                   Ascend's Multilink Protocol Plus (MP+)

Status of This Memo

   This memo provides information for the Internet community. This memo
   does not specify an Internet standard of any kind. Distribution of
   this memo is unlimited.

Abstract

   This document proposes an extension to the PPP Multilink Protocol
   (MP) [1]. Multilink Protocol Plus (MP+) is a new control protocol for
   managing multiple data links that are bundled by MP.

Table Of Contents

1. Introduction

   The PPP Multilink Protocol (MP), is a set of features that provide
   inverse multiplexing at the packet/fragment level by bundling
   multiple independent links between a fixed pair of systems, providing
   a virtual link with greater bandwidth than any of the constituent
   members.

   Once multiple channels have been established MP is responsible for
   managing channel use to insure in-sequence delivery of user packets.

   MP+ is an extension to MP that adds an inband control channel to
   provide a new level of session management and control.

   MP+ also allows remote device management of (unconfigured) systems.
   This feature allows a network operations center to dial into an
   unconfigured system and remotely manage it, before ethernet
   interface, IP address, and other LCP and system configuration
   information is entered. (This does require local configuration of
   the WAN interfaces to the extent required to answer an incoming
   call).

1.1 Functional Description

   The features of MP+ include:

   * Ability to negotiate to add and subtract channels when bandwidth
     needs change.

   * Phone number management so calling stations need not know every
     possible number; answering stations can manage their own resources.

   * A simple remote management interface.

To perform the above functions MP+ is split into a call management
layer and a reliable delivery layer. The call management layer is
the source and sink of MP+ control messages. The reliable delivery
layer adds a simple acknowledge and retry mechanism.

MP+ only takes network bandwidth when in the process of performing a
user request, e.g. adding and subtracting bandwidth.

NOTE: Neither MP, or MP+ define the process that makes the bandwidth
requirement determination. That is outside the scope of either of
these protocols and will likely be implementation dependent.

1.2 Conventions

The following language conventions are used in the items of
specification in this document:

MUST, SHALL or MANDATORY -- the item is an absolute requirement
of the specification.

SHOULD or RECOMMENDED --   the item should generally be followed
for all but exceptional circumstances.

MAY or OPTIONAL -- the item is truly optional and may be followed
or ignored according to the needs of the implementor.

2. General Overview

PPP
   In order to establish communications over a point-to-point link,
   each end of the PPP [2] link must first send LCP packets to
   configure the data link during link establishment phase. After
   the link has been established, PPP provides for an authentication
   phase.

MP The goal of multilink operation is to bundle multiple
   independent links between a fixed pair of systems, providing a
   virtual link with greater bandwidth than any of the constituent
   members.

MP+ MP+ is also negotiated during initial LCP option negotiation. A
   system indicates to its peer that it is willing to do MP+ by
   sending the MP+ option as part of the initial LCP option
   negotiation. The MP+ option MUST NOT be negotiated unless MP is
   also negotiated. When used, MP+ adds a virtual unit-to-unit
   control channel.

A peer may elect to:

Acknowledge both the MP and MP+ options, indicating that both MP and
MP+ will be used.

Acknowledge the MP option and reject the MP+ option. Operation will
fall back to MP.

Reject both options. Standard PPP will be used for this connection.

2.1. Operation

   Standard PPP
      In standard PPP the LCP negotiation phase is followed by an
      optional authentication phase, and then one or more NCPs are
      initiated.

   PPP with MP The LCP negotiation phase and authentication phase are
      identical to standard PPP. The ability to initiate an MP
      aggregate data link is indicated by sending an MP option - as
      described in [1].

   PPP with MP and MP+ When MP+ is negotiated at LCP startup, the same
      procedures are followed as when MP is negotiated alone. The MP+
      LCP option is negotiated to indicate the ability to use the MP+
      feature.The first connection between endpoints causes the MP+
      process to be started for the connection.

3. MP+ Frame Formats

```
                    +--------------+--------------+
      PPP Header:   | Address 0xff | Control 0x03 |
                    +--------------+--------------+
                    | PID(H)  0x00 | PID(L)  0x73 |
                    +-+-+-+-+------+--------------+
      MP Header:    |1|1|0|0|0|0|0|1| seq # high  |
                    +-+-+-+-+------+--------------+
                    |   sequence number low bits  |
                    +--------------+--------------+
                    |         control data        |
                    |              .              |
                    |              .              |
                    |              .              |
                    +--------------+--------------+
      PPP FCS:      |             FCS             |
                    +--------------+--------------+
```
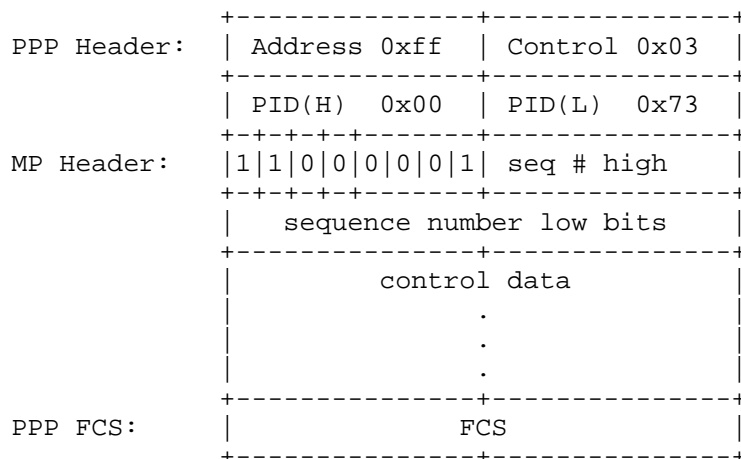
   Figure 1: Multilink Plus Frame Format (long sequence number format)

```
                      +---------------+---------------+
         PPP Header:  | Address 0xff  | Control 0x03  |
                      +---------------+---------------+
                      | PID(H)  0x00  | PID(L)  0x73  |
                      +-+-+-+-+-------+---------------+
         MP Header:   |1|1|0|1| sequence number      |
                      +-+-+-+-+-------+---------------+
                      |          control data         |
                      |               .               |
                      |               .               |
                      |               .               |
                      +---------------+---------------+
         PPP FCS:     |              FCS              |
                      +---------------+---------------+
```
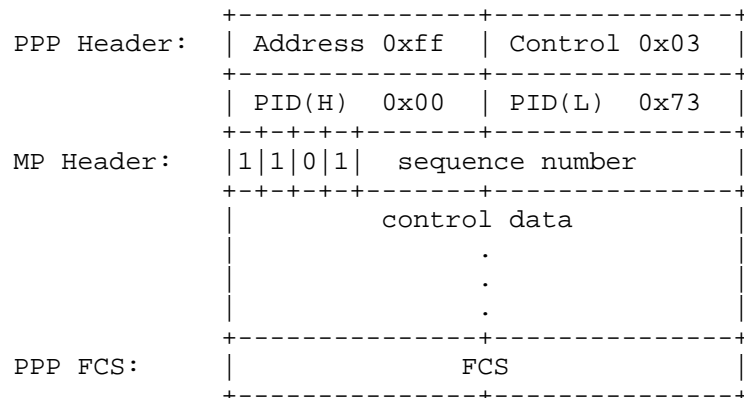
   Figure 2: Multilink Plus Frame Format (short sequence number format)

   MP+ frames use a similar structure to MP fragments.

   The MP+ assigned PID is designated 00 73.

   MP+ control uses the following two rules:

   - MP+ control frames have their own sequence number space,
     controlled by MP+.

   - MP+ control frames MUST NOT be fragmented.

NOTE: Implementations of this protocol prior to the date of submission
      of this specification to the IETF use the same PID as MP, but
      sets the LSB of the reserved bits in the MP header to 1 - this
      is how the MP+ packets are discriminated from MP fragments.
      So the header of the MP+ packet looks like:

            00 3d c1 ......

      As compared to an MP packet that looks like:

            00 3d c0 ...... or
            00 3d 80 ...... or
            00 3d 40 ......

3.1. Error Control (EC) Layer (MP+ control only)

   The error control layer that runs over the virtual inband channel is
   as simple as it can get, while handling the possibility of errors on
   the line.

   An assumption is made that errors are infrequent, and that at the
   same time messages are rarely, if ever, dropped on the floor. The
   implication of this is that "timing out" on retransmission of
   messages does no harm. If a message cannot get through, then it
   simply is retried some number of times. After giving up, the only
   recourse is to notify the call management layer (of MP) that the
   session has died.

```
                    +---------------+---------------+
    PPP Header:     | Address 0xff  | Control 0x03  |
                    +---------------+---------------+
                    | PID(H)  0x00  | PID(L)  0x73  |
                    +-+-+-+-+-------+---------------+
    MP+ Header:     |1|1|0|0|0|0|0|1| seq # high    |
                    +-+-+-+-+-------+---------------+
                    |    sequence number low bits   |
                    +---------------+---------------+
    EC Header:      | Error Control Message Type    |
                    |    32 bits reserved           |
                    +---------------+---------------+
    MP+ Data:       |    MP+ Message                |  May not be
                    |                               |  present.
```
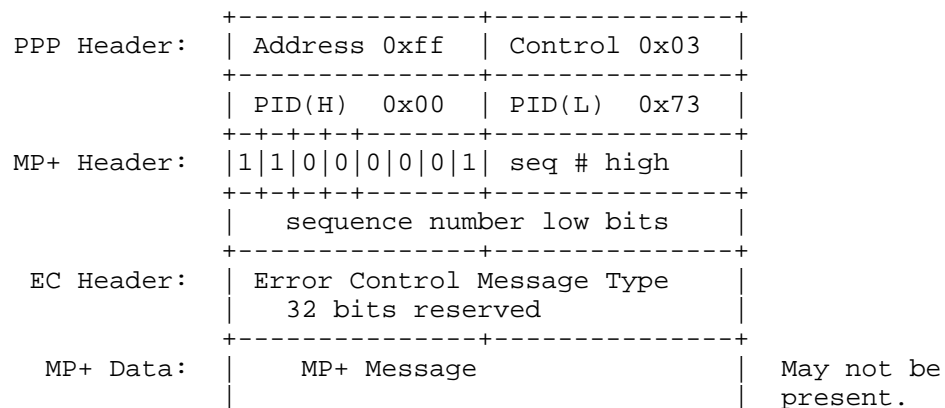
   Figure 3: MP+ control message format (shown long sequence number
             format)

  Error Control Message Type:

   1   DATA_MSG: This message contains MP+ data transferred
                            between the peers.

   2   ACK_MSG: An acknowledgement of a previous data message.

   When set to DATA_MSG, the remainder of the frame contains an MP+
   Control message.

   When set to ACK_MSG, the remainder of the frame consists only of the
   PPP Frame Check Sum (FCS).

3.1.1. Error Control State Machine

   This layer is controlled by a simple state machine. There are three
   states:

   Stopped   There is no connection between peers.

   Idle   There is a connection between peers;
                   no unacknowledged messages pending.

   Pending   There is a connection between peers;
                   awaiting an acknowledgement to the
                   last message sent.

   Messages from the call management layer are queued for transmission
   whenever the link is in the pending state. For simplicity, only
   one outstanding message may be in the link at any given time. The
   entire procedure is defined in table 1.

Event	State		
	Stopped	Idle	Pending
Start	1,Idle	-,*	-,*
Received ACK_MSG current tx sequence number	**	2,Start	5,Idle|Pending
Received ACK_MSG last tx sequence number	**	-,*	-,*
Received ACK_MSG other tx sequence number	**	2,Start	2,Start
Received DATA_MSG current rx sequence number	**	6,*	6,*
Received DATA_MSG previous rx sequence number	**	7,*	7,*
Received DATA_MSG other rx sequence number	**	2,Start	2,Start
Receive Invalid Frame	**	2,Start	2,Start
Retransmit Timer Expire	**	**	4,Start|*

_____

| Transmit Request from call | -,* | 3,Pending | 8,* |
| management layer | | | |

_____

| Stop | 9,Start | 9,Start | 9,Start |

_____

Table 1: Error Control State Machine

Legend:

     -        No action

     *   Stay in same state

     **   Invalid or meaningless event for state, ignored.

Notes:

   [1] Data from the call management layer will always be copied before
       being queued for transmission. The call management layer is
       responsible for its own buffers.

   [2] MP always copies data for transmission and returns immediately.
       Any buffers allocated to build control messages MUST be released
       immediately upon return from MP transmission requests.

Actions:

    1   Reset rx sequence number
           Reset tx sequence number
           Reset tx retransmit count
           Stop retransmit timer

    2   Report error to user
           Stop retransmit timer
           Stop frame transmit timer
           Free buffers

    3   Save call management message in pending transmit queue
           Build DATA_MSG from first message in pending transmit
           queue using current tx sequence number.
           Send message to MP for transmission.
           Reset tx retransmit count

```
 4        Increment tx retransmit count
          If tx retransmit count >= RETRANSMIT_COUNT
                  Action 2 (followed by state change to the Start state)
          else
                  Build DATA_MSG from first message in pending
                  transmit queue using current tx sequence number.
                  Send message to MP for transmission.

 5        Dequeue first element on pending transmit queue and release
          its buffer
          Increment the tx sequence number
          Stop the retransmit timer
          if pending transmit queue not empty
                  Build DATA_MSG from first message in pending
                  transmit queue using current tx sequence number.
                  Send message to MP for transmission.
                  Reset tx retransmit count

 6        Build ACK_MSG using the current rx sequence number
          Send ack message to MP for transmission
          Pass message to call managment layer
          Increment rx sequence number

 7        Build ACK_MSG using the previous rx sequence number
          Send the ack message to MP for transmission

 8        Add the message to the end of the pending transmit queue

 9        Stop retransmit timer
          Free buffers
```

3.2. Multilink Plus Control Messages

```
   Message Type                  Value
   VERSION_EXCHANGE_REQ          1
   VERSION_EXCHANGE_RSP          2
   ADD_REQ                       3
   ADD_RSP                       4
   ADD_COMPLETE                  5
   REMOVE_REQ                    6
   REMOVE_RSP                    7
   REMOVE_COMPLETE               8
   CLOSE_REQ                     9
   CLOSE_RSP                     10
   REMOTE_MGMT_REQ               11
   REMOTE_MGMT_RSP               12
   REMOTE_MGMT_RX_REQ            13
   REMOTE_MGMT_TX_REQ            14
```

```
        REMOTE_MGMT_TX_RSP              15
        CLEAR_REQ                       16
```

3.3. Multilink Plus Message Formats

   The fields of all messages defined here MUST be encoded/decoded in
   Network Byte Order (big endian).

3.3.1. VERSION_EXCHANGE_REQ Message Format

   The version exchange message is sent by the call originator to inform
   the answerer the version of the MP+ protocol being used as well as
   any other information that may need to be conveyed outside of the
   normal PPP parameter negotiation.

```
        +---------------+---------------+
        |          Message type         |
        |          0x00000001           |
        +---------------+---------------+
        |       Protocol Version        |
        +---------------+---------------+
        |       Protocol Revision       |
        +---------------+---------------+
        |       Session Identifier      |
        +---------------+---------------+
        |         Hardware Type         |
        +---------------+---------------+
        |          Nailed Mode          |
        +---------------+---------------+
        |   Use Multiple Trunk Groups   |
        +---------------+---------------+
        |       Descriptor Length       |
        +---------------+---------------+
        |          Descriptor           |
        +---------------+---------------+
```

   Figure 4: Version Exchange Request

   A message sent from call originator to call answerer specifying the
   callers protocol version and other state info and requesting the
   answerer to respond with its version info.

   Protocol Version:
           caller MP+ protocol version number.
           2 octets fixed length (initially 1)

     Protocol Revision:
             caller MP+ protocol revision number.
             2 octets fixed length (initially 4)

     Session Identifier:
             A non-zero identifier unique to the caller.
             2 octets fixed length.

     Hardware Type:
             caller hardware type (can be vendor defined).
             2 octets fixed length.

     Nailed Mode:
             caller nailed mode from the session profile.
             2 octets fixed length.

     Use Multiple Trunk Groups:
             non-zero if the call may use channels from multiple trunk
             groups.
             2 octets fixed length

     Descriptor Length:
             length of the end point descriptor.
             2 octets fixed length

     Descriptor:
             the end point descriptor. This field allows for vendor
             specific identification of the peer.
             Variable length as defined above.

3.3.2. VERSION_EXCHANGE_RSP Message Format

   The version exchange response message is sent by the call answerer in
   response to a version exchange request message.   The answerer uses
   the message to inform the caller the version of the MP+ protocol
   being used as well as any other information that needs to be conveyed
   outside of the normal PPP parameter negotiation.

```
      +---------------+---------------+
      |          Message type         |
      |          0x00000002           |
      +---------------+---------------+
      |        Protocol Version        |
      +---------------+---------------+
      |        Protocol Revision       |
      +---------------+---------------+
      |        Session Identifier       |
      +---------------+---------------+
      |          Hardware Type          |
      +---------------+---------------+
      |           Nailed Mode           |
      +---------------+---------------+
      |    Use Multiple Trunk Groups    |
      +---------------+---------------+
      |         Descriptor Length        |
      +---------------+---------------+
      |            Descriptor            |
      +---------------+---------------+
```

   Figure 5: Version Exchange Response

   A message sent from call answerer to the call originator specifying
   the answerers protocol version and other state info. Sent in
   response to receiving a version exchange request.

   Protocol Version:
           caller MP+ protocol version number.
           2 octets fixed length (initially 1)

   Protocol Revision:
           caller MP+ protocol revision number.
           2 octets fixed length (initially 4)

   Session Identifier:
           A non-zero identifier unique to the answerer.
           2 octets fixed length.

Hardware Type:
        caller hardware type (can be vendor defined).
        2 octets fixed length.

Nailed Mode:
        caller nailed mode from the session profile.
        2 octets fixed length.

Use Multiple Trunk Groups:
        non-zero if call may use channels from multiple trunk groups.
        2 octets fixed length

Descriptor Length:
        length of the remote descriptor in 4-octet units.
        2 octets fixed length

Descriptor:
        the remote unit descriptor. This field allows for vendor
        specific identification of the peer.
        Variable length Nx4 octets long - total length defined above.

3.3.3. ADD_REQ Message Format

A message of this type is sent by either caller or answerer to
initiate an increase of bandwidth. When sent by the caller the
request is asking for permission to dial a certain number of
channels; the response will contain permission and the phone numbers
of the channels to dial. When sent by the answerer, this message
contains the phone numbers to dial. The message looks like:

```
+--------------+--------------+
|        Message type         |
|        0x00000003           |
+--------------+--------------+
| Number of channels requested |
+--------------+--------------+
|    Number of phone numbers   |
+--------------+--------------+
|    A phone number list for   |
|      each phone number       |
|              .               |
|              .               |
|              .               |
+--------------+--------------+
```
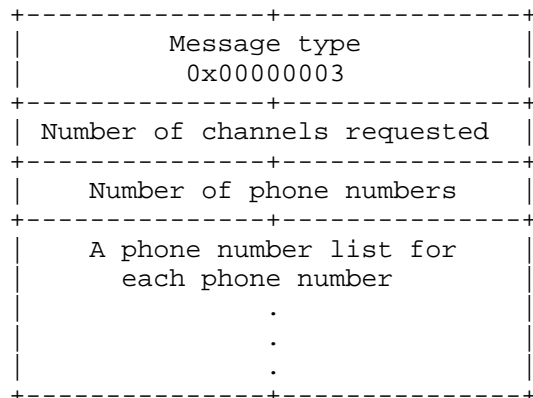
Figure 6: Add Request

A message sent by either caller or answerer to request that additional
bandwidth be added to a session.

Number of channels requested:
        The maximum number of channels to add.
        2 octets fixed length.

Number of phone numbers:
        The number of phone numbers provided. This
        value will always be zero when the caller
        initiates an add and will be at least
        Number of channels requested when the answerer
        initiates the add.
        2 octets fixed length.

Phone number list:
        A list of up to 32 phone number lists
        containing the phone numbers to call.
        Each description is of fixed length as described below:

        Each phone number is represented by a phone number list.
        The format of a phone number list is:

```
+--------------+--------------+
|           in use flag       |
+--------------+--------------+
|         call service type   |
+--------------+--------------+
|          Phone number       |
|            20 octets        |
|       plus null terminator  |
|         (21 octet total)    |
|                             |
|                             |
|                             |
|                             |
|                             |
|                             |
|              +--------------+
|              | Must be 0    |
+--------------+--------------+
|           must be 0         |
+--------------+--------------+
```
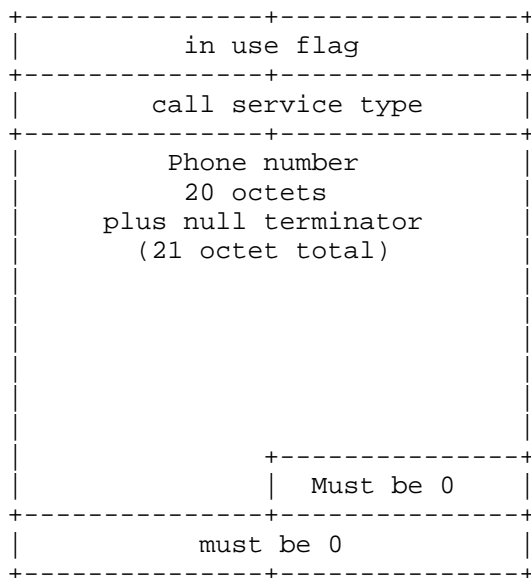
Figure 7: Phone number list

A structure containing information about a connection within the
system.

in use flag:
        non-zero if the phone number indicated
        in this descriptor is currently in use.
        2 octets fixed length

call service type:
        Defines the type of service, switched, nailed,
        or other, associated with a phone number.
        1   Nailed
        2   Switched
        >=3   Undefined

Phone number:
        The null terminated phone number of this channel.
        Fixed length 21 octets. Each octet contains IA5 character
        representation of a digit (or #, *).

Must be 0:
        Filler to force alignment to 32-bit boundary.

3.3.4 ADD_RSP Message Format

A message of this type gives permission to dial some number of
channels and, when sent by the answerer of the original call, gives
the phone numbers of channels to dial.

```
+--------------+--------------+
|         Message type        |
|         0x00000004          |
+--------------+--------------+
|  Number of channels allowed |
+--------------+--------------+
|    Number of phone numbers  |
+--------------+--------------+
|   A  phone number list for  |
|       each phone number     |
|              .              |
|              .              |
|              .              |
+--------------+--------------+
```
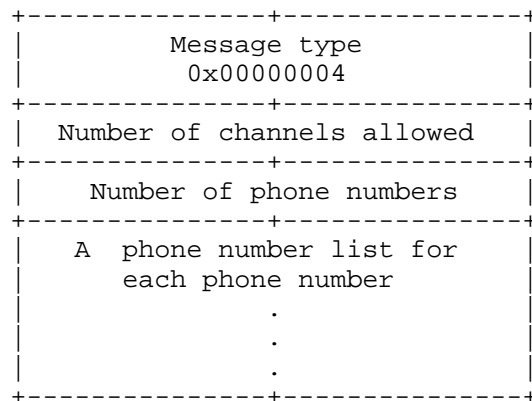
Figure 8: Add Response

A message sent by either caller or answerer to indicate the number of
channels that may be added to a session.

Number of channels allowed:
        The actual number of channels to add. This
        may be less than the number requested.
        2 octets fixed length.

Number of phone numbers:
        The number of phone numbers provided. This
        value will always be zero when sent by the
        caller and will be at least channelCount
        when sent by the answerer.
        2 octets fixed length.

Phone number list:
        A list of up to 32 phone number lists
        containing the phone numbers to call.
        Each description is of fixed length as described above.

3.3.5. ADD_COMPLETE Message Format

This message is sent by the caller to the answerer after all calls
have been placed. The message is used to notify the answerer that
the add transaction is complete and it may return to the idle state.

```
+---------------+---------------+
|          Message type         |
|           0x00000005          |
+---------------+---------------+
|         Channels added        |
+---------------+---------------+
|           Must be zero        |
+---------------+---------------+
```

Figure 9: Add Complete

A message sent by caller to indicate the number of channels that were
added successfully. This message was added in MP+ Version 1.1.

Channels added:
        The actual number of channels added.
        2 octets fixed length

Must be zero:
        Padding to 32-bit boundary.
        2 octets fixed length

3.3.6. REMOVE_REQ Message Format

   A message of this type is sent when a peer decides, for any reason,
   to remove channels from use. The purpose of the message is to tell
   the remote end of the remove and give it a chance to adjust the
   number of channels to remove.

```
   +---------------+---------------+
   |         Message type          |
   |         0x00000006            |
   +---------------+---------------+
   | Number of channels to remove  |
   +---------------+---------------+
   |          Must be zero         |
   +---------------+---------------+
```

   Figure 10: Remove Request

   A message sent by either caller or answerer to request that bandwidth
   be removed from a session.

   Number of channels to remove:
           The maximum number of channels to remove.
           2 octets fixed length

   Must be zero:
           Padding to 32-bit boundary.
           2 octets fixed length

3.3.7. REMOVE_RSP Message Format

   This message is sent in response to a remove request. The responder
   specifies the number of channels that can be removed. If the
   response specifies 0 channels the remove is cancelled.

```
   +---------------+---------------+
   |         Message type          |
   |         0x00000007            |
   +---------------+---------------+
   | Number of channels to remove  |
   +---------------+---------------+
   |          Must be zero         |
   +---------------+---------------+
```

   Figure 11: Remove Response

A message sent in response to a remove request specifying the number
of channels that the peer agrees can be removed.

Number of channels to remove:
        The maximum number of channels to remove.
        May be zero, in which case the remove is
        cancelled.
        2 octets fixed length

Must be zero:
        Padding to 32-bit boundary.
        2 octets fixed length

3.3.8. REMOVE_COMPLETE Message Format

This message is sent by the initiator of a remove transaction when
the agreed upon number of channels have been removed. The message is
used to notify the peer that the remove transaction is complete and
it may return to the idle state.

```
+---------------+---------------+
|          Message type         |
|           0x00000008          |
+---------------+---------------+
| Number of channels removed    |
+---------------+---------------+
|          Must be zero         |
+---------------+---------------+
```
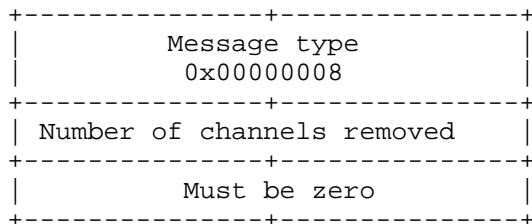
Figure 12: Remove Complete

A message sent by the caller or answerer to indicate how many channels
were actually removed. This message was added in MP+ CM version 1.1.

Number of channels removed:
        The number of channels that were removed.
        2 octets fixed length

Must be zero:
        Padding to 32-bit boundary.
        2 octets fixed length

3.3.9. CLOSE_REQ Message Format

   This message is sent when the peer requests to close the whole
   session. This is typically due to a configuration option that
   indicates when a system should request to close the session (an
   example being, a link has been idle for greater than a preconfigured
   time period).

```
   +---------------+---------------+
   |         Message type          |
   |          0x00000009           |
   +---------------+---------------+
```

   Figure 13: MP+ close request.

   There are no data fields associated with this message.

3.3.10. CLOSE_RSP Message Format

   If the peer agrees that closing the session is acceptable based on
   it's own configuration (an example reject reason would be that the
   peer is configured with a *minimum* number of channels to keep
   active).

```
   +---------------+---------------+
   |         Message type          |
   |          0x0000000a           |
   +---------------+---------------+
   |          OK To Close          |
   +---------------+---------------+
   |          Must be zero         |
   +---------------+---------------+
```

   Figure 14: MP+ close response

   The response to a close request. May be sent by caller or answerer.

   OK To Close:
           If non-zero, peer said I can close all channels.
           2 octets fixed length

   Must be zero:
           Padding to 32-bit boundary.
           2 octets fixed length

3.3.11. REMOTE_MGMT_REQ Message Format

   This message is sent from a master station to a slave station when
   the master wishes to manage the remote station. The message is also
   used to cancel remote management once it's been started.

```
       +---------------+---------------+
       |         Message type          |
       |          0x0000000b           |
       +---------------+---------------+
       |             Mode              |
       +---------------+---------------+
       |          Must be zero         |
       +---------------+---------------+
```
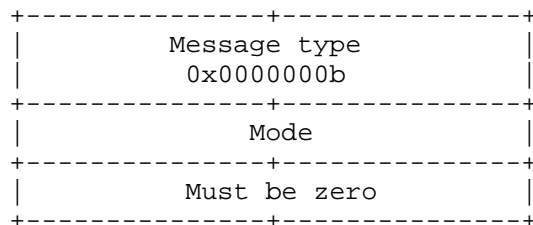
   Figure 15: Remote Management Request

   A message sent from master to slave to initiate or clear a remote
   management session.

   Mode:
           One to start session. Zero to stop session.
           2 octets fixed length

   Must be zero:
           Padding to 32-bit boundary.
           2 octets fixed length

3.3.12. REMOTE_MGMT_RSP Message Format

   The slave side of a remote management session has the opportunity to
   reject remote management. The master side is informed of accept/deny
   status via the remote management response.

```
       +---------------+---------------+
       |         Message type          |
       |          0x0000000c           |
       +---------------+---------------+
       |             Mode              |
       +---------------+---------------+
```
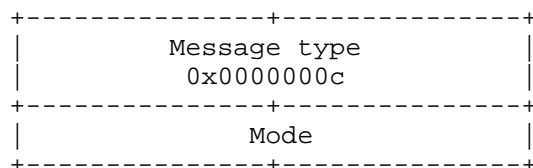
   Figure 16: Remote Management Response

   A message sent from slave to master to allow or deny initiation of a
   remote management session.

   Mode:
           One to accept session. Zero to deny session.
           2 octets fixed length

   Must be zero:
           Padding to 32-bit boundary.
           2 octets fixed length

3.3.13. REMOTE_MGMT_RX_REQ Message Format

   This message type is used to convey keyboard input from the
   management master to be processed by the management slave. The
   message format consists of an octet count (in network byte order) and
   then an array of octets to be processed. It looks like:

```
      +---------------+---------------+
      |          Message type         |
      |           0x0000000d          |
      +---------------+---------------+
      |         character count       |
      +---------------+---------------+
      |       array of characters     |
      |               .               |
      |               .               |
      |               .               |
      +---------------+---------------+
```
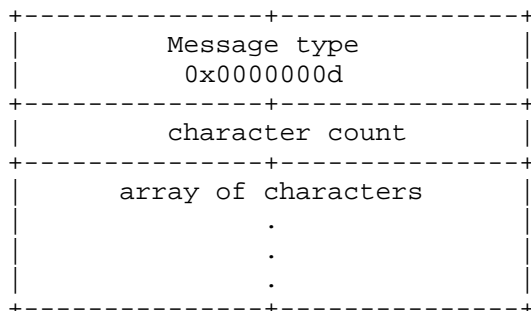
   Figure 17: Remote Management Receive Request

   A message sent from master to slave, conveying keystrokes typed on the
   masters keyboard that will be processed by the slave.

   character count:
           Number of characters to process.
           2 octets fixed length

   array of characters:
           Array of characters to process.

3.3.14. REMOTE_MGMT_TX_REQ Message Format

   The remote management slave conveys output to be displayed on the
   masters terminal with a remote management transmit request message.
   Only one message may be outstanding. The next transmit request may
   not be sent until the previous has been acknowledged.

```
       +---------------+---------------+
       |          Message type         |
       |          0x0000000e           |
       +---------------+---------------+
       |          character count      |
       +---------------+---------------+
       |       array of characters     |
       |               .               |
       |               .               |
       |               .               |
       +---------------+---------------+
```
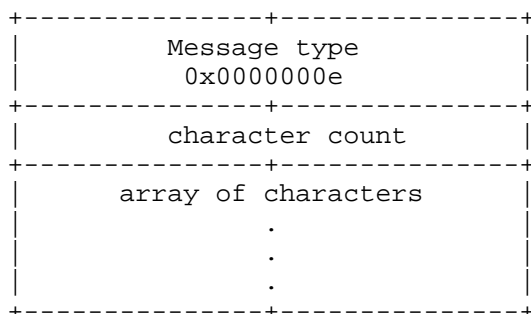
   Figure 18: Remote Management Transmit Request

   A message sent from slave to master, conveying output to be output on
   the master's display.

   Character count:
           Number of characters to process.
           2 octets fixed length

   array of characters:
           Array of characters to process.

3.3.15. REMOTE_MGMT_TX_RSP Message Format

   This message is used to acknowledge remote management transmit
   requests. The slave may send the next transmit request once this
   message has been received.

```
       +---------------+---------------+
       |          Message type         |
       |          0x0000000f           |
       +---------------+---------------+
```

   Figure 19: Remote Management Transmit Response

   There are no data fields associated with this message.

3.3.16. CLEAR_REQ Message Format

   A message sent to initiate a friendly shutdown of an MP+ link. The
   sender will stop sending data immediately. The receiver of the
   message will also stop sending user data and start a clean shutdown
   of all NCPs and the LCP of each member link of the bundle. When the
   last member link terminates, the session is completely closed.

```
+---------------+---------------+
|         Message type          |
|           0x00000010          |
+--------------+---------------+
```

   Figure 20: Clear Request

   There are no data fields associated with this message.

3.4. Events

   The MP+ state machine is event driven. Reception of an event triggers
   an action and possibly a state change. The events processed by the
   MP+ state machine can be roughly classed into two types:

      Events that originate within the unit, e.g. notification that a
      call has cleared, an MP+ session may be started, etc.

      Events that originate with the reception of an MP+ control
      message from the peer unit.

   Both types are processed by the state machine in the sequence they
   occurred. The events processed are:

   MP+START_SESSION: Notification from PPP/MP that an
                          MP+ session is starting.

   MP+SESSION_DOWN: Notification from the error-control
                          layer that end-to-end connectivity
                          has been lost and control messages can
                          not be delivered.

   MP+SESSION_TERM: Session termination notification from
                          PPP/MP. This event is not sent until
                          the last channel of a multi-channel
                          session is cleared.

   MP+TIMER_EXPIRED: Timers are used in various states and
                          sub-states. This event is signaled whenever
                          a timer expires.

    MP+CALL_COMPLETE: A call placed during an add request has
                         completed. The call may have succeeded or
                         failed.

    MP+UTILIZATION: Notification from MP/PPP that link
                         utilization has crossed a threshold and that
                         channels may need to be added/removed.
                         (The number of channels to add/remove will be
                          passed with the notification).

    MP+RX_VERSION_REQ: A Version Exchange request message has
                         been received from the peer.

    MP+RX_VERSION_RSP: A Version Exchange response message has
                         been received from the peer.

    MP+ADD_REQ: An Add request message has been received
                         from the peer.

    MP+ADD_RSP: An Add response message has been received
                         from the peer.

    MP+ADD_COMP: An Add Complete message has been received
                         from the peer.

    MP+REMOVE_REQ: A Remove request message has been received
                         from the peer.

    MP+REMOVE_RSP: A Remove response message has been
                         received from the peer.

    MP+REMOVE_COMP: A Remove Complete message has been received
                         from the peer.

    MP+RX_RM_REQ: A Remote Management request has been received
                         from the peer.

    MP+RX_RM_RSP: A Remote Management response has been received
                         from the peer.

    MP+RX_RM_RX_REQ: A Remote Management Receive Request has been
                         received from the far end.

    MP+RX_RM_TX_REQ: A Remote Management Transmit Request has
                         been received from the peer.

    MP+RX_RM_TX_RSP: A Remote Management Transmit Response has
                        been received from the peer.

    MP+RX_CLEAR: A request to shut down the session has been
                        received from the peer.

    MP+CLOSE_REQ: A Close Request message has been received
                        from the peer.

    MP+CLOSE_RSP: A Close Response message has been received
                        from the peer.

    MP+START_RM   Request to start a remote management session
                        with this station being the master.

    MP+SEND_RMS: Request to send data to a remote management
                        master from a slave.

    MP+SEND_RMM: Request to send data to a remote management
                        slave from a master.

    MP+RECV_RMM: Request to send an ack to a remote management
                        slave for data received from the slave.

    MP+STOP_RM: Request to stop a remote management session.

3.5. State Machine

3.5.1 States

    To ease readability and understanding the major states are considered
    as separate state machines, each having two to four sub-states. The
    sub-states are named by the letters, A, B, C, and D. State
    information is maintained for every interface.

    The major states are:

    MP+STATE_INITIAL: The state of an unused session. Session table
                      entries are intialized to this state at startup
                      and return to this state when sessions are
                      terminated.

    MP+STATE_IDLE: The state of an active session that is not
                      performing any MP+ function.

    MP+STATE_ADD: The state of a session when an add transaction is
                      in progress.

     MP+STATE_REMOVE: The state of a session when a remove transaction is
                        in progress.

     MP+STATE_CLOSE: The state of a session that is in the process of
                        being closed.

   State transitions are triggered by the reception of an event. Tables
   2 through 6 contain the state tables for the major states. All state
   tables use the following symbols.

        -          No action

        *   Stay in same state

        +   Target state is defined by the action taken

        **   An error has occurred, log an error message but no
                   state change.

   States and sub-state transitions are noted as state:sub-state, e.g.,
   initial:A. Alternative transitions are listed on separate lines.

3.5.2 Common Actions

   Some actions are common to all states, they are defined here.

Error Close Action

        Called to close a session when an error occurs. Actions are:

        [1] Stop timer if running.

        [2] Log an error message.

        [3] Close the MP+EC layer for this session.

        [4] Close MP for this session

        [5] Clean up, restore state variables to their initial state.

Term Action

        Processed when a MP+SESSION_TERM event occurs in most states.
        Actions are:

        [1] Stop timer if running.

        [2] Close the MP+EC layer for this session.

[3] Call the passed termination callback function if not null.

[4] Clean up, restore state variables to their initial state.

Ignore Action

We don't care about this event in this state. Do nothing.

Timer Action

This action is called when a timer expires in one of the on-line states. The timer is used to implement add and remove locks. A lock is set when an add or remove fails and is not cleared until a bandwidth change or the timer expires. This keeps us from retrying add's and subtracts until there is a likelyhood that it will succeed.

[1] Check add lock flag.

[1] If set an add lock occured last timeout period so triple the timeout value (to a max of 81 minutes).

[2] If not set restore the timeout value to its initial value of one minute.

[2] Clear the add lock flag.

[3] Clear the remove lock flag.

[4] Restart the retry timer.

Enter Remove [local] Action

The local unit is initiating a remove transaction. The desired bandwidth is given.

[1] Restart the idle timer.

[2] Calculate number of channels to remove (difference between number in use and number in desired).

[3] Build and send a remove request and send to remote.

[4] Go to REMOVE:A.

Enter Remove [remote] Action

        The remote unit is initiating a remove transaction. The incoming
        message contains the number of channels to remove.

        [1]   Restart the idle timer.

        [2]   Request the number of channels required. If greater than the
        number available after removing the number of channels indicated
        in the incoming message reduce the number of channels to remove
        and set a remove lock.

        [3] Build a remove response message indicating the number of
        channels we will allow the requester to remove and send to the
        remote.

        [4]   Go to REMOVE:B.

Enter Add [local] Action

        The local unit is initiating an add transaction. We are given
        the number of channels desired. The steps are:

        [1]   Restart the idle timer.

        [2]   Calculate number of channels to add (difference between
        number desired and number in use).

        [3]   Reserve number of channels, retrieving their phone numbers.

        [4]   If number of channels reserved less than the number desired
        set an add lock.

        [5]   If number of channels reserved greater than zero.

                [1] Build an add request. If the answerer the request
                includes the phone numbers for the caller to dial.

                [2] If caller, go to ADD:A.

                [3] If answerer, go to ADD:C.

        [6]   Go to IDLE state.

Enter Remove [remote] Action

      The remote unit is initiating a remove transaction. The
      incoming message contains the number of channels to remove.

      [1] Clear the remove lock.

      [2] Restart the idle timer.

      [3] Request the number of channels required. If greater
      than the number available after removing the number of
      channels indicated in the incoming message reduce the
      number of channels to remove.

      [4] Build a remove response message indicating the number
      of channels we will allow the requester to remove and
      send to the remote.

      [5] Go to REMOVE, sub-state B.

Enter Add [remote answerer] Action

      We've received a message from the remote requesting that
      bandwidth be added.   The message contains the number of
      channels to add. Since the remote is the answerer, the mes-
      sage also contains the phone nubmers to dial. We may dial
      less than the number requested.

      [1] Restart idle timer.

      [2] If the number of channels requested will put us over
      the maximum number of channels allowed for the session
      reduce the channel count.

      [3] For each channel to add,

          [1] Integrate the phone number returned from the
          answerer with the original phone number dialed.

          [2] Request that a session be extended by dialing
          the integrated phone number. A callback is
          passed with the request so call success or fail-
          ure can be reported back to MP+.

      [4] Go to ADD:B. Note: This change must actu-
      ally occur before requesting the first outgoing call.
      If not, the callback could be called (and ignored)
      because the session is not in the correct state.

Enter Add [remote caller] Action

        We've received a message from the remote requesting that
        bandwidth be added. The message contains the number of
        channels to add. Since the remote is the caller, it needs
        us to send the phone numbers to dial. We may send fewer
        phone numbers than requested

        [1] Restart idle timer.

        [2] If the number of channels requested will put us over
        the maximum number of channels allowed for the session
        reduce the channel count.

        [3] Reserve the adjusted number of channels, retrieving
        their phone numbers.

        [4] If the number of channels reserved is less than the
        adjusted number requested.

        [5] Build an add response message, including the phone
        numbers for the channels we will let the caller add
        and send it to the far end.

        [6] Go to ADD:C.

Enter Idle Action

        The IDLE state is entered at the end of normal transactions.
        At entry the current status of the connection should be
        checked and new transactions initiated if necessary. To be
        safe, we can also use this state as a catch all place to
        release any bandwidth reserved for adds. The functions to
        perform are:

        [1] Restart the idle timer using the current retry value.

        [2] Release any reserved bandwidth not actually in use.

        [3] Check if bandwidth change reqested during last trans-
             action. If change indicated:

             [1] Query channel counts.

             [2] If current bandwidth less than suggested band-
                  width and removes are not locked, store the
                  requested bandwidth and initiate a remove trans-
                  action (Enter Remove Action).

                    [3] If current bandwidth greater than suggested
                         bandwidth and adds are not locked:

                              [1] Store the requested bandwidth.

                              [2] Intiate an add transaction (Enter Add
                                   [local] Action).

          [4] Go to the IDLE state.

Remote Management Request Action

          We received a request to start/stop remote management.
          If this is a start request
                    If we can/allow remote management:
                              Build and send a Remote management response Allow
                              message.
                    Else
                              Build and send a Remote management response Deny
                              message.
          Else (this is a stop)
                    Notify the remote management slave process to terminate.

Remote Management Response Action

          We received a response to our remote management start request.
          If the response was an Allow response
                    Notify the remote management master process, we can
                    start sending keystrokes/commands
          Else
                    The peer denied the request, so notify the remote
                    management master process of failure.

Remote Management Receive Data Action

          We (the slave) received data from the remote management master.
          Pass the received data to the remote management slave process.
          This is typically keystroke data received from the remote user
          interface.

Remote Management Transmit Data Action

          We (the master) received data from the remote management slave.
          Pass the received data to the remote management master process.
          This is typically screen-updates to be passed to the user
          interface.

Remote Management Transmit Data Response Action

        We (the slave) received an ack to data we previously sent to the
        master. Notify the remote management slave process so that it
        can queue further transmissions.

Remote management (Master) start Action

        Build a REMOTE_MGMT_REQ start message and send to the far end.
        Send a proceeding message to the RM master process.

Remote management (Slave) data Action

        Build a REMOTE_MGMT_TX_REQ message with the data passed from
        the remote management slave process, send it to the far end.

Remote management (Master) data Action

        Build a REMOTE_MGMT_RX_REQ message with data passed from the
        remote management master process, send it to the far end.

Remote management data acknowledgement Action

        Build a REMOTE_MGMT_TX_RSP message and send it so the slave can
        send the next block. There is no data associated with this
        message.

Remote management (Master) stop Action

        Build a REMOTE_MGMT_REQ stop message and send to the far end.

3.5.3. MP+STATE_INITIAL state machine

   All sessions start from this state, sub-state A. The state is not
   exited until version exchange succeeds.

   The sub-states are:

        A   Initial state
        B   Sent version request, waiting for version response.
        C   Waiting for version request.

Event	Sub-state		
	A	B	C
MP+START_SESSION	1,+	**	**
MP+SESSION_DOWN	**	2,Initial:A	2,Initial:A
MP+SESSION_TERM	**	3,Initial:A	3,Initial:A
MP+TIMER_EXPIRED	**	4,Initial:A	7,Initial:B
MP+RX_VERSION_REQ	**	8,Initial:A	5,+
MP+RX_VERSION_RSP	**	6,+	**
MP+START_RM	9,*	9,*	9,*
All other events	**	**	**

Table 2: Initial State Machine

Actions:

1   Start timer, 60 seconds if originator, 30 seconds if answerer.

        Start MP+

        If originator

                Build and send version exchange request

                Go to INITIAL, sub-state B.

        Go to INITIAL, sub-state C.

2   Do Error Close Action, go to INITIAL, sub-state A.

3   Do Term Action, go to INITIAL, sub-state A.

4   Do Error Close Action, go to INITIAL, sub-state A.

5   Stop the idle timer.

        Compare protocol versions, if they do not match Do Error Close
        Action, go to INITIAL, sub-state A.

Store off info received from remote.

Build a version exchange response and send to remote end.

Do Enter Idle Action which causes a state change.

6   Stop the retry timer.

Compare protocol versions, if they do not match Do Error Close
Action, go to INITIAL, sub-state A.

Store off info received from remote.

Check the base channel count in the callers profile.

If greater than 1

          Set the requested bandwidth to the base channel count.

          Do Enter Add Caller action which causes a state change.

Do Enter Idle Action which causes a state change.

7   Both sides think they are the answerer. This is possible if both
          dial each other at the same time and the first channel that
          completed PPP negotiation happened to be the channel associated
          with the incoming call on both units. We resolve this by trying
          to become the originator.
          If both sides try to become the originator the one with the
          largest endpoint discriminator will fall back to being the
          answerer.

               Restart Idle timer at 60 seconds

               Build and send Version Exchange Request message

               Go to Initial:B

8   Both sides think they are the originator. This can happen if
          both dial each other at the same time and the first channel
          that completed PPP negotiation happened to be the channel
          associated with the originating call on both units. MP+
          determines which will be the caller and which the answerer by
          comparing the endpoind discriminator in the version exchange
          request with the local endpoint discriminator. The unit with
          the smaller endpoint is arbitrarily called the originator. The
          actions are:

        Compare local endpoint discriminator with endpoint discrimator
        in message.

        If local endpoint discriminator is less than the remote value
        we are the caller, ignore the incoming message.

        Otherwise, if local endpoint discriminator is greater than the
        remote value we are the answerer:

                Compare protocol versions, if they do not match
                Do Error Close Action, go to INITIAL, sub-state A.

                Store off info received from remote.

                Build a version exchange response and send to remote end.

                Do Enter Idle Action which causes a state change.

        If the two values match, there is a problem, Do Error Close
        Action,go to INITIAL, sub-state A.

9   Log an error message.
        Notify the user interface of remote management failure.

3.5.4. MP+STATE_IDLE state machine

   The Idle state is the state of an active session with no call
   management activity in progress.

   There are no sub-states.

Event	State
	A
MP+SESSION_DOWN	1,Initial:A
MP+SESSION_TERM	2,Initial:A
MP+TIMER_EXPIRED	3,*
MP+UTILIZATION	4,+
MP+RX_ADD_REQ	5,+
MP+RX_REMOVE_REQ	6,Remove:B
MP+RX_RM_REQ	7,+

```
_____
MP+RX_RM_RSP             8,+
_____
MP+RX_RM_RX_REQ          9,+
_____
MP+RX_RM_TX_REQ          10,+
_____
MP+RX_RM_TX_RSP          11,+
_____
MP+RX_CLOSE_REQ          12,+
_____
MP+START_RM              13,*
_____
MP+SEND_RMS              14,*
_____
MP+SEND_RMM              15,*
_____
MP+RECV_RMM              16,*
_____
MP+STOP_RM               17,*
_____
All other events         **
_____
```

Table 3: Idle State Machine

Actions:

1   Do Error Close Action, go to INITIAL, sub-state A.

2   Do Term Action, go to INITIAL, sub-state A.

3   Do Timer Action.

4   Note that a bandwidth change has been reqested.

        Do Enter Idle Action which may cause a state change.

5   If we are the caller:
                Do Enter Add [remote answerer] Action.
        Else
                Do Enter Add [remote caller] Action.

6   Do Enter Remove [remote] Action

7   Do Remote Management Request Action

8   Do Remote Management Response Action

    9   Do Remote Management Receive Data Action

    10   Do Remote Management Transmit Data Action

    11   Do Remote Management Transmit Data Response Action

    12   Clear remove lock.
              If local recommended channels == 0, then:
                      send a Close Response message with OK To Close
                      set to TRUE.
              Else
                      send a Close Response message with OK To Close
                      set to FALSE.
              Do Enter Idle Action.

    13   Do Remote management (Master) start Action

    14   Do Remote management (Slave) data Action

    15   Do Remote management (Master) data Action

    16   Do Remote management data acknowledgement Action

    17   Do Remote management (Master) stop Action

3.5.5. MP+STATE_ADD state machine

    The add state is used by both caller and answerer when an add
    transaction is in progress.

    The sub-states are:

        A   Add request sent to answerer, waiting for add response
                 from the answerer.
        B   Caller waiting for call complete notification for calls
                 placed.
        C   Answerer waiting for add complete message from caller.

Event	Sub-state		
	A	B	C
MP+SESSION_DOWN	1,Initial:A	7,Closing:A	1,Initial:A
MP+SESSION_TERM	2,Initial:A	7,Closing:B	2,Initial:A
MP+TIMER_EXPIRED	3,+	3,+	3,+
MP+UTILIZATION	4,*	4,*	4,*
MP+CALL_COMPLETE	**	8,Idle:A	**
MP+RX_VERSION_REQ	-,*	**	**
MP+ADD_REQ	5,Add:B	**	**
MP+ADD_RSP	6,+	**	**
MP+ADD_COMP	**	**	9,Idle:A
MP+RX_RM_REQ	10,+	10,+	10,+
MP+RX_RM_RSP	11,+	11,+	11,+
MP+RX_RM_RX_REQ	12,+	12,+	12,+
MP+RX_RM_TX_REQ	13,+	13,+	13,+
MP+RX_RM_TX_RSP	14,+	14,+	14,+
MP+RX_REMOVE_REQ	-,*	**	**
MP+START_RM	15,*	15,*	15,*
MP+SEND_RMS	16,*	16,*	16,*
MP+SEND_RMM	17,*	17,*	17,*
MP+RECV_RMM	18,*	18,*	18,*
MP+STOP_RM	19,*	19,*	19,*
All other events	**	**	**

Table 4: Add State Machine

Actions:

1   Phone numbers (may) have been reserved, they must be released
          before the normal error processing occurs.

                    Release all reserved phone numbers

                    Do   Error Close Action.

2   Phone nubmers (may) have been reserved, they must be released
          before the normal close processing occurs.

                    Release all reserved phone numbers

                    Do   Term Action.

3   Do   Timer Action

4   Note that a bandwidth change has been reqested. This will be
          processed the next time IDLE state is entered.

5   An add collision has occured. Since the answerer has sent phone
          numbers we will try to use what he as sent, within the limits of
          the local system.

                    Compare local channels to add with current channels to
                    add.

                    If the local channels to add is less than the remote
                    channels to add

                              If the remote number of channels requested will
                              put us over the maximum number of channels
                              allowed for the session reduce the channel count
                              and set an add lock.

                              Re-reserve the channels. If the number reserved
                              are less than the number of phone numbers
                              provided by the far end, set an add lock and
                              reduce the number of channels to add to what we
                              could reserve.

                    Now treat the remote add request as if it were an add
                    response and process by:

                              Integrate the phone number returned from the
                              answerer with the original phone number dialed.

Request that a session be extended by dialing
the integrated phone number. A callback is
passed with the request so call success or
failure can be reported back to MP+.

Go to ADD:B. Note: This change must actually
occur before requesting the first outgoing call.
If not, the callback could be called (and
ignored) because the session is not in the
correct state.

6   If the answerer provided fewer phone numbers than requested set
         an add lock.

         If the number of channels is zero send an add complete message
         (there's nothing to do) and go to the IDLE state.

         For each phone number returned

                 Integrate the phone number returned from the answerer
                 with the original phone number dialed.

                 Request that a session be extended by dialing the
                 integrated phone number. A callback is passed with the
                 request so call success or failure can be reported back
                 to MP+.

         Go to ADD:B. Note: This change must actually occur before
         requesting the first outgoing call. If not, the callback could
         be called (and ignored) because the session is not in the correct
         state.

7   Restart idle timer for abort.

8   Increment the count of calls completed.

         If the call succeeded, increment the count of calls that
         succeeded.

         If the count of calls completed equals the number of calls placed

                 If number of calls completed is not the same as the
                 nubmer that succeeded set an add lock.

                 Build an add complete message and send it to the far end.

                 If at least one channel was added clear any remove lock.

        Go to the IDLE state.

9   If number of channels requested not equal to number connected
        set add lock.

        If at least one channel was added clear any remove lock.

        Go to the IDLE state.

10   Do Remote Management Request Action

11   Do Remote Management Response Action

12   Do Remote Management Receive Data Action

13   Do Remote Management Transmit Data Action

14   Do Remote Management Transmit Data Response Action

15   Do Remote management (Master) start Action

16   Do Remote management (Slave) data Action

17   Do Remote management (Master) data Action

18   Do Remote management data acknowledgement Action

19   Do Remote management (Master) stop Action

3.5.6. MP+STATE_REMOVE state machine

   The state of a session while processing a remove transaction.

   The sub-states are:

        A   Remove request sent, waiting for remove response
        B   Remove response sent, waiting for remove complete

Event	Sub-state	
	A	B
MP+SESSION_DOWN	1,Initial:A	1,Initial:A
MP+SESSION_TERM	2,Initial:A	2,Initial:A
MP+TIMER_EXPIRED	3,+	3,+
MP+UTILIZATION	4,*	4,*
MP+RX_ADD_REQ	5,+	**
MP+RX_REMOVE_REQ	6,+	**
MP+RX_REMOVE_RSP	7,Idle:A	**
MP+RX_REMOVE_COMP	**	8,Idle:A
MP+RX_CLOSE_REQ	-,*	**
MP+RX_RM_REQ	9,*	9,*
MP+RX_RM_RSP	10,*	10,*
MP+RX_RM_RX_REQ	11,*	11,*
MP+RX_RM_TX_REQ	12,*	12,*
MP+RX_RM_TX_RSP	13,*	13,*
MP+START_RM	14,*	14,*
MP+SEND_RMS	15,*	15,*
MP+SEND_RMM	16,*	16,*
MP+RECV_RMM	17,*	17,*
MP+STOP_RM	18,*	18,*
All other events	**	**

Table 5: Remove State Machine

Actions:

1   Do   Error Close Action

2   Do   Term Action

3   Do   Timer Action

4   Note that a bandwidth change has been reqested. This will be
         processed the next time IDLE state is entered.

5   Our remove conflicted with the remote end Add. The add takes
         preference.

         Set a remove lock.

         If we are the caller Do Enter Add [remote answerer] Action .

         Otherwise Do Enter Add [remote caller] Action .

6   Two remove requests collided. We give preference to the caller
         (an arbitrary decision).

         If caller, ignore message.

         Else
                 Check maximum number of channels needed by the local end.
                 Reduce the requested remove count and set a remove lock if
                 necessary.

                 Build and send a remove response to the remote.

                 Go to Remove:B.

7   Compare the number of channels requested with the number allowed
         in the response. If fewer allowed set a remove lock.

         Look at the current bandwidth. If the number to remove would
         bring the current bendwidth below requirements reduce the number
         of channels to remove.

         If still channels to remove:
                 Remove the channels.
                 Clear any add lock.
         Send a remove complete indicating the number of channels removed.

8   If at least one channel was removed clear any add lock.

9   Do Remote Management Request Action

10   Do Remote Management Response Action

11   Do Remote Management Receive Data Action

12   Do Remote Management Transmit Data Action

13   Do Remote Management Transmit Data Response Action

14   Do Remote management (Master) start Action

15   Do Remote management (Slave) data Action

16   Do Remote management (Master) data Action

17   Do Remote management data acknowledgement Action

18   Do Remote management (Master) stop Action

3.5.7. MP+STATE_CLOSE state machine

   The close state is used when we are gracefully closing a session or
   when we were notified that a session terminated mid-transaction.

   The sub-states are:

   A   Waiting for call complete after session down notification
   B   Waiting for call complete after session terminate
           notification.
   C   Waiting for close response after session close request
           sent.

Event	Sub-state		
	A	B	C
MP+SESSION_DOWN	**	-,*	7,Initial:A
MP+SESSION_TERM	1,Close:B	**	8,Initial:A
MP+TIMER_EXPIRED	2,Initial:A	5,Initial:A	6,*
MP+UTILIZATION	-,*	-,*	9,*
MP+CALL_COMPLETE	3,+	6,Initial:A	**
MP+ADD_REQ	**	-,*	10,+

```
_____
MP+REMOVE_REQ                  **                **            11,Remove:B
_____
MP+CLOSE_REQ                  -,*               -,*            12,*
_____
MP+CLOSE_RSP                  -,*               -,*            13,+
_____
MP+START_RM                   4,*               4,*            4,*
_____
All other events               **                **             **
_____
```

Table 6: Close State Machine

Actions:

1   The session was closed while waiting for call completes.
        Just go to Close:B.

2   We timed out waiting for completes. Just process the link down,
        now.
        Do Error Close Action.

3   Increment the number of calls complete.

        If equal to the number of calls placed then:
                Do Error Close Action, go to Initial:A.
        Else
                No state change.

4   Log an error message.
        Notify the user interface of remote management failure.

5   We didn't get all the notifications that we expect. Give up and
        close the session anyway. Do Term Action .

6   Increment the number of calls complete.

        If equal to the number of calls placed then:
                Do Term Action, go to Initial:A.
        Else
                No state change

7   Do Error Close Action

8   Do Term Action

9   Note that a bandwidth change has been reqested. This will be

          processed the next time IDLE state is entered.

10   This is an Add & Close collision. Add wins. Perform current
          remote add action.
          If we are originator
                  Do Add [Remote Answerer] Action
          else
                  Do Add [Remote Caller] Action

11   This is a Remove & Close collision, the Remove will win:
                  Set remove lock to FALSE
                  Do Remove [Remote] Action.

12   This is a Close collision. But since we both agree:
          If we are originator
                  Send a Close Response with okToClose set to TRUE.
          Else
                  Send a Close Response with okToClose set to FALSE.

13   If Close Response is received with okToClear is TRUE then:
                  Do Term Action
          Else
                  set remove lock to TRUE and do Enter Idle Action.

4. PPP LCP Extensions

MP+ Configuration Option

   The Multilink Protocol Plus introduces the use of an additional LCP
   Configuration Option:

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  type = 22    |  length = 4   |     Currently unused          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
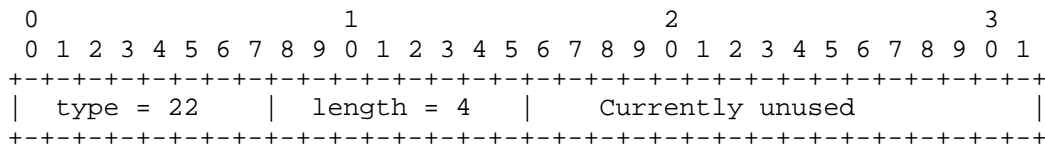
   Figure 21: MP+ Option

   Type - 22.

   NOTE: The current implementation uses option 0. This is not an
   assigned number, so an IANA assigned official identifier has been
   obtained (22).

   The option, when sent to a peer, advises the peer that:

        the unit is capable of running the MP+ protocol;

The peer can accept or reject the option.

NOTE: The MP+ option MUST NOT be included unless MP is also
negotiated.

5. Security Considerations

Security issues are not discussed in this memo.

6. References

[1] K. Sklower, B. Lloyd, G. McGregor, D. Carr, "The PPP Multilink
Protocol (MP)".

[2] Simpson, W., Editor, "The Point-to-Point Protocol (PPP)", STD
51, RFC 1661, Daydreamer, July 1994.

7. Author's Address

Kevin Smith
Ascend Communications
1275 Harbor Bay Parkway
Alameda, CA 94502

Phone: (510) 769-6001
FAX: (510) 814-2300
EMail: ksmith@ascend.com