 SCTP Payload Protocol Identifier

A request will be made to IANA to assign an IUA value for the Payload Protocol Identifier in SCTP Payload Data chunk. The following SCTP Payload Protocol Identifier will be registered:

```
IUA    "1"
```

The SCTP Payload Protocol Identifier is included in each SCTP Data chunk, to indicate which protocol the SCTP is carrying. This Payload Protocol Identifier is not directly used by SCTP but MAY be used by certain network entities to identify the type of information being carried in a Data chunk.

The User Adaptation peer MAY use the Payload Protocol Identifier as a way of determining additional information about the data being presented to it by SCTP.

7.2 IUA Protocol Extensions

This protocol may also be extended through IANA in three ways:

-- through definition of additional message classes,
-- through definition of additional message types, and
-- through definition of additional message parameters.

The definition and use of new message classes, types and parameters is an integral part of SIGTRAN adaptation layers. Thus, these extensions are assigned by IANA through an IETF Consensus action as defined in [RFC2434].

The proposed extension must in no way adversely affect the general working of the protocol.

7.2.1 IETF Defined Message Classes

The documentation for a new message class MUST include the following information:

(a) A long and short name for the message class.
(b) A detailed description of the purpose of the message class.

7.2.2 IETF Defined Message Types

Documentation of the message type MUST contain the following information:

(a) A long and short name for the new message type.
(b) A detailed description of the structure of the message.
(c) A detailed definition and description of intended use of each field within the message.
ti3 (d) A detailed procedural description of the use of the new message type within the operation of the protocol.
(e) A detailed description of error conditions when receiving this message type.

When an implementation receives a message type which it does not support, it MUST respond with an Error (ERR) message with an Error Code of Unsupported Message Type.

7.2.3 IETF-defined TLV Parameter Extension

Documentation of the message parameter MUST contain the following information:

(a) Name of the parameter type.
(b) Detailed description of the structure of the parameter field. This structure MUST conform to the general type-length-value format described in Section 3.1.5.
(c) Detailed definition of each component of the parameter value.
(d) Detailed description of the intended use of this parameter type, and an indication of whether and under what circumstances multiple instances of this parameter type may be found within the same message type.

8.0 Acknowledgements

The authors would like to thank Alex Audu, Maria Sonia Vazquez Arevalillo, Ming-te Chao, Keith Drage, Norm Glaude, Nikhil Jain, Bernard Kuc, Ming Lin, Stephen Lorusso, John Loughney, Barry Nagelberg, Neil Olson, Lyndon Ong, Heinz Prantner, Jose Luis Jimenez Ramirez, Ian Rytina, Michael Tuexen and Hank Wang for their valuable comments and suggestions.

9.0 References

[1] ITU-T Recommendation Q.920, 'Digital Subscriber signaling System No. 1 (DSS1) - ISDN User-Network Interface Data Link Layer - General Aspects'

[2] T1S1.7/99-220 Contribution, 'Back-hauling of DSS1 protocol in a Voice over Packet Network'

[3] Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M., Zhang, L. and V. Paxson, "Stream Control Transmission Protocol", RFC 2960, October 2000.

[4] Ong, L., Rytina, I., Garcia, M., Schwarzbauer, H., Coene, L., Lin, H., Juhasz, I., Holdrege, M., and C. Sharp, "Architectural Framework for Signaling Transport", RFC 2719, October 1999.

[5] Fraser, B., "Site Security Handbook", FYI 8, RFC 2196, September 1997.

[6] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[7] Bradner, s., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[8] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

10.0 Authors' Addresses

Ken Morneault
Cisco Systems Inc.
13615 Dulles Technology Drive
Herndon, VA. 20171
USA

Phone: +1-703-484-3323
EMail: kmorneau@cisco.com

Malleswar Kalla
Telcordia Technologies
PYA 2J-341
3 Corporate Place
Piscataway, NJ 08854
USA

Phone: +1-732-699-3728
EMail: mkalla@telcordia.com

Selvam Rengasami
Telcordia Technologies
NVC-2Z439
331 Newman Springs Road
Red Bank, NJ 07701
USA

Phone: +1-732-758-5260
EMail: srengasa@telcordia.com

Greg Sidebottom
Nortel Networks
3685 Richmond Road
Nepean, Ontario
Canada K2H5B7

Phone: +1-613-763-7305
EMail: gregside@nortelnetworks.com

10. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.