Network Working Group Brian Bidulock
INTERNET-DRAFT OpenSS7 Corporation

Expires in six months January 2, 2003

Correlation Id and Hearbeat Procedures (CORID)
Supporting Lossless Fail-Over between SCTP Associations
for
Signalling User Adaptation Layers
<draft-bidulock-sigtran-corid-01.ps>

Status of this Memo

Abstract

This Internet-Draft describes Correlation Id and Heartbeat procedures to support lossless fail-over between SCTP [RFC 2960] associations for SS7 [Q.700] Signalling User Adaptation Protocols [M2UA...TUA] supporting the concept of a *Routing Context* or *Interface Identifier*. These procedures permit lossless fail-over between Application Server Processes (ASPs) at a Signalling Gateway (SG) and fail-over between Signalling Gateway Processes (SGPs) and Signalling Gateways (SGs) at an Application Server Process (ASP). Lossless fail-over permits these fail-overs to occur without loss or duplication of UA-User messages.

1. Introduction

1.1. Scope

This Internet-Draft describes Correlation Id and Heartbeat (CORID) procedures to support lossless fail-over between SCTP [RFC 2960] associations for SS7 [Q.700] Signalling User Adaptation Protocols [M2UA...TUA] above MTP3 [Q.704] supporting the concept of a *Routing Context* or *Interface Identifier*. These procedures permit lossless fail-over between Application Server Processes (ASPs) at a Signalling Gateway (SG) and fail-over between Signalling Gateway Processes (SGPs) and Signalling Gateways (SGs) at an Application Server Process (ASP). Lossless fail-over permits these fail-overs to occur without loss or duplication of UA-User messages.

UA implementations with *CORID* are intended to be compatible with UA implementations not supporting this configuration; however, the full benefits acheived by the *CORID* procedures will not be realized unless implementations at both endpoints implement *CORID*.

1.2. Change History

Changes from Version 0.0 to Version 1.0:

- added change history,
- updated version numbers and dates,
- updated acknowledgements,
- corrected section reference typos,
- added postscript diagrams,
- changed most SSNM messages to divertable,
- updated interworking to perform timed diversion on recovery,
- update *Traffic Flow Id* to be a simple unique identifier, no longer containing a stream identifier,
- updated author's address.

1.3. Terminology

CORID supplements the terminology used in the UA documents [M2UA...TUA] by adding the following terms:

Changeback – the MTP3 [Q.704] procedure for redirecting signalling traffic back to a primary linkset from an alternate linkset.

Changeover – the MTP3 [Q.704] procedure for diverting signalling traffic from a failed primary linkset to an alternate linkset.

Lossless Fail-Over – is mechanism for fail-over between SCTP [RFC 2960] associations (i.e, between ASPs, IPSPs, SGPs or SGs) that provides for the elminitation of duplication or loss of UA-User messages between SG and AS.

Message Duplication – a situation where multiple copies of a UA-User message arrives at a Signalling Endpoint.

Message Loss – a situation where instances of a UA-User message is lost in transit between Signalling Endpoints.

Message Mis-sequencing – a situation where UA-User messages that are intended to arrive in sequence, arrive at a terminating Signalling Endpoint in an order other than the order in which the messages were transmitted at the originating Signalling Endpoint.

Signalling Endpoint (SEP) – in this document, a *Signalling Enpoint* is an SS7 SEP [Q.700] or an Application Server.

Signalling Peer Process (SPP) – refers to an ASP, SGP or IPSP.

Signalling User Adaptation Layer (UA) – one or more of the Stream Control Transmission Protocol (SCTP) [RFC 2960] SS7 Signalling User Adaptation Layers [M2UA...TUA] supporting the *Correlation Id* parameter and the *BEAT* message.

Time-controlled Changeover – the MTP3 [Q.704] procedure for diverting signalling traffic from a failed primary linkset to an alternate linkset where sequence number information cannot be exchanged between signalling points or where it is undesirable to use the normal changeover procedures.

1.4. Overview

The existing UA [M2UA...TUA] procedures do not include procedures to avoid loss or duplication of messages when a UA peer must fail-over between SCTP [RFC 2960] associations between diverse Application Server Processes (ASPs), Signalling Gateway Processes (SGPs), Signalling Gateways (SGs), and IP Signalling Processes (IPSPs).

CORID provides procedures to eliminate message loss, duplication or mis-sequencing under all failure, deactivation, recovery and activation scenarios. *CORID* provides the following capabilities that are not provided for in the existing UA specifications:

- Support for elimintating mis-sequencing of UA-User messages at signalling endpoints (Application Servers or SS7 SEPs) when diverting messages between ASPs, SGPs, SGs, or IPSPs by supporting *BEAT* procedures analogous to the MTP3 [Q.704] Changeback procedure.

- Support for eliminating duplication of UA-User messages at signalling endpoints (Application Servers or SS7 SEPs) or SS7 endpoints across fail-over between ASPs, SGPs, SGs, or IPSPs.

- Support for elimination of message loss of UA-User messages between Signalling Gateways (SGs) and Application Servers (ASs) across fail-over between ASPs, SGPs, SGs, or IPSPs.

1.4.1. Configuration

For carrier-class operation, the SS7 Signalling User Adaptation Layers recommend that Signalling Gateways and Application Servers be configured such that there is no single point of failure within the SG/AS architecture or in the intervening network. The SS7 UAs also recommend that no Application Server be configured for less than two (2) Application Server Processes.

All of the UAs describe an override, loadsharing and broadcast traffic mode. The UA protocols place no restrictions on the distribution algorithm which is used for distributing traffic over multiple Signalling Processes. Additional traffic distribution proposals have been put forward for Load Selection [LOADSEL] and Load Grouping [LOADGRP]

Fail-over between Application Server Processes (ASPs) and Signalling Gateway Processes (SGPs) is not detailed in the UA protocols [M2UA...TUA], but it is clear that when an SCTP association fails and the ASP transitions to the ASP-DOWN state from the perspective of the SGP peer, the traffic which the associated ASP was previously responsible needs to be diverted to an alternate ASP (if available) in the same Application Server pool.

1.4.2. Conditions at Fail-Over

The details of this diversion of traffic is not specified, however, a dichotomy exists when such fail-over occurs as a result of the loss of an SCTP association between these Signalling Peer Processes (SPPs). When an SPP loses its SCTP association with another SPP, and diverts traffic towards another SPP, there exists the possibility that messages previously destined to the peer SPP exist in several categories, as follows:

Category (1) – Queued in the sending SPP process,

Category (2) – queued for transmission, but not yet transmitted by the transport provider (SCTP),

Category (3) – queued for retransmission, but not yet acknowledged by the peer transport provider (SCTP), and,

Category (4) – acknowledged by the peer transport provider (SCTP) and deleted from the sending transport provider's (SCTP's) retransmission queue.

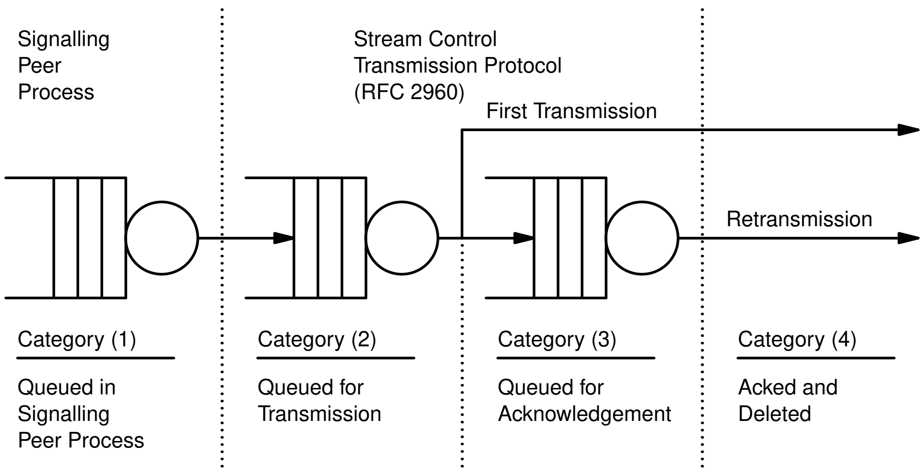

Figure 1. Buffer Categories at SCTP Association Failure

These categories are illustrated in *Figure 1*. Note that to retransmit categories (2) and (3) (and perhaps categories (1)) on another link requires sent data acknowledgment or buffer retrieval capability by the underlying transport provider.

As there is no SPP peer-to-peer acknowledgement, for messages in Categories (3) and (4), the message might or might not have been delivered to the peer SPP. Therefore, at the time of failure of an SCTP association between two Signalling Peer Processes (SPPs), it is not possible for either SPP to determine which of the messages in categories (3) and (4) above (transmitted, but not yet acknowledged; transmitted and acknowledged) were successfully received by the peer before failure. Without information concerning which messages in this category were successfully received by the peer, the SPP either risks message loss or message duplication when it diverts traffic from the failed association.

1.4.3. Sources of Message Loss and Duplication

If the messages from category (3) or (4) are retransmitted on an alternate association, the SPP diverting the traffic risks message duplication. This is because some messages of the category might possibly have been successfully received by the peer before fail-over.

If the messages from category (3) and (4) are discarded before diverting messages from categories (1) and (2) and then new traffic on an alternate association, the SPP risks message loss. This is because some of the messages in category (3) and (4) might possibly have *not* been received by the peer SPP before the association failed.

This is the dychotomy: regardless of the nature of a policy concerning the disposition of messages at an SPP experiencing failure to its peer, without information concerning messages successfully received by the peer, the SPP risks message loss or duplication.

It should be possible to induce such a system to demonstrate message loss or duplication.

Because SS7 performance requirements [Q.706] have more stringent requirements against duplication of messages than loss of messages, the only policy is to discard messages in category (3).

To avoid loss of messages to meet SS7 performance requirements [Q.706] in consideration of this dichotomy, implementation cost may be driven higher than would be the case if a procedure were established to exchange information between the Signalling Processes on either side of a failed association.

This Internet-Draft provides Correlation Id and Heartbeat procedures for fail-over for the SS7 signalling UAs which will remove the possibility of message loss or duplication in the event that an SCTP association failure while communication between the Application Server and Signalling Gateway is still possible.

1.4.4. Conditions at Recovery

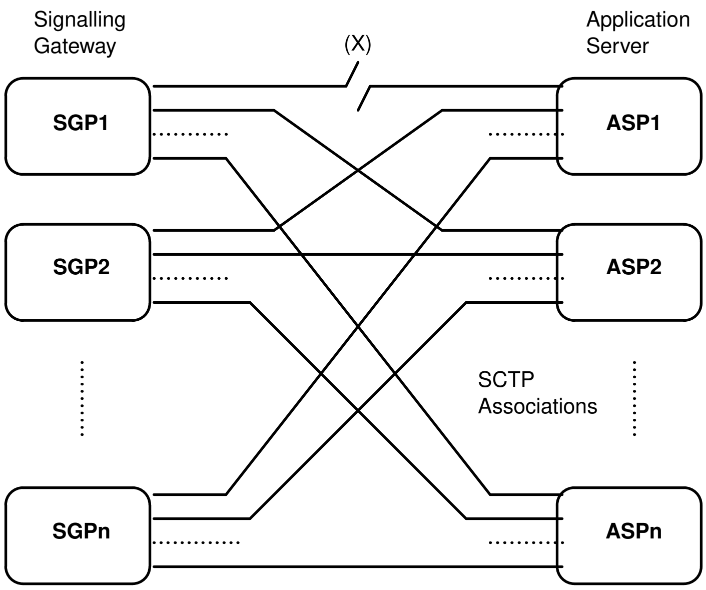

Figure 2. Example (A) Configuration of ASPs and SGPs

Figure 2 illustrates an example (A) configuration of ASPs and SGPs. In this example, the ASP and SGP are interconnected with a full-mesh arrangement of SCTP Associations. Each ASP is interconnected to each SGP by an association.

When a failure of the SCTP assocation occurs, it is, for example, between 'SGP1' and 'ASP1' as indicated by the (X) in the *Figure 2*. When a recovery occurs, it is also the SCTP association between 'SGP1' and 'ASP1' that recovers.

The normal procedure for dealing with such a failure[1] is for SGP1 to mark ASP1 in the ASP-DOWN state and to redirect traffic over the remaining ASPs in the Application Server[2].

When the SCTP association between ASP1 and SGP1 recovers and ASP1 succesfully activates for the AS using the ASP Active Procedures[3], once ASP1 has entered the ASP-ACTIVE state for the AS, message mis-sequencing can occur if traffic is immediately applied on the newly active association.

The UA procedures[3] provide no detail concerning the restarting of traffic to recovering ASPs in the AS.

1.4.5. Sources of Message Mis-Sequencing

Because the SGPs can be experiencing different loads or other local factors, each SGP may differ. Therefore, restoring a traffic flow to a newly active SGP, without first ensuring that messages are purged through the old path before the diversion, can result in message mis-sequencing. This is example (C) illustrated in *Figure 3*.

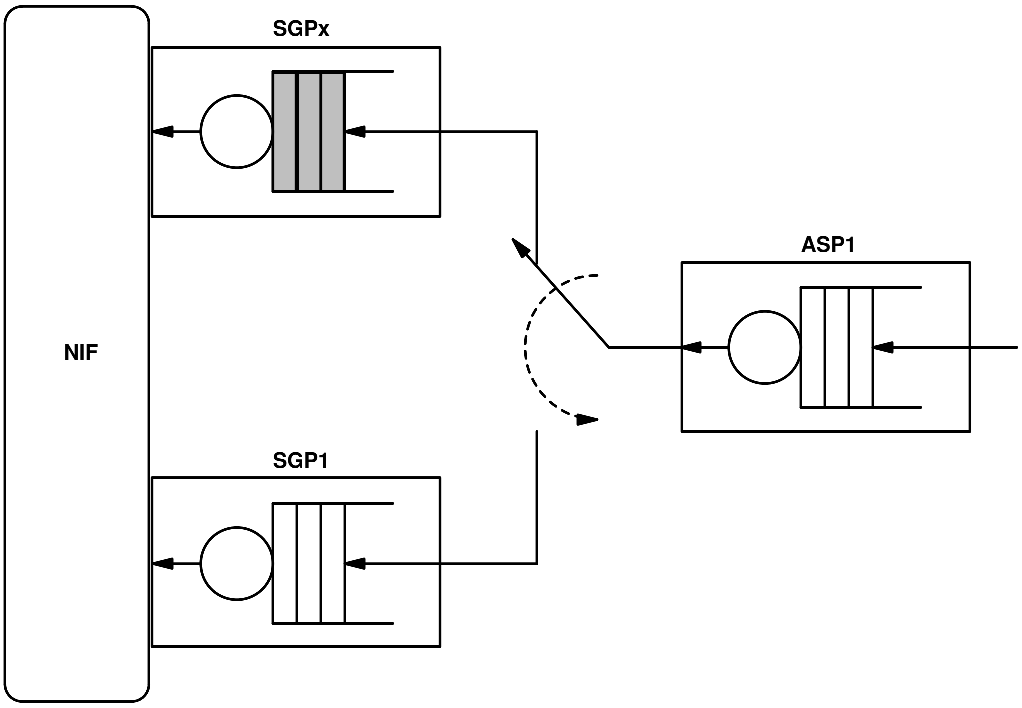

Figure 3. Example (C) Restoration of a Traffic Flow

Before switching traffic back to SGP1 from SGPx, SGPx is queueing traffic from ASP1 to the SS7 network. (This queueing could either be within the SGP or as a result of queueing within the transport protocol.) [RFC 2960] If the traffic flow from ASP1 is switched rapidly to SGP1, a race condition exists between messages in SGPx's queue and messages in SGP1's queue. A rapid switch can result in mis-sequencing.

As SGPx and SGP1 do not necessarily have to belong to the same SG (and because there exists queuing within the transport protocol itself), close queue synchornization between SGPx and SGP1 cannot be expected.

CORID provides both a time-controlled and a Heartbeat procedure for restoration of traffic to avoid mis-sequencing during restoration.

1.5. Functional Areas

The *CORID* procedures to avoid message loss, duplication and mis-sequencing under these types of scenarios requires protocol parameters that provide a clear identification of the independent traffic flows involved. Then, procedures are required to control the fail-over and restoration of the identified traffic flows to avoid message loss.

The SS7 MTP3 [Q.704] provide an excellent example of the types of procedures that can be applied to the problem of switching traffic flows across redundant processes[4].

1.5.1. Identification of Traffic Flows

Traffic flows between Server Processes in the UAs are managed on the basis of the Application Server to which the traffic flows correspond. Traffic flows from SG to AS are identified by the *Routing Key* or *Routing Context* to which they correspond[5].

An Application Server Process can be active and handling traffic for any number or combination of traffic flows. That is, the ASP can be actively handling traffic for any number of Application Servers. When an SCTP [RFC 2960] association fails, it is necessary to identify both the sequence of the last message succesfully received and processed by the Signalling Process, as well as the traffic flow within which that sequence applies.

Therefore, this document identifies a message in a traffic flow by the *Routing Context*, *Traffic Flow Id* and the correlation (sequence) number within that flow as identified by the *Correlation Identifier*. The *Correlation Identifier* is a combination of *Traffic Flow Id* and *Correlation Number* which is applied to all divertable traffic.

(For details on the assignment of *Traffic Flow Identifiers* and Correlation numbers, see Section 4.1.2 "Correlation".)

1.5.1.1. SGP Starting New SGP-to-ASP Traffic

When traffic is originally started for a traffic flow, the first divertable message in the traffic flow is assigned a *Correlation Number* of one (1) by the sending Signalling Process. Subsequent divertable messages within the routing context are given the *Correlation Id* number of two (2), three (3), and so on.

Because SCTP is a sequenced reliable transport [RFC 2960], it is only necessary to communicate this *Correlation Id* number between SPP peers for the intial message which is sent to the peer. Each Signalling Peer Process **MUST** be capable of counting the messages which have been sent or received on the SCTP association, and assigning each subsequent message the next sequential *Correlation Id* number.

1.5.1.2. SPP Diverting peer SPP Traffic

Should, for example, the association fail between the SGP and the ASP, the SGP will recover any buffers from categories (1), (2), (3) and (4), and immediately restart traffic, in sequence, on another active ASP within the AS. When the SGP restarts traffic on this alternate ASP, if the messages belong to Category (4) or (3) (i.e, they were transmitted on but not acknowledged by the underlying transport, or transmitted and acknowledged), the SGP will label the initial message sent with the *Correlation Id* of the message at the time that it was originally sent. When the SGP sends tmessages from Category (2), (1) and newly arriving traffic, the SGP will not tag the messages with a *Correlation Id*, but instead will label them internally with the next sequential *Correlation Numbers* for the traffic flow.

Thus, the alternate Signalling Peer Process which is receiving diverted traffic will be able to distinguish the problematic Category (3) and (4) messages from those which follow. When an tagged message is received, the Signalling Peer Process is now aware that the messages were previously sent to the normal SPP to which the SCTP association was lost. When an untagged message arrives, the receiving Signalling Peer Process is aware that this and subsequent messages within the traffic flow represent previously unsent traffic.

(Detailed procedures for the tagging of messages are described in Section 4.1.3 and 4.1.5.2.1; for diversion, Sections 4.2.2, 4.2.3 and 4.1.6.)

1.5.1.3. SPP Receiving Diverted Traffic

At the Signalling Process receiving the diverted traffic for the *Routing Context*, three actions are possible (or, combinations of the three):

(1) Ignore the *Correlation Id* and process the messages blind at the risk that message duplication will occur,

(2) discard all messages tagged with a *Correlation Id* at the risk of increased message loss, or,

(3) perform the procedures described in Section 4.1.5.2.2 minimizing the message duplication and loss resulting from the diversion.

Only by performing the procedures described in Section 4.1.5.2.2 will message duplication and loss be minimized.

1.5.1.4. SPP Restoring Traffic

Should, for example, the association recover between the SGP and ASP, the ASP will need to rebalance the load across the available SGPs and the newly available SGP. As discussed, if the ASP switches traffic immediately, message mis-sequencing can occur. Two procedures are provided by *CORID* for restoring traffic without message mis-sequencing: a Heartbeat procedure and a timer procedure.

The Heartbeat procedure withholds divertable traffic from the SGP currently active for each traffic flow and sends a *BEAT* message on each flow. Once the *BEAT ACK* is received by the ASP, the ASP is assured that there is no divertable traffic pending on the SGP and the traffic flow can be switched to the recovered SGP. The Heartbeat procedure is applicable to recovery between SGPs in the same SG as well as SGPs in different SGs.

The Timer procedure witholds divertable traffic from the SGP currently active for the traffic flow and waits until a timer expires. Once the timer expires, the ASP is resonably assured that there is no traffic pending on the SGP and the traffic flow can be switched to the recovered SGP. The Timer procedure is applicable to recovery between SGPs where the SGPs do not support *CORID*.

Restoration of traffic is described in detail in Sections 4.2.3 and 4.1.6.

1.6. Sample Configurations

A typical Example (C) configuration (multiple Signalling Gateways) is illustrated in *Figure 4*. In this configuration a number of Application Server Processes (ASPs) serving a number of Application Servers (ASs) are connected to two Signalling Gateways (SGs). The SGs appear as mated SS7 Signalling Transfer Points (STPs) [Q.705] to the SS7 Network. Traffic originating at Signalling Endpoints (SEP) in the SS7 network and directed toward SEP in the IP network (i.e., Application Servers) is loadshared over the STPs by the Signalling Link Selection (SLS) [Q.704] value associated with each message. Traffic originating at the SEP in the IP network (i.e, AS) is loadshared over the SGs in the same fashion.

2. 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 [RFC 2119].

3. Protocol Elements

The following protcol element definitions are provided by *CORID* in extension to the existing protocol element definitions for the UAs [M2UA...TUA].

3.1. Parameters

The following subsections describe the parameters used for *CORID*, their format and the message in which they are used.

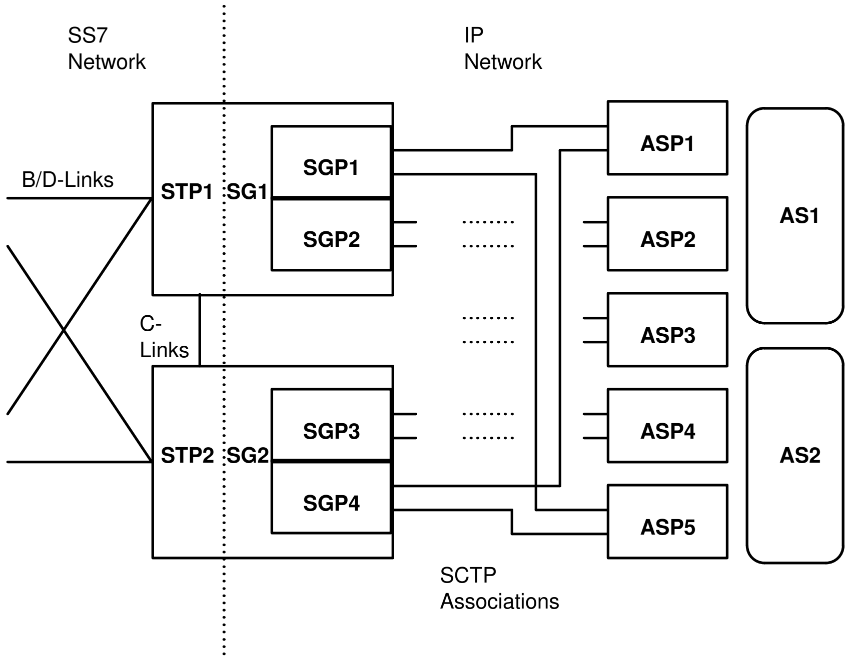

Figure 4. Example (C) Sample Multiple-SG Configuration

3.1.1. Correlation Id

The *Correlation Id* parameter is used in the *BEAT*, *BEAT ACK*, *ASPAC*, *ASPAC ACK*, and UA-User data messages. It is used here to identify data messages sent to a peer SPP.

The *Correlation Id* parameter is formatted 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 = 0x0019          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                       Correlation Id #1                       |
+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -+
|                       Correlation Id #2                       |
+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -+
\                               .                               \
/                               .                               /
\                               .                               \
/                                                               /
+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -+
|                       Correlation Id #n                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The *Correlation Id* parameter contains one or more of the following field:

Correlation Id field: 8-bytes

The *Correlation Id* field is formatted 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Correlation Number                       |
+- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -+
|                        Traffic Flow Id                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Correlation Number field: 32-bits (unsigned integer)

The *Correlation Number* field identifies a particular message within a traffic flow. When the *Correlation Id* parameter is included in the *BEAT (ACK)* or *ASPAC (ACK)* message, this field identifies the last sent message for the indicated traffic flow. When the *Correlation Id* parameter is included a UA-User data message, this field identifies the *Correlation Number* of the message in which it is contained.

Traffic Flow Id field: 32-bits (unsigned integer)

The *Traffic Flow Id* field identifies a particular indepdently sequenced traffic flow to which the *Correlation Number* field value applies. The *Traffic Flow Id* field identifies a traffic flow associated with an Application Server. When used for tagging messages or in the *BEAT (ACK)* or *ASPAC (ACK)* message for a Loadshare AS Load Selection [LOADSEL] or Loadshare Load Group [LOADGRP], the *Traffic Flow Id* field **MUST** identify traffic flow (*Load Selection*) within an Application Server.

For an Override or Broadcast AS (or Load Group), the Traffic Flow Id is not required and **SHOULD** be coded zero (0). In this case, the *Correlation Id* parameter **SHOULD** only contain one *Correlation Id* field.

For details on *Traffic Flow Id* assignment, see Section 4.1.2.2.

When the *Correlation Id* parameter is included in the *BEAT*, *BEAT ACK*, *ASPAC*, *ASPAC ACK*, and UA-User data messages, only one *Routing Context* (or *Interface Identifier*) representing a single Application Server **MUST** be associated (specified or implied) with the message.

3.2. Messages

3.2.1. ASP Active (ASPAC)

CORID supplements the *ASPAC* mesage by permitting the following optional parameters to be included in the message:

Extension Parameters

Correlation Id	Mandatory

The format of the resulting *ASPAC* 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 = 0x000b          |          Length = 8           |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                       Traffic Mode Type                       |
+-------------------------------+-------------------------------+
|         Tag = 0x0006          |          Length = 8           |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                        Routing Context                        |
+-------------------------------+-------------------------------+
|         Tag = 0x0001          |           Length=8            |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                 Interface Identifier (integer)                |
+-------------------------------+-------------------------------+
|         Tag = 0x0003          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                  Interface Identifier (text)                  /
\                                                               \
+-------------------------------+-------------------------------+
|         Tag = 0x0019          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                        Correlation Id                         /
\                                                               \
+-------------------------------+-------------------------------+
|         Tag = 0x0004          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                          Info String                          /
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The *Correlation Id* parameter is used by the ASP in the *ASPAC* message to indicate the correlation identifier for the first UA-User message to be transmitted in each traffic flow from the Application Server being activated to the Signalling Gateway. The Application Servers for which the *Correlation Id* apply is either indicated in the *ASPAC* message by providing the associated *Routing Contexts* (or *Interface Identifiers*), or, if there is no *Routing Context* (or *Interface Identifier*) parameter in the message, the associated Application Servers are implied by the SGP and ASP configuration data.

When the *Correlation Id* parameter is present in the *ASPAC* message, the message **SHOULD** only contain one *Routing Context* (or *Interface Identifier*) in the *Routing Context* (or *Interface Identifier*) parameter. When the *Correlation Id* parameter is not present, but required by the SGP, the value of the *Correlation Id* is assumed to be zero (0).

The *ASPAC* message **MAY** contain additional extension parameters provided for by other extensions.

No other changes to the *ASPAC* message format are provided by this extension.

3.2.2. ASP Active Acknowledgement (ASPAC ACK)

CORID supplements the *ASPAC ACK* mesage by permitting the following optional parameters to be included in the message:

Extension Parameters	
Correlation Id	Mandatory

The format of the resulting *ASPAC ACK* 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 = 0x000b          |          Length = 8           |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                       Traffic Mode Type                       |
+-------------------------------+-------------------------------+
|         Tag = 0x0006          |          Length = 8           |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                        Routing Context                        |
+-------------------------------+-------------------------------+
|         Tag = 0x0001          |           Length=8            |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                 Interface Identifier (integer)                |
+-------------------------------+-------------------------------+
|         Tag = 0x0003          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                   Interface Identifier (text)                 /
\                                                               \
+-------------------------------+-------------------------------+
|         Tag = 0x0019          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                        Correlation Id                         /
\                                                               \
+-------------------------------+-------------------------------+
|         Tag = 0x0004          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                          Info String                          /
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The *Correlation Id* parameter is used by the SGP in the *ASPAC ACK* message to indicate the correlation identifier for the first UA-User message to be transmitted from the Signalling Gateway to the Application Server being activated for each traffic flow. The Application Servers for which the *Correlation Id* apply is either indicated in the *ASPAC ACK* message by providing the associated *Routing Contexts* (or *Interface Identitifiers*), or, if there is no *Routing Context* or *Interface Identifier* parameter in the message, the associated Application Servers are implied by the SGP and ASP configuration data.

When the *Correlation Id* parameter is present in the *ASPAC ACK* message, the message **SHOULD** only contain one *Routing Context* (*Interface Identifier*) in the *Routing Context* (*Interface Identifier*) parameter. When the *Correlation Id* parameter is not present, but required by the ASP, the value of the *Correlation Id* is assumed to be zero (0).

The *ASPAC ACK* message **MAY** contain additional extension parameters provided for by other extensions.

No other changes to the *ASPAC ACK* message format are provided by this extension.

3.2.3. Heartbeat (BEAT)

CORID supplements the *BEAT* message by permitting the following optional parameters to tbe indicated in the message:

Extension Parameters		
Routing Context	Conditional	*1
Interface Identifier	Conditional	*2
Correlation Id	Conditional	

Note 1: The *Routing Context* parameter is only included in those UAs that support *Routing Context* [M3UA...TUA].

Note 2: The *Interface Identifier* parameter is only included in those UAs that support *Interface Identifier* [M2UA].

The format of the resulting *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 = 0x0006          |          Length = 8           |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                        Routing Context                        |
+-------------------------------+-------------------------------+
|         Tag = 0x0001          |           Length=8            |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                 Interface Identifier (integer)                |
+-------------------------------+-------------------------------+
|         Tag = 0x0003          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                  Interface Identifier (text)                  /
\                                                               \
+-------------------------------+-------------------------------+
|         Tag = 0x0019          |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                        Correlation Id                         /
\                                                               \
+-------------------------------+-------------------------------+
|         Tag = 0x0009          |             Length            |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                        Heartbeat Data                         /
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The *Routing Context* parameter is used by the SPP in the *BEAT* message to indicate the Application Server for which the message applies when the *CORID* heatbeat procedures are used. The Application Servers for which the *BEAT* message apply is either indicated in the *BEAT* message by providing the associated *Routing Contexts* (or *Interface Identifier*), or, if there is no *Routing Context* (or *Interface Identifier*) paraemter in the message, the associated Application Servers are implied by the SPP configuration data.

When the *Routing Context* (or *Interface Identifier*) is present in the *BEAT* message, the message **SHOULD** only contain one *Routing Context* (*Interface Identifier*) in the *Routing Context* (*Interface Identifier*) parameter. When the *Routing Context* (*Interface Identifier*) is not present in the *BEAT* message, but required by the SPP, the *BEAT* message is assumed to be a normal *BEAT* message not supporting the procedures of *CORID* an a normal *BEAT ACK* response **MUST** be generated.

The *Correlation Id* parameter is used by the SPP in the *BEAT* message to indicate the correlation identifier for the last UA-User message that was transmitted to the peer SPP for each traffic flow for the given SCTP stream upon which the **BEAT** message is sent. The Application Servers fro which the *Correlation Id* applies is either indicated in the *BEAT* message by providing the associated *Routing Context* (*Interface Identifier*), or, if there is no *Routing Context* (or *Interface Identifier*) parameter in the message, the associated Application Servers are implied by the SPP configuration data.

When the *Correlation Id* parameter is present in the *BEAT* message, the message **SHOULD** only contain one *Routing Context* (*Interface Identifier*) in the *Routing Context* (*Interface Identifier*) parameter. When the *Correlation Id* parameter is not present, but required by the SPP, the value of the *Correlation Id* is assumed to be zero (0) for all affected traffic flows.

The *BEAT* mesage **MAY** contain additional extension parameters provided for by other extensions.

No other changes to the *BEAT* message format are provided by this extension.

3.2.4. Heartbeat Acknowledgement (BEAT ACK)

CORID supplements the *BEAT ACK* message by permitting the following optional parameters to tbe indicated in the message:

Extension Parameters		
Routing Context	Conditional	*1
Interface Identifier	Conditional	*2
Correlation Id	Conditional	

Note 1: The *Routing Context* parameter is only included in those UAs that support *Routing Context* [M3UA...TUA].

Note 2: The *Interface Identifier* parameter is only included in those UAs that support *Interface Identifier* [M2UA].

The format of the resulting *BEAT ACK* 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 = 0x0006         |           Length = 8          |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                        Routing Context                        |
+-------------------------------+-------------------------------+
|          Tag = 0x0001         |           Length=8            |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
|                Interface Identifier (integer)                 |
+-------------------------------+-------------------------------+
|          Tag = 0x0003         |            Length             |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                  Interface Identifier (text)                  /
\                                                               \
+-------------------------------+-------------------------------+
|          Tag = 0x0019         |           Length              |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                        Correlation Id                         /
\                                                               \
+-------------------------------+-------------------------------+
|          Tag = 0x0009         |             Length            |
+- - - - - - - - - - - - - - - -+- - - - - - - - - - - - - - - -+
\                                                               \
/                        Heartbeat Data                         /
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The *Routing Context* parameter is used by the SPP in the *BEAT ACK* message to indicate the Application Server for which the message applies when the *CORID* heatbeat procedures are used. The Application Servers for which the *BEAT ACK* message apply is either indicated in the *BEAT ACK* message by providing the associated *Routing Contexts* (or *Interface Identifier*), or, if there is no *Routing Context* (or *Interface Identifier*) paraemter in the message, the associated Application Servers are implied by the SPP configuration data.

When the *Routing Context* (or *Interface Identifier*) is present in the *BEAT ACK* message, the message **SHOULD** only contain one *Routing Context* (*Interface Identifier*) in the *Routing Context* (*Interface Identifier*) parameter. When the *Routing Context* (*Interface Identifier*) is not present in the *BEAT* message, but required by the SPP, the *BEAT* message is assumed to be a normal *BEAT* message not supporting the procedures of *CORID* an a normal *BEAT ACK* response **MUST** be generated.

The *Correlation Id* parameter is used by the SPP in the *BEAT ACK* message if it appeared in the *BEAT* message. In this case, the *Correlation Id* parameter that the SPP places in the *BEAT ACK* message **MUST** be the same as that in the corrsponding received *BEAT* message.

When the *Correlation Id* parameter is present in the *BEAT ACK* message, the message **SHOULD** only contain one *Routing Context* (*Interface Identifier*) in the *Routing Context* (*Interface Identifier*) parameter.

The *BEAT ACK* mesage **MAY** contain additional extension parameters provided for by other extensions.

No other changes to the *BEAT ACK* message format are provided by this extension.

4. Procedures

CORID provides the following procedures in extension to the procedures of the UAs [M2UA...TUA].

4.1. Traffic Handling

In some circumstances, the SPP must treat traffic differently than normal in fitting with the *CORID* procedures. This traffic handling is described in the sections below:

4.1.1. Classification

Divertable messages are any UA-User messages destined for an Application Server. Divertable messages are UA-User data and some management (non-ASP management) messages that have an explicit or implied *Routing Context* (*Interface Identifier*) and have strict requirements preventing loss, duplication or mis-sequencing. All SSNM messages containing an explicit or implied *Routing Context* **SHALL** be classified as divertable, with the exception of *DAUD* which **SHOULD** be classified as divertable between ASPs or SGPs belonging to the same SG, and **SHOULD** be classified as non-divertable between SGs.

UA-User messages that qualify as divertable messages in addition to SSNM are listed in *Table 1*. Although some messages in some message classes might be considered as non-divertable, all messages in the message classes listed in *Table 1* **SHALL** be treated as divertable.

Table 1. Divertable Messages by UA

UA	**Class**	**Msg**	**Notes**
M2UA	MAUP	Data Data Ack	
M3UA	Transfer	DATA	
SUA	CL	CLDT CLDR	Note 1
	CO	CORE COAK CODT RESRE RESCO RELRE	
TUA	TCM	TUNI TQRY TCNV TRSP	Note 2
		TUAB TPAB TNOT	
	CHM	CINV CRES	Note 3
		CERR CREJ CCAN	
All	SSNM	DUNA DAVA SCON DUPU	Note 4
		DAUD	Note 5

Note 1: All those marked "Return on Error".
Note 2: All those without components or marked "Return on Error".
Note 3: All those in operation class 1, 2 or 3.
Note 4: All those with implied *Routing Context* or containing explicit *Routing Context* parameters in the message.
Note 5: See Section 4.1.1.

4.1.2. Correlation

Each independent traffic flow for a given Application Server as identified by a *Routing Context* (*Interface Identifier*) **MUST** be correlated using a *Correlation Id*. The *Correlation Id* consists of a *Correlation Number* and a traffic flow identifier. The *Correlation Number* is used to number each message within the given traffic flow.

4.1.2.1. Assignment of Correlation Ids

To accomodate all combinations of traffic modes at AS and SG, divertable messages are correlated by independent traffic flow. That is, each sent divertable message is labelled with a traffic flow identifier and a *Correlation Number* for the AS that is incremented for each message sent for the traffic flow. In the same fashion, each received divertable message is labelled with the identity of the traffic flow on which it was received and a *Correlation Number* for the AS that is incremented for each message received on that traffic flow.

An SPP maintains two correlation counters for each traffic flow for each AS: for each traffic flow, one counter tracks the *Correlation Number* of messages sent to the AS and the other tracks the *Correlation Number* of messages received from the AS. Before traffic is started for an AS on a traffic flow, these counters are set to zero (0). The first divertable message for the AS on the flow **MUST** then be assigned a coorrelation number of one (1); and subsequent divertable messages, the *Correlation Number* of two (2), three (3), and so forth.

Whenever traffic is started for the AS (using the ASP Active Procedures), the correlation counters **SHALL** be synchronized by exchanging correlation numbers and traffic flow identifiers in the *Correlation Id* parameter in the *ASPAC* and *ASPAC ACK* messages. For new traffic, the *Correlation Number* **MUST** zero (0); for restarting traffic, it is **SHOULD** be the *Correlation Number* of the last message transferred. (See Section 4.2.3.)

4.1.2.2. Assignment of Traffic Flow Ids

Traffic Flow Ids **SHALL** identify a switchable traffic flow within an Application Server. The *Traffic Flow Id* value **SHALL** be assigned by the sending SPP[6].

Traffic Flow Ids assigned by an SPP and **MUST** be communicated to the peer SPP in an *ASPAC* or *ASPAC ACK* message.

For traffic distributions that do not loadshare (i.e, Override and Broadcast), the *Traffic Flow Id* field is not required and **MAY** be set to zero (0). In this case, the *Correlation Id* parameter in the *BEAT*, *BEAT ACK*, *ASPAC* or *ASPAC ACK* message **SHOULD** only contain one *Correlation Id* field (see Section 3.1.1).

Following are rules for the assignment of *Traffic Flow Ids* at an SPP:

(i) If an SPP belongs to a regular Override or Broadcast AS, no *Traffic Flow Id* need be assigned or included by the SPP in the *Correlation Id* parameter.

(ii) If an SPP belongs to a regular Loadshare AS, a *Traffic Flow Id* is assigned and included in the *Correlation Id* parameter. The *Traffic Flow Id* Id assigned **MUST** unambiguously identify the traffic flow within the AS.

(iii) If an SPP belongs to a Load Selector [LOADSEL], a *Traffic Flow Id* is assigned and included in the *Correlation Id* parameter regardless of the *Traffic Mode Type* of the AS. The *Traffic Flow Id* assigned **MUST** unambiguously identify the the Load Selection within the AS.[7]

(iv) If an SPP belongs to a Load Group [LOADGRP], a *Traffic Flow Id* is assigned and included in the *Correlation Id* for a Loadshare AS or Load Group. An assigned *Traffic Flow Id* **MUST** unambiguously identify the Load Selection within the AS. For a non-loadshare AS and Load Group, no *Traffic Flow Id* need be assigned or included in the *Correlation Id* parameter.[8]

4.1.3. Tagging

Each sent or received message for an AS is labelled when it is first sent or received. The message is labelled with the traffic flow id associated with the SPP to or from which the message was sent or received, and the correlation number assigned within the traffic flow (see Section 4.1.2.1).

Tagged messages contain a *Correlation Id* parameter: an untagged message is tagged by adding a *Correlation Id* parameter to the message. When a message is tagged, it **SHALL** be tagged with the same values of the traffic flow id (if required) and *Correlation Number* with which it was originally labelled.

Although each message is labelled with a traffic flow id and correlation number, the message is not necessarily tagged with the *Correlation Id* parameter when the message is sent. Messages for an AS that are sent for the first time **MUST NOT** be tagged. Messages retransmitted **MUST** be tagged.

4.1.4. Buffering

To support *CORID* and SPP will have to, under some circumstances, buffer messages. When divering traffic, the SPP requires buffers to hold unsent messages awaiting diversion; when sending traffic, the SPP requires buffers to hold local copies of sent messages in the event of failure.

4.1.4.1. SPP witholding unsent messages

CORID procedures require that an SPP at times withhold AS traffic. To perform this, the SPP allocates a diversion buffer and places in the buffer all subsequent messages that would otherwise be sent to the SPP for the AS.

4.1.4.2. Local copies of sent messages

To reduce loss of messages, an SPP **SHOULD** buffer messages until it can be assured that the peer SPP has received and processed the message. When a message is sent to an SPP supporting *CORID* a local copy of the message **MUST** be kept until it is discarded in accordance with a *CORID* procedure.[9]

(i) A local copy **SHOULD NOT** be discarded when it is acknowledged by the peer SCTP.

(ii) a local copy **SHOULD NOT** be discarded until the sending SPP is confident that the peer SPP has received and processed the message.

(iii) To ensure that stale messages do not propagate through the system, an SPP **SHOULD NOT** keep local copies of sent messages for longer than a maximum lifetime T(lifetime). Any local coppies of sent messages that are older (measured from the moment at which they were sent to the peer SPP) than T(lifetime) **SHOULD** be discarded.

4.1.5. Message Handling

The SPP supporting *CORID* handles tagged and untagged messages differently.

4.1.5.1. Untagged Messages

Under some circumstances, an SPP will send or receive untagged messages. Untagged messages (see Section 4.1.3) are messages which do not contain a *Correlation Id* parameter.

4.1.5.1.1. SPP sending untagged messages

An SPP sends untagged messages to a peer SPP whenever the message is being sent for an Application Server for the first time. All divertable messages which have been transmitted for the first time **MUST NOT** be sent tagged.

Local copies of untagged messages awaiting acknowledgement or expiry are labelled with the *Routing Context* (*Interface Identifier*) for the Application Server to which they were sent, the traffic flow id of the SPP to which they were sent, and the *Correlation Number* of the message. The *Correlation Number* with which a message is labelled **MUST** be the next sequential *Correlation Number* for the AS and traffic flow. These labels can be used later to tag a message that is marked for diversion.

4.1.5.1.2. SPP receiving untagged messages

When an SPP receives an untagged message, it associates with the message the next sequential *Correlation Number* for the *Routing Context* (*Interface Identifier*) and traffic flow id for which the message was received. Untagged messages are received in order and **MAY** be processed when received. The SPP **SHOULD** keep track of the *Correlation Id*s that have been processed for the AS.

4.1.5.2. Tagged Messages

Under some circumstances, an SPP will send or receive tagged messages. Tagged messages (see Section 4.1.3) are messages which contain a *Correlation Id* parameter.

4.1.5.2.1. SPP sending tagged messages

An SPP sends tagged traffic whenever it sends traffic that is marked for diversion. That is, whenever an SPP sends divertable messages to an SPP other than the original SPP for which those messages were labelled, the SPP **MUST** tag the message with the *Correlation Id* parameter that contains the labelled traffic flow id (if required) and *Correlation Number*.

In addition, when a ASP becomes active for a Broadcast AS, an SGP **MUST** tag the first message in each traffic flow towards the ASP to allow the ASP to synchronize its entry into the Broadcast AS.

4.1.5.2.2. SPP receiving tagged messages

The handling of tagged messages is the mechanism that provides for the reduction of message loss, duplication and mis-sequencing. An SPP receiving divertable messages containing a *Correlation Id* parameter **SHALL** perform the following actions:

(i) The SPP determines (by implementation-dependent means [10]) whether the message has already been processed for the AS.

(ii) If the message has not already been processed for the AS, it is processed as normal.

(iii) If the message has already been processed for the AS, it is discarded.

(iv) If, as a result of some failure, the SPP cannot determine with any certainly whether the tagged message has been processed for the AS, or not, the SPP **MUST** discard the message[11].

4.1.5.3. Heartbeat Messages

Under some circumstances, an SPP will send or receive *BEAT* messages with the intention of pushing the messages on the stream on which the *BEAT* message is sent.

4.1.5.3.1. SPP sending BEAT messages

An SPP sends *BEAT* messages whenever it witholds traffic to or from an AS in preparation for diversion. That is, whenever an SPP withholds divertable messages, the SPP **MUST** send a *BEAT* message with an implied Application Server or explicit *Routing Context* (*Interface Identifier*) plus the *Correlation Id* parameter with the *Traffic Flow Ids* for a particularl stream, on each stream used by the AS for which traffic is being diverted.

4.1.5.3.2. SPP receiving BEAT messages

The handling of *BEAT* messages is the mechanism that provide for the reduction of message loss, duplication and mis-sequencing during diversion between active SPP. An SPP receiving a *BEAT* message containing an explicit or implied *Routing Context* (*Interface Identifier*) and *Correlation Id* parameter on a stream other than stream zero (0) **SHALL** perform the following actions:

(i) The SPP will wait until any internal queue of messages for the Application Server indicated by the *Routing Context* (*Interface Identifier*) and the traffic flows indicated by the *Correlation Id* parameter in the *BEAT* message have drained.

(ii) If the SGP can determine the traffic flows in the SS7 network which require changeback, the SPP **MAY** then initiate a changeback procedure [Q.704] to the SS7 network and await completion of the changeback procedure.

(iii) Once internal message queues for the Application server have drained (i.e. all messages for the indicated Application Server have been processed), and any changeback procedure to the SS7 network has completed at an SGP, the SPP will respond with a *BEAT ACK* message which contains the *Routing Context* (*Interface Identifier*) parameter, the *Correlation Id* parameter unchanged, and the opqaue information contained in the *Heatbeat Data* parameter of the *BEAT* message. (This *BEAT ACK* message may be sent on any stream.)

4.1.6. Diversion

When an SPP supporting *CORID* wishes to reroute traffic from one SPP or AS to another, it performs a diversion.

4.1.6.1. SPP diverting traffic from a failed, deactivated or overridden peer SPP

When diverting traffic due to a failed, deactivated or overridden peer SPP, the diverting SPP will be in one of the following situations:

(i) no alternate SPP exists,

(ii) an alternate SPP exists in the same AS or SG,

(iii) an alternate SPP exists in a different AS or SG.

4.1.6.1.1. Alternate SPP in same AS or SG, or No Alternate SPP

When an SPP diverts AS traffic away from a failed, deactivated or overridden peer SPP to an alternate peer SPP in the same AS or SG, the SPP **SHALL** perform the following actions:

(i) The SPP tags (see Section 4.1.3) each untagged message that is marked for diversion.

(ii) If an alternate SPP is available (active for the AS), the SPP sends the messages marked for divertion to the alternate SPP.

(iii) If no alternate SPP exists (the AS is AS-PENDING), the SPP buffers the marked messages in a buffer used for buffering messages while the AS is in the AS-PENDING state.

(iv) The SPP then diverts AS traffic, beginning with traffic withheld for the AS, to the alternate SPP or AS-PENDING buffer.

This procedure corresponds to the Sequenced Changeover procedure used by the SS7 MTP [Q.704].

4.1.6.1.2. Alternate SPP in different AS or SG

When an SPP diverts AS traffic away from a failed or deactivated peer SPP to an alternate peer SPP in a different AS or SG, the SPP **SHALL** perform the following actions:

(i) The SPP starts timer T(divert) and continues buffering AS traffic until the timer expires.

(ii) When T(divert) expires, and the failed or deactivated SPP has not recovered, the SPP continues with the following actions:

(iii) The SPP discards all tagged messages and messages marked for diversion.

(iv) The SPP starts AS traffic, beginning with the contents of the diversion buffer, to the alternate SPP.[12]

This procedure corresponds to the Time-Controlled Changeover procedure used by the SS7 MTP [Q.704].

4.1.6.2. SPP diverting traffic from an active peer SPP

When an SPP wishes to divert AS traffic away from an active peer SPP, the SPP **SHALL** perform the following actions:

(i) The SPP witholds and buffers AS traffic for the SPP from which the traffic is being diverted.

(ii) The SPP sends a *BEAT* message with the associated *Routing Context* (*Interface Identifier*) and the *Traffic Flow Ids* being diverted in the *Correlation Id* parameter, plus a unique identifier[13] in the *Heartbeat Data* parameter on each SCTP stream on which the traffic being withheld for diversion was previously sent. The *Correlation Id* parameter **SHOULD** only contain the *Traffic Flow Ids* that correspond to traffic flows on the SCTP stream upon which the particular *BEAT* message is sent. If the Application Server is not implied by the SCTP association, the *BEAT* message must also contain the *Routing Context* (*Interface Identifier*) coresponding to the Application Server.

(iii) The SPP starts a timer T(restore).

(iv) If the SPP receives the *BEAT ACK* message(s) for the concerned Application Server that contain the unique identifier(s) in the *Heartbeat Data* parameter before timer T(restore) expires, the SPP diverts the traffic, beginning with the withheld traffic, to the target SPP and cancels the T(restore) timer.

(v) If the timer T(restore) expires, the diverting SPP diverts traffic, beginning with the withheld traffic, to the target SPP.

(vi) If an SPP receives a *BEAT ACK* message(s) for the concerned Application Server containing a unique identifier for which the timer T(restore) has already expired, the SPP ignores the message.

The purpose of this *BEAT* procedure is to avoid mis-sequencing by ensuring that all messages sent for the AS to the old SPP have been processed before messages are sent to the new SPP. This avoids races between (and possible mis-sequencing of) messages sent on the old SPP and messages sent on the new SPP.

This procedure corresponds to the Changeback procedure used by the SS7 MTP [Q.704].

4.2. ASP Management Procedures

CORID extends the ASP Management procedures of the UAs with the following procedures:

4.2.1. ASP Down Procedures

CORID extends the ASP Down procedures of the UAs as follows:

4.2.1.1. SPP detecting loss of SCTP association

When an SPP receives an SCTP COMMUNICATION LOST or RESTART indication and there are Application Servers active for the association, the SPP **SHALL** perform the following actions with regard to active AS traffic for the association:

(i) The SPP witholds AS traffic for the peer SPP in a diversion buffer.

(ii) The SPP marks for diversion all local copies of AS messages already sent to the peer SPP.

(iii) The SPP then **SHALL** perform the actions described in Section 4.1.6.1.

4.2.1.2. ASP sending ASPDN

An ASP **MUST NOT** send an *ASPDN* message until it has completed the ASP Inactive Procedures with the intended SGP for every AS.

4.2.1.3. SGP or IPSP receiving ASPDN

An SGP or IPSP, upon rreceiving an *ASPDN* message from an ASP-ACTIVE ASP, **MUST** perform the ASP Inactive Procedures with regard to *CORID* (see Section 4.2.2.2) for every AS for which the ASP is ASP-ACTIVE and then complete the *ASPDN* procedures.

4.2.1.4. ASP receiving ASPDN ACK

An SGP or IPSP, upon receiving an unsolicited *ASPDN ACK* message from an active SGP, **MUST** perform the ASP Inactive Procedures with regard to *CORID* (see Section 4.2.2.3) for every AS for which the ASP is ASP-ACTIVE and then complete the *ASPDN ACK* procedures.

4.2.2. ASP Inactive Procedures

CORID extends the ASP Inactive procedures of the UAs as follows:

4.2.2.1. ASP sending ASPIA

When an ASP wishes to deactivate an Application Server with an SGP, the ASP **SHALL** perform the following actions for traffic pertaining to the AS:

(i) The ASP withholds sending AS traffic to the SGP or IPSP.

(ii) The ASP stops processing AS traffic recevied from the SGP or IPSP. Any messages received for the Application Server after the last processed message **MAY** be discarded.

(iii) The ASP starts a T(divert) timer.

(iv) The ASP **SHALL** perform the applicable UA ASP Inactive Procedures[14].

4.2.2.2. SGP receiving ASPIA or sending ASPIA ACK

An SGP receiving an *ASPIA* message for an AS, or wishing to send an unsolicited *ASPIA ACK* to deactivate an AS, **SHALL** perform the following actions for the traffic pertaining to each AS for which deactivation is performed:

(i) The SGP withholds sending AS traffic to the ASP.

(ii) The SGP stops processing AS traffic received from the ASP. Any messages received for the AS at the SGP after receiving the *ASPIA* message **MUST** be discarded.

(iii) The SGP marks for diversion all local copies of AS messages sent to the ASP.

(iv) The SGP then **SHALL** perform the actions described in Section 4.1.6.1.

(v) The ASP **SHALL** perform the applicable UA ASP Inactive Procedures[14].

4.2.2.3. ASP receiving ASPIA ACK

Upon receiving an *ASPIA ACK* message the ASP **SHALL** perform the following actions for the traffic pertaining to the AS identified by the *Routing Context* (*Interface Identifier*) in the received *ASPIA ACK* message or implied by the SCTP association on which the *ASPIA ACK* message was received:

(i) The T(divert) timer is cancelled (if running).

(ii) The ASP marks for diversion any local copies of AS messages sent to the SGP.

(iii) The ASP then **SHALL** perform the actions described in Section 4.1.6.1.

(iv) The ASP **SHALL** perform the applicable UA ASP Inactive Procedures[14].

4.2.2.4. T(divert) timer expiry

If the T(divert) timer expires before receiving an *ASPIA ACK* for the AS, the ASP **SHALL** perform the actions described in Section 4.2.2.3.

4.2.3. ASP Active Procedures

CORID extends the ASP Active procedures of the UAs as follows:

4.2.3.1. ASP sending ASPAC

When an ASP wishes to activate an Application Server for an SGP, the ASP **SHALL** perform the following actions for traffic pertaining to the AS:

(i) The ASP determines the *Correlation Number* of the last message sent to this SGP for the AS for each traffic flow.

(ii) If the ASP has not sent a message to the SGP for the traffic flow, the *Correlation Number* zero (0) is used.

(iii) If the ASP has sent messages to the SGP for the traffic flow, but cannot determine the *Correlation Number* of the last message sent due to local failure, the *Correlation Number* zero (0) is used.

(iv) The ASP includes the *Correlation Number*(s) determined above in the *Correlation Id* parameter in the *ASPAC* message used to active the AS. (See Section 3.1.1.)

(v) The ASP **SHALL** perform the applicable UA ASP Active Procedures[15].

4.2.3.2. SGP receiving ASPAC

When an SGP receives an *ASPAC* message for an Application Server, the SGP **SHALL** perform the following actions with regard to traffic for the AS:

(i) The SGP sets the *Correlation Number* of the next received message from the ASP for each traffic flow to the value, contained in the *Correlation Id* parameter in the *ASPAC ACK* message, plus one (1).

(ii) The SGP determines the *Correlation Number* of the last message sent to this ASP for each traffic flow.

(iii) If the SGP has not sent a message to the ASP for a traffic flow, the *Correlation Number* zero (0) is used.

(iv) If the SGP has sent messages to the ASP for a traffic flow, but cannot determine the *Correlation Number* of the last message sent due to local failure, the *Correlation Number* zero (0) is used.

(v) The SGP includes the *Correlation Number*(s) determined above in the *Correlation Id* parameter in the *ASPAC ACK* message used to acknowledge activation of the AS. (See Section 3.1.1.)

(vi) The SGP **SHALL** perform the applicable UA ASP Active Procedures[15], including the sending of *ASPIA ACK*.

(vii) The SGP then **SHALL** perform the actions described in Section 4.1.6.2.

4.2.3.3. ASP receiving ASPAC ACK

When an ASP receives an expected *ASPAC ACK* message for an Application Server, the ASP **SHALL** perform the following actions with regard to AS traffic:

(i) The ASP sets the *Correlation Number* of the next received message from the SGP for each traffic flow to the value, contained in the *Correlation Id* parameter in the *ASPAC ACK* message, plus one (1).

(ii) The ASP **SHALL** perform the applicable UA ASP Active Procedures[15].

(iii) The ASP then **SHALL** perform the actions described in Section 4.1.6.2.

If an ASP receives an unexpected *ASPAC ACK* (i.e, one for which no ASPAC was sent and the ASP is already in the ASP-ACTIVE state for the AS), then the ASP **SHALL** ignore the message for the purposes of *CORID*. The ASP **SHALL**, however, perform the applicable UA ASP Active Procedures[15].

4.3. Interworking Procedures

Because the *CORID* procedures provided here rely upon close synchronization of *Correlation Number* between SPP, if one of the SPP does not support these *CORID* procedures, neither SPP is able to take advantage of the full benefits of the procedures. The SPP supporting *CORID* **MAY** fall back to the interworking procedures provided in this section, or to procedures based on the original (non-*CORID*) UA procedures.

A peer SPP that does not support the *CORID* procedures can either be identified by local configuration information, the ASP Extenstions [ASPEXT] procedure, or at ASP Activation time. The lack of support for *CORID* can be determined at ASP Activation time when the peer SPP does not place a *Correlation Id* parameter (as it **MUST** if both peers support *CORID*) in the *ASPAC (ACK)* message.

When interworking to an SPP that does not support *CORID*, the SPP supporting *CORID* **SHALL** perform all of the procedures as though the peer SPP supported *CORID* with the following exceptions:

(i) The SPP **MUST NOT** send messages marked for diversion and tagged to the peer SPP not supporting *CORID*. All such messages **MAY** be discarded.

(ii) When diverting traffic between a failed, deactivated or overriden peer SPP and an alternate peer SPP not supporting *CORID*, the actions described in Section 4.1.6.1.2 **MUST** always be used instead of the procedures in Section 4.1.6.1.1, except when there is no alternate SPP.

(iii) When diverting traffic from an active peer SPP not supporting *CORID*, the actions described in Section 4.1.6.2 **SHALL** be followed with the exception of Section 4.1.6.2(ii), (iv) and (vi), which **MUST NOT** be performed.

(iv) The SPP **MUST NOT** place a *Correlation Id* parameter in the *ASPAC* or *ASPAC ACK*. So, the actions described in Sections 4.2.3.1(i)-(iv), 4.2.3.2(i)-(v) and 4.2.3.3(i)-(ii) do not apply.

(v) The SPP **MUST NOT** place a *Routing Context* (*Interface Identifier*) paramete rin the *BEAT* or *BEAT ACK*. So, the actions described in Sections 4.1.6.2(ii), (iv), and (vi) do not apply.

5. Examples

5.1. Example Configuration

5.2. Initialization

Figure 5 illustrates the initialization sequence that is used for all of the examples .

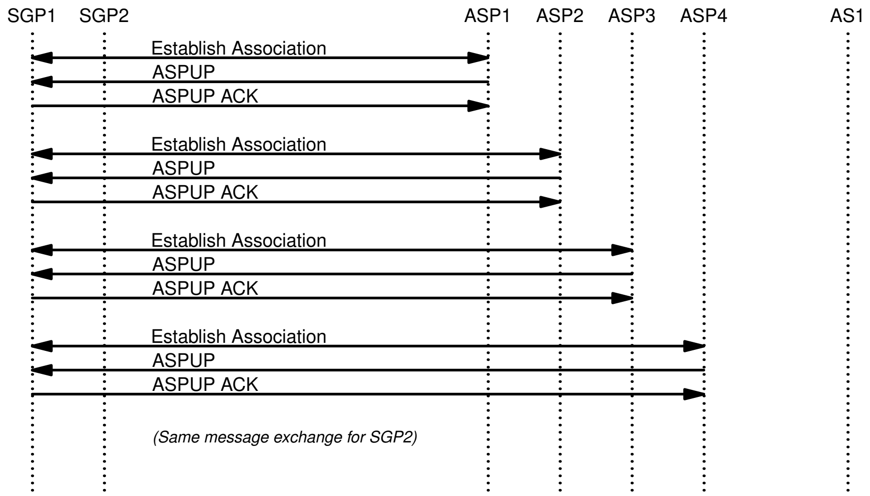

Figure 5. Initialization for Examples

The sequence of events in the exmaple illustrated in *Figure 5* is as follows:

(1) ASP1 establishes an SCTP association to SGP1 and sends an *ASUP* message. SGP1 responds with an *ASPUP ACK*.

(2) ASP2 establishes an SCTP association to SGP1 and sends an *ASUP* message. SGP1 responds with an *ASPUP ACK*.

(3) ASP3 establishes an SCTP association to SGP1 and sends an *ASUP* message. SGP1 responds with an *ASPUP ACK*.

(4) ASP4 establishes an SCTP association to SGP1 and sends an *ASUP* message. SGP1 responds with an *ASPUP ACK*.

(5) ASP1 establishes an SCTP association to SGP2 and sends an *ASUP* message. SGP2 responds with an *ASPUP ACK*. ASP2 establishes an SCTP association to SGP2 and sends an *ASUP* message. SGP2 responds with an *ASPUP ACK*. ASP3 establishes an SCTP association to SGP2 and sends an *ASUP* message. SGP2 responds with an *ASPUP ACK*. ASP4 establishes an SCTP association to SGP2 and sends an *ASUP* message. SGP2 responds with an *ASPUP ACK*.

5.3. Starting Traffic

These are examples of starting traffic.

5.3.1. Initial Startup

Figure 6 illustrates an example of an ASP joining a Loadshare Application Server.

```
    SGP1  SGP2                          ASP1 ASP2 ASP3 ASP4    AS1
     :     :                             :    :    :    :       :
(1)  :<----:--ASPAC (RC1)----------------:    :    :    :       :
     :     :                             :    :    :    :       :
(2)  :-----:--ASPAC ACK (RC1)----------->:    :    :    :       :
     :     :                             :    :    :    :       :
     :<====:==DATA=======================:    :    :    :       :
     :     :                             :    :    :    :       :
(3)  :-----:--NTFY (RC1,AS-ACTIVE)------>:    :    :    :       :
     :-----:--NTFY (RC1,AS-ACTIVE)-------:--->:    :    :       :
     :-----:--NTFY (RC1,AS-ACTIVE)-------:----:--->:    :       :
     :-----:--NTFY (RC1,AS-ACTIVE)-------:----:----:--->:       :
     :     :                             :    :    :    :       :
     :=====:=DATA=======================>:    :    :    :       :
     :     :                             :    :    :    :       :
(4)  :     :<-ASPAC (RC1)----------------:    :    :    :       :
     :     :                             :    :    :    :       :
(5)  :     :--ASPAC ACK (RC1)----------->:    :    :    :       :
     :     :                             :    :    :    :       :
     :     :<=DATA=======================:    :    :    :       :
     :     :                             :    :    :    :       :
(6)  :     :--NTFY (RC1,AS-ACTIVE)------>:    :    :    :       :
     :     :--NTFY (RC1,AS-ACTIVE)-------:--->:    :    :       :
     :     :--NTFY (RC1,AS-ACTIVE)-------:----:--->:    :       :
     :     :--NTFY (RC1,AS-ACTIVE)-------:----:----:--->:       :
     :     :                             :    :    :    :       :
     :     :==DATA======================>:    :    :    :       :
     :     :                             :    :    :    :       :
     :     :                             :    :    :    :       :
```

Figure 6. Example – Initial Startup

The sequence of events in the exmaple illustrated in *Figure 6* is as follows:

(1) ASP1 sends an ASPAC message to SGP1 contining the *Routing Context* (*Interface Identifier*) corresponding to AS1 (RC1 or IID1). Because ASP1 has never sent traffic to SGP1 for AS1, the initial value of all *Correlation Numbers* for each traffic flow activated is zero (0) and the *Correlation Id* parameter need not be included in the *ASPAC* message. (See Section 4.2.3.1.)

(2) SGP1 sends an ASPAC ACK message to SGP1 in response. Because SGP1 has never send traffic to ASP1 for AS1, the initial value of all *Corrleation Numbers* for each traffic flow activated is zero (0) and the *Corredlation Id* parameter need not be included in the *ASPAC* message. (See Section 4.2.3.2.)

Test.

(3)

(4)

(5)

(6)

(7)

5.3.2. Joining a Broadcast

Figure 7 illustrates an example of an ASP joining a Broadcast Application Server.

```
      SGP1  SGP2                            ASP1 ASP2 ASP3 ASP4    AS1
       :     :                               :    :    :    :       :
 (1)   :<----:-Establish Association------>:    :    :    :       :
       :<----:-ASPUP-----------------------:    :    :    :       :
       :-----:-ASPUP ACK------------------>:    :    :    :       :
       :     :                               :    :    :    :       :
 (2)   :<----:-Establish Association-------:--->:    :    :       :
       :<----:-ASPUP-----------------------:----:    :    :       :
       :-----:-ASPUP ACK-------------------:--->:    :    :       :
       :     :                               :    :    :    :       :
 (3)   :<----:-Establish Association-------:----:--->:    :       :
       :<----:-ASPUP-----------------------:----:----:    :       :
       :-----:-ASPUP ACK-------------------:----:--->:    :       :
       :     :                               :    :    :    :       :
 (4)   :<----:-Establish Association-------:----:----:--->:       :
       :<----:-ASPUP-----------------------:----:----:----:       :
       :-----:-ASPUP ACK-------------------:----:----:--->:       :
       :     :                               :    :    :    :       :
       :     : (Same message exchange for SGP2) :    :    :       :
       :     :                               :    :    :    :       :
```

Figure 7. Example – Joining a Broadcast AS

The sequence of events in the exmaple illustrated in *Figure 7* is as follows:

(1)

(2)

(3)

(4)

(5)

(6)

(7)

5.4. Fail-Over, Deactivation and Blocking

These are examples of fail-over, deactivation and blocking.

5.4.1. Association Recovery – Loadshare

Figure 8 illustrates an example of SCTP association recovery in a Loadshare Application Server.

```
      SGP1  SGP2                            ASP1 ASP2 ASP3 ASP4    AS1
       :     :                               :    :    :    :       :
 (1)   :<----:-Establish Association------>:    :    :    :       :
       :<----:-ASPUP-----------------------:    :    :    :       :
       :-----:-ASPUP ACK------------------>:    :    :    :       :
       :     :                               :    :    :    :       :
 (2)   :<----:-Establish Association-------:--->:    :    :       :
       :<----:-ASPUP-----------------------:----:    :    :       :
       :-----:-ASPUP ACK-------------------:--->:    :    :       :
       :     :                               :    :    :    :       :
 (3)   :<----:-Establish Association-------:----:--->:    :       :
       :<----:-ASPUP-----------------------:----:----:    :       :
       :-----:-ASPUP ACK-------------------:----:--->:    :       :
       :     :                               :    :    :    :       :
 (4)   :<----:-Establish Association-------:----:----:--->:       :
       :<----:-ASPUP-----------------------:----:----:----:       :
       :-----:-ASPUP ACK-------------------:----:----:--->:       :
       :     :                               :    :    :    :       :
       :     : (Same message exchange for SGP2) :    :    :       :
       :     :                               :    :    :    :       :
```

Figure 8. Example – Association Recovery

The sequence of events in the exmaple illustrated in *Figure 8* is as follows:

(1)

(2)

(3)

(4)

(5)

(6)

(7)

5.4.2. Assocation Failure – Override

Figure 9 illustrates an example of SCTP association failure in an Override Application Server.

```
      SGP1  SGP2                          ASP1 ASP2 ASP3 ASP4    AS1
        :     :                             :    :    :    :       :
 (1)    :<----:-Establish Association------>:    :    :    :       :
        :<----:-ASPUP-----------------------:    :    :    :       :
        :-----:-ASPUP ACK------------------>:    :    :    :       :
        :     :                             :    :    :    :       :
 (2)    :<----:-Establish Association-------:--->:    :    :       :
        :<----:-ASPUP-----------------------:----:    :    :       :
        :-----:-ASPUP ACK-------------------:--->:    :    :       :
        :     :                             :    :    :    :       :
 (3)    :<----:-Establish Association-------:----:--->:    :       :
        :<----:-ASPUP-----------------------:----:----:    :       :
        :-----:-ASPUP ACK-------------------:----:--->:    :       :
        :     :                             :    :    :    :       :
 (4)    :<----:-Establish Association-------:----:----:--->:       :
        :<----:-ASPUP-----------------------:----:----:----:       :
        :-----:-ASPUP ACK-------------------:----:----:--->:       :
        :     :                             :    :    :    :       :
        :     : (Same message exchange for SGP2) :    :    :       :
        :     :                             :    :    :    :       :
```

Figure 9. Example – Assocation Failure

The sequence of events in the exmaple illustrated in *Figure 9* is as follows:

(1)

(2)

(3)

(4)

(5)

(6)

(7)

5.4.3. Deactivation – Loadshare

This is an example of deactivation of an ASP in a Loadshare Application Server.

5.4.4. Management Blocking – Override

Figure 10 illustrates an example of management blocking of an SGP in an Override Application Server.

```
      SGP1  SGP2                          ASP1 ASP2 ASP3 ASP4    AS1
       :     :                             :    :    :    :       :
 (1)   :<----:-Establish Association------>:    :    :    :       :
       :<----:-ASPUP-----------------------:    :    :    :       :
       :-----:-ASPUP ACK------------------>:    :    :    :       :
       :     :                             :    :    :    :       :
 (2)   :<----:-Establish Association-------:--->:    :    :       :
       :<----:-ASPUP-----------------------:----:    :    :       :
       :-----:-ASPUP ACK-------------------:--->:    :    :       :
       :     :                             :    :    :    :       :
 (3)   :<----:-Establish Association-------:----:--->:    :       :
       :<----:-ASPUP-----------------------:----:----:    :       :
       :-----:-ASPUP ACK-------------------:----:--->:    :       :
       :     :                             :    :    :    :       :
 (4)   :<----:-Establish Association-------:----:----:--->:       :
       :<----:-ASPUP-----------------------:----:----:----:       :
       :-----:-ASPUP ACK-------------------:----:----:--->:       :
       :     :                             :    :    :    :       :
       :     : (Same message exchange for SGP2) :    :    :       :
       :     :                             :    :    :    :       :
```

Figure 10. Example – Deactivation

The sequence of events in the exmaple illustrated in *Figure 10* is as follows:

(1)

(2)

(3)

(4)

(5)

(6)

(7)

5.5. Recovery

These are examples of recovery.

5.5.1. Association Recovery – Loadshare

Figure 11 illustrates an example of the recovery of an ASP in a Loadshare Application Server.

```
      SGP1  SGP2                          ASP1 ASP2 ASP3 ASP4    AS1
       :     :                             :    :    :    :       :
 (1)   :<----:-Establish Association------>:    :    :    :       :
       :<----:-ASPUP-----------------------:    :    :    :       :
       :-----:-ASPUP ACK------------------>:    :    :    :       :
       :     :                             :    :    :    :       :
 (2)   :<----:-Establish Association-------:--->:    :    :       :
       :<----:-ASPUP-----------------------:----:    :    :       :
       :-----:-ASPUP ACK-------------------:--->:    :    :       :
       :     :                             :    :    :    :       :
 (3)   :<----:-Establish Association-------:----:--->:    :       :
       :<----:-ASPUP-----------------------:----:----:    :       :
       :-----:-ASPUP ACK-------------------:----:--->:    :       :
       :     :                             :    :    :    :       :
 (4)   :<----:-Establish Association-------:----:----:--->:       :
       :<----:-ASPUP-----------------------:----:----:----:       :
       :-----:-ASPUP ACK-------------------:----:----:--->:       :
       :     :                             :    :    :    :       :
       :     : (Same message exchange for SGP2) :    :    :       :
       :     :                             :    :    :    :       :
```

Figure 11. Example – Association Recovery

The sequence of events in the exmaple illustrated in *Figure 11* is as follows:

(1)

(2)

(3)

(4)

(5)

(6)

(7)

5.5.2. AS-Pending Recovery

Figure 12 illustrates an example of the recovery of an ASP for an AS in the AS-PENDING state.

```
          SGP1  SGP2                          ASP1 ASP2 ASP3 ASP4     AS1
           :     :                             :    :    :    :        :
    (1)    :<----:-Establish Association------>:    :    :    :        :
           :<----:-ASPUP-----------------------:    :    :    :        :
           :-----:-ASPUP ACK------------------>:    :    :    :        :
           :     :                             :    :    :    :        :
    (2)    :<----:-Establish Association-------:--->:    :    :        :
           :<----:-ASPUP-----------------------:----:    :    :        :
           :-----:-ASPUP ACK-------------------:--->:    :    :        :
           :     :                             :    :    :    :        :
    (3)    :<----:-Establish Association-------:----:--->:    :        :
           :<----:-ASPUP-----------------------:----:----:    :        :
           :-----:-ASPUP ACK-------------------:----:--->:    :        :
           :     :                             :    :    :    :        :
    (4)    :<----:-Establish Association-------:----:----:--->:        :
           :<----:-ASPUP-----------------------:----:----:----:        :
           :-----:-ASPUP ACK-------------------:----:----:--->:        :
           :     :                             :    :    :    :        :
           :     : (Same message exchange for SGP2) :    :    :        :
           :     :                             :    :    :    :        :
```

Figure 12. Example – AS-PENDING Recovery

The sequence of events in the exmaple illustrated in *Figure 12* is as follows:

(1)

(2)

(3)

(4)

(5)

(6)

(7)

5.6. Interworking

These are examples of interworking between nodes not supporting *CORID* with nodes supporting *CORID*.

5.6.1. ASP does not Support CORID

Figure 13 illustrates an example where the ASP does not support *CORID*, but the SGP does.

```
     SGP1  SGP2                          ASP1 ASP2 ASP3 ASP4     AS1
      :     :                             :    :    :    :        :
(1)   :<----:-Establish Association------>:    :    :    :        :
      :<----:-ASPUP-----------------------:    :    :    :        :
      :-----:-ASPUP ACK------------------>:    :    :    :        :
      :     :                             :    :    :    :        :
(2)   :<----:-Establish Association-------:--->:    :    :        :
      :<----:-ASPUP-----------------------:----:    :    :        :
      :-----:-ASPUP ACK-------------------:--->:    :    :        :
      :     :                             :    :    :    :        :
(3)   :<----:-Establish Association-------:----:--->:    :        :
      :<----:-ASPUP-----------------------:----:----:    :        :
      :-----:-ASPUP ACK-------------------:----:--->:    :        :
      :     :                             :    :    :    :        :
(4)   :<----:-Establish Association-------:----:----:--->:        :
      :<----:-ASPUP-----------------------:----:----:----:        :
      :-----:-ASPUP ACK-------------------:----:----:--->:        :
      :     :                             :    :    :    :        :
      :     : (Same message exchange for SGP2) :    :    :        :
      :     :                             :    :    :    :        :
```

Figure 13. Example – Interworking

The sequence of events in the exmaple illustrated in *Figure 13* is as follows:

(1)

(2)

(3)

(4)

(5)

(6)

(7)

6. Security

CORID does not introduce any new security risks or considerations that are not already inherent in the UA [M2UA...TUA] Please see the "Security" sections of M2UA, M3UA, SUA and TUA [M2UA...TUA] for security considerations and recommendations that are applicable to each of these UAs.

7. IANA Considerations

CORID redefines the format of the *Correlation Id* parameter for M2UA, M3UA, SUA and TUA. *CORID* also redifines the *ASPAC* and *ASPAC ACK* messages to include the *Correlation Id* parameter as a mandatory parameter of those messages.

8. Timers

Following are the **RECOMMENDED** timer values:

T(divert)	0.5-2 seconds
T(restore)	0.5-2 seconds
T(lifetime)	implementation dependent

Acknowledgments

The authors would like to thank Tolga Asveren, Ken Morneault, Greg Sidebottom, John Loughney, Sandeep Mahajan, Barry Nagelberg, and Nitin Vairagare for their valuable comments and suggestions.

Notes

[1] As described in the UA documents [M2UA...TUA].

[2] For illustration purposes only, all ASPs in *Figure 2* are members of the one Application Server which is represented at all of the SGPs.

[3] See Section 4.3.4.3 of the specific UA document [M2UA...TUA].

[4] See, for example, Clause 5 "Changeover", Clause 6 "Changeback", Clause 7 "Forced Rerouting" and Clause 8 "Controlled Rereouting" of the MTP3 specifications [Q.704].

[5] This is true for all User Adaptation layers with the exception of M2UA [M2UA]. In M2UA, the Application Server and traffic flows are identified by an equivalent of the *Routing Key*: the *Link Key*, and the equivalent of the *Routing Context*: the *Interface Identifier*. An Application Server may also represent multiple *Interface Identifiers*.

[6] That is, the *Traffic Flow Id* will be assigned by the SPP **sending** the message. *Traffic Flow Ids* are only used to determine whether messages belong to the same traffic flow, therefore, the *Traffic Flow Id* need only uniquely identify a traffic flow within an Application Server at the sending SPP.

[7] **IMPLEMENTATION NOTE:–** A simple way to assign the *Traffic Flow Id* when performing Load Selection [LOADSEL] is to simply assign the same value to the *Traffic Flow Id* as is assigned to the *Load Selector*.

[8] **IMPLEMENTATION NOTE:–** A simple way to assign the *Traffic Flow Id* when performing Load Grouping [LOADGRP] is to simply assign the same value to the *Traffic Flow Id* as is assigned to the *Load Selector* or *Load Group Identifier*.

[9] **IMPLEMENTATION NOTE:–** A simple way to meet the requirements for keeping local copies of messages is to keep a local copy of all messages sent to an SPP supporting *CORID* until a fixed buffer allocation is exceeded, or until the local copy lifetime expires. T(lifetime) and buffer capacity can then be adjusted to ensure that local copies of messages are not discarded too early resulting in message loss during fail-over.

[10] **IMPLEMENTATION NOTE:–** Determining which messages have already been processed for the AS may require some ASP-to-ASP or SGP-to-SGP synchronization that is outside the scope of the UA documents [M2UA...TUA] and also outside the scope of this document.

If the received traffic flow id matches that of the SPP on which the message was received, this might be a simple matter of comparing the correlation number of the message to the *Correlation Number* of the last message processed for the Application Server.

[11] **IMPLEMENTATION NOTE:–** The reason for discarding tagged messages at the receiver for which it cannot be determined with any certainty whether the message was processed for the AS or not is because, for SS7, message loss is preferrable to message duplication [Q.706].

[12] **IMPLEMENTATION NOTE:–** When restarting traffic with the contents of the diversion buffer, it might be necessary to reassign *Routing Context* (*Interface Identifier*) values within the messages if the *Routing Context* (*Interface Identifier*) values were assigned before buffering, and if the *Routing Context* (*Interface Identifier*) values associated with the AS traffic for the alternate SPP are different than the *Routing Context* (*Interface Identifier*) values associated with the same AS traffic for the failed SPP.

[13] **IMPLEMENTATION NOTE:–** Although the unique identifier placed in the *Heartbeat Data* is implementation dependent, a useful identifier would be the tuple formed by the *Routing Context* (*Interface Identifier*), *Correlation Id* corresponding to the last message(s) sent to the SPP from which the included traffic flows are to be diverted.

[14] For the "ASP Inactive Procedures", see Section 4.3.4.4 of the specific UA document [M2UA...TUA].

[15] For the "ASP Active Procedures", see Section 4.3.4.3 of the specific UA document [M2UA...TUA].

References

RFC 2960.
R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. J. Schwarzbauer, T. Taylor, I. Rytina, H. Kalla, L. Zhang and V. Paxson, “Stream Control Transmission Protocol (SCTP),” RFC 2960, The Internet Society (February 2000). [Normative]

Q.700. ITU, “Introduction to CCITT Signalling System No. 7,” ITU-T Recommendation Q.700, ITU-T Telecommunication Standardization Sector of ITU, Geneva (March 1993). [Informative]

M2UA. K. Morneault, R. Dantu, G. Sidebottom, B. Bidulock and J. Heitz, “Signaling System 7 (SS7) Message Transfer Part 2 (MTP2) - User Adaptation Layer,” RFC 3331, Internet Engineering Task Force - Signalling Transport Working Group (September, 2002). [Normative]

M3UA. G. Sidebottom, K. Morneault and J. Pastor-Balbas, (eds), “Signaling System 7 (SS7) Message Transfer Part 3 (MTP3) - User Adaptation Layer (M3UA),” RFC 3332, Internet Engineering Task Force - Signalling Transport Working Group (September, 2002). [Normative]

SUA. J. Loughney, G. Sidebottom, L. Coene, G. Verwimp, J. Keller and B. Bidulock, “SS7 SCCP-User Adaptation Layer (SUA),” <draft-ietf-sigtran-sua-14.txt>, Internet Engineering Task Force - Signalling Transport Working Group (June 30, 2002). Work In Progress. [Normative]

ISUA. B. Bidulock, “SS7 ISUP-User Adaptation Layer (ISUA),” <draft-bidulock-sigtran-isua-00.txt>, Internet Engineering Task Force - Signalling Transport Working Group (January 5, 2003). Work In Progress. [Normative]

TUA. B. Bidulock, “SS7 TCAP-User Adaptation Layer (TUA),” <draft-bidulock-sigtran-tua-01.txt>, Internet Engineering Task Force - Signalling Transport Working Group (January 2, 2003). Work In Progress. [Informative]

Q.704. ITU, “Message Transfer Part – Signalling Network Functions and Messages,” ITU-T Recommendation Q.704, ITU-T Telecommunication Standardization Sector of ITU, Geneva (March 1993). [Informative]

LOADSEL.
B. Bidulock, “Load Selection Extension for Signalling User Adaptation Layers (LOADSEL),” <draft-bidulock-sigtran-loadsel-01.txt>, Internet Engineering Task Force - Signalling Transport Working Group (January 2, 2003). [Informative]

LOADGRP.
B. Bidulock, “Load Grouping Extension for Signalling User Adaptation Layers (LOADGRP),” <draft-bidulock-sigtran-loadgrp-01.txt>, Internet Engineering Task Force - Signalling Transport Working Group (January 2, 2003). [Informative]

Q.706. ITU, “Signalling System No. 7 – Message Transfer Part Signalling Performance,” ITU Recommendation Q.706, ITU-T Telecommunication Standardization Sector of ITU, Geneva (March 1993). [Informative]

Q.705. ITU, “Signalling System No. 7 – Signalling Network Structure,” ITU-T Recommendation Q.705, ITU-T Telecommunication Standardization Sector of ITU, Geneva (March 1993). [Informative]

RFC 2119.
S. Bradner, “Key words for use in RFCs to Indicate Requirement Levels,” RFC 2119 - BCP 14, Internet Engineering Task Force (March 1997). [Normative]

ASPEXT.
B. Bidulock, “Application Server Process (ASP) Extension Framework,” <draft-bidulock-sigtran-aspext-01.txt>, Internet Engineering Task Force - Signalling Transport Working Group (January 2, 2003). Work In Progress. [Normative]

Author’s Addresses

Brian Bidulock
OpenSS7 Corporation
1469 Jeffreys Crescent
Edmonton, AB T6L 6T1
Canada

Phone: +1-780-490-1141
Email: bidulock@openss7.org
URL: http//www.openss7.org/

This draft expires July, 2003.

List of Tables

List of Illustrations

Table of Contents

Copyright Statement