Except for the following rules, the determination of which party-states are incompatible is left as an implementation decision:

a) If a STATUS message indicating any party-state except the Null party-state is received in the Null party-state, then the receiving entity shall send a DROP PARTY ACKNOWLEDGE message with cause #101, *"message not compatible with call state"*; and remain in the Null party-state.

b) If a STATUS message indicating any party-state except the Null party-state is received in the Drop Party Initiated party-state, no action shall be taken.

c) If a STATUS message, indicating the Null party-state, is received in any party-state except the Null party-state, the receiver shall internally clear the party and enter the Null party-state. If no other party of the call is in the Active, Add Party Initiated or Add Party Received party-states call clearing will be initiated with a RELEASE message.

When in the Null party-state, the receiver of a STATUS message indicating the Null party-state shall take no action other than to discard the message and shall remain in the Null party-state.

A STATUS message may be received indicating a compatible party-state but containing one of the following causes:

#96 *"mandatory information element is missing";*

#97 *"message type non-existent or not implemented"*;

#99 *"information element non-existent or not implemented"*; or

#100 *"invalid information element contents"*.

In this case, the actions to be taken are an implementation option. If other procedures are not defined, the receiver shall clear the party with the appropriate procedure defined in §5.6.3, using the cause specified in the received STATUS message.

5.7 List of Timers

The description of timers in the following tables should be considered a brief summary. The precise details are found in §§5.5 and 5.6, which should be considered the definitive description.

5.7.1 Timers in the Network Side

The timers specified in Table 5-9 are maintained in the network side of the UNI.

Table 5-9 Timers in the network side (page 1 of 2)

TIMER NUM-BER	DEFAULT TIME OUT VALUE	STATE OF CALL	CAUSE FOR START	NORMAL STOP	AT THE FIRST EXPIRY	AT THE SECOND EXPIRY	IMPLEMEN-TATION
T301	Not supported in this Implementation Agreement						
T303	4 s	Call Present	SETUP sent.	CONNECT, CALL PROCEED-ING, or RELEASE COMPLETE received.	Retransmit SETUP; restart T303. If retransmission of SETUP is not supported, clear network connection and enter Null state.	Clear network connection. Enter Null state.	Mandatory
T308	30 s	Release Indication	RELEASE sent.	RELEASE COMPLETE or RELEASE received.	Retransmit RELEASE and restart T308.	Release call reference and enter Null state. (Note 1)	Mandatory
T309	10 s	Any stable state	SAAL disconnec-tion. Calls in the active state are not lost.	SAAL reconnected	Clear network connection. Release virtual channel and call reference	Not applicable.	Mandatory

Table 5-9 Timers in the network side (page 2 of 2)

TIMER NUM-BER	DEFAULT TIME OUT VALUE	STATE OF CALL	CAUSE FOR START	NORMAL STOP	AT THE FIRST EXPIRY	AT THE SECOND EXPIRY	IMPLEMEN-TATION
T310	10 s	Incoming Call Proceeding	CALL PROCEED-ING received.	CONNECT or RELEASE received.	Clear call in accordance with §5.5.4	Not applicable.	Mandatory
T316	2 min	Restart Request	RESTART sent.	RESTART ACKNOWL-EDGE received.	RESTART may be retransmitted several times.	RESTART may be retransmitted several times	Mandatory
T317	(Note 2)	Restart	RESTART received	Internal clearing of call references.	Maintenance notification	Not applicable.	Mandatory
T322	4 s	Any call state	STATUS ENQUIRY sent.	STATUS, RELEASE, or RELEASE COMPLETE received.	STATUS ENQUIRY may be retransmitted several times.	STATUS ENQUIRY may be retransmitted several times	Mandatory
T398	4 s	Drop Party Initiated party-state	DROP PARTY sent	DROP PARTY ACKNOWL-EDGE or RELEASE received.	Send DROP PARTY ACKNOWL-EDGE or RELEASE (see §5.6.3.4.1)	Timer is not restarted.	Mandatory
T399	14 s	Add Party Initiated party-state	ADD PARTY sent	ADD PARTY ACKNOWL-EDGE, ADD PARTY REJECT, or RELEASE received.	Clear the party and enter the Null party-state (see §5.6.2.5.3)	Timer is not restarted.	Mandatory

Note 1 - The restart procedures in §5.5.5 may be used.

Note 2 - The value of this timer is implementation dependent but should be less than the value of T316.

5.7.2 Timers in the User Side

The timers specified in Table 5-10 are maintained in the user side of the UNI.

Table 5-10 Timers in the user side (page 1 of 2)

TIMER NUMBER	DEFAULT TIME OUT VALUE	STATE OF CALL	CAUSE FOR START	NORMAL STOP	AT THE FIRST EXPIRY	AT THE SECOND EXPIRY	IMPLEMENTATION
T303	4 s	Call Initiated	SETUP sent.	CONNECT, CALL PROCEEDING or RELEASE COMPLETE received.	Retransmit SETUP; restart T303. If retransmission of SETUP is not supported, clear internal connection and enter Null state.	Clear internal connection. Enter Null state.	Mandatory
T308	30 s	Release Request	RELEASE sent.	RELEASE COMPLETE or RELEASE received.	Retransmit RELEASE and restart T308.	Release call reference and enter Null state. (Note 1)	Mandatory
T309	10 s	Any stable state	SAAL disconnection. Calls in the active state are not lost.	SAAL reconnected.	Clear internal connection. Release virtual channel and call reference.	Not applicable.	Mandatory
T310	10 s	Outgoing Call Proceeding	CALL PROCEEDING received.	CONNECT or RELEASE received.	Send RELEASE.	Not applicable.	Mandatory
T313	4 s	Connect Request	CONNECT sent.	CONNECT ACKNOWLEDGE received.	Send RELEASE.	Not applicable.	Mandatory

Table 5-10 Timers in the user side (page 2 of 2)

TIMER NUM-BER	DEFAULT TIME OUT VALUE	STATE OF CALL	CAUSE FOR START	NORMAL STOP	AT THE FIRST EXPIRY	AT THE SECOND EXPIRY	IMPLEMEN - TATION
T316	2 min	Restart Request	RESTART sent	RESTART ACKNOWL - EDGE received	RESTART may be retransmitted several times	RESTART may be retransmitted several times	Mandatory
T317	(Note 2)	Restart	RESTART received.	Internal clearing of call references.	Maintenance notification.	Not applicable.	Mandatory
T322	4 s	Any call state	STATUS ENQUIRY sent.	STATUS, RELEASE, or RELEASE COMPLETE received.	STATUS ENQUIRY may be retransmitted several times.	STATUS ENQUIRY may be retransmitted several times.	Mandatory
T398	4 s	Drop Party Initiated party-state	DROP PARTY sent.	DROP PARTY ACKNOWL - EDGE or RELEASE received.	Send DROP PARTY ACKNOWL - EDGE or RELEASE (see §5.6.3.3)	Timer is not restarted.	Mandatory
T399	14 s	Add Party Initiated party-state	ADD PARTY sent.	ADD PARTY ACKNOWL - EDGE, ADD PARTY REJECT, or RELEASE received.	Clear the party and enter the Null party-state (see §5.6.2.5.3)	Timer is not restarted.	Mandatory

Note 1 - The restart procedures in §5.5.5 may be used.

Note 2 - The value of timer T317 is implementation dependent but should be less than the value of T316.

5.8 Address Registration

This section specifies the procedures for address registration at the UNI. These procedures are specified as an extension to the Interim Local Management Interface (ILMI) specified in §4.

Equipment at the Private UNI must support the Address Registration procedures described in this section.

Equipment at the Public UNI may support the Address Registration procedures described in this section.

5.8.1 Overview

In order establish an ATM connection at the UNI, both the user and the network must know the ATM address(es) which are in effect at that UNI. These ATM addresses can then be used in Calling Party Number information elements of signalling messages sent by the user, and in Called Party Number information elements of signalling messages sent to the user. The address registration procedures in this section provide the means for the dynamic exchange of addressing information between the user and the network at the UNI, at initialization and at other times as required. Through this dynamic exchange the user and network can agree on the ATM address(es) in effect.

As specified in §5.1.3.1, the Private ATM Address Structure consists of multiple fields. Two of these fields, the End System Identifier (ESI) and the Selector (SEL) fields form the “user part” and are supplied by the user-side of the UNI. All other fields form a “network prefix” for ATM addresses in that they typically have the same value for all ATM addresses on the same ATM UNI; the value of the network prefix is supplied by the network side of the UNI. The network-side is allowed to supply multiple network prefixes for use at a single UNI; however, it is expected that just one will normally be supplied. An ATM address for a terminal on the user-side of a Private UNI is obtained by appending values for the ESI and SEL fields to (one of) the network prefix(es) for that UNI.

For the purposes of address registration, the value of the SEL field is irrelevant, i.e., one user part is a duplicate of another if they have the same ESI value even if they have different SEL values.

ATM addresses in Public networks either have the same Private ATM Address Structure as Private networks, or else they are E.164 addresses (see §5.1.3.2). For Public network E.164 addresses, the network-side supplies the whole ATM address. In effect, the network prefix is the E.164 address and can only be validly combined with a null user part. In this situation, the use of multiple network prefixes at a single UNI occurs whenever multiple E.164 addresses are used at that UNI.

5.8.2 Capabilities

The address registration procedures in this section provide the following capabilities:

- initialization-time exchange of addressing information,
- restrictions on network-prefix/user-part combinations,
- rejection of unacceptable values, either rejection of a specific network prefix by the user, or of a specific user part by the network,
- dynamic addition/deletion of additional network prefixes and user parts,
- de-registration of addresses on a UNI "link down" condition.

The following sections describe each capability in more detail.

5.8.2.1 Initialization-time Exchange of Addressing Information

These procedures allow the user and the network to exchange addressing information to allow ATM addresses to be registered at the UNI and used in signalling messages. This includes the capability for the network to support more than one network prefix (e.g., in a transitional period to allow both an old and a new network prefix value), and for the user to support more than one user part.

5.8.2.2 Restrictions on Network-Prefix/User-Part Combinations

Not all combinations of network prefix and user part may be valid. For example, if the network prefix is an E.164 address (using the E.164 numbering plan), then no user part is necessary or allowed. Alternatively, at a Public UNI which supports Address Registration and connects a Private ATM network and a Public ATM network, the user (the Private ATM network) may wish to restrict combinations of network prefixes offered by the Public network and user parts that it registers (e.g., to simplify its routing task).

5.8.2.3 Rejection of Unacceptable Values

Certain user part values, as specified by the user, may be unacceptable to the network. An example would be if the user attempted to register a user part which is already registered on a different UNI. In this case, the network may wish to block registration of the duplicate. Similarly, a user may wish to block registration of a network prefix supplied by the network. For example, at a Public UNI which supports Address Registration and supports E.164 addresses, both using the E.164 numbering plan and using NSAP encoding, the user might wish to block registration one of the address formats.

5.8.2.4 Dynamic Addition/Deletion

The list of network prefixes and user parts, and their valid combinations may change over time during the operation of the UNI. For example, on the user-side of a private UNI, new user parts may appear or disappear when (individually addressable) components are connected to or disconnected from the access device. On the network-side, the list of valid network prefixes may change due to topology changes.

5.8.2.5 De-registration on UNI "link down" Condition.

The addresses registered at the UNI are de-registered when the UNI is declared to be down. For example, in order for a user device using the Private ATM Address structure to be unplugged from one UNI and plugged in to another UNI on the same network, the user-part(s) registered by the device must be de-registered from the first UNI and re-registered on the second.

5.8.3 General Description of Procedures

For the exchange of addressing information, two MIB tables are defined, one to contain network prefixes, and the other to contain registered ATM addresses. The Network Prefix table contains one entry for each network prefix. The Address table contains one entry for each registered ATM address. The MIB definitions of these tables is given in §5.8.6. The basic approach of the procedures are as follows.

For network prefixes, it is the network-side which supplies the values for the user-side to accept or reject. Thus, it is the user-side which implements the Network Prefix table, and the network-side which issues ILMI SetRequest messages to create/delete entries in the table in order to register/de-register network prefixes.

For registered addresses, it is the user-side which supplies the values for the network-side to accept or reject. Thus, it is the network-side which implements the Address table, and the user-side which issues ILMI SetRequest messages to create/delete entries in the table in order to register/de-register ATM addresses.

At initialization, the registration of network prefixes occurs first. Next, the user-side combines each of the user parts it wishes to use with one or more of the registered network prefixes to form a set of ATM addresses. The user-side then registers these addresses.

After initialization, the network-side issues ILMI SetRequest messages to create/delete entries in the Network Prefix table as and when new network prefixes need to be added or existing network prefixes deleted. Similarly, the user-side issues ILMI SetRequest messages to create/delete entries in the Address table as and when new ATM addresses need to be registered or existing ATM addresses de-registered. If and when the UNI link goes down, all addresses are de-registered.

5.8.4 Management Information Base for Address Registration

Address registration is defined as an extension to the ILMI through the definition of two additional MIB groups: the Prefix group, and the Address group.

5.8.4.1 Prefix Group (CR)

The Prefix group is required to be implemented by the UME on the user-side of the UNI. This group consists of one MIB table, the Network Prefix Table, indexed by the UNI port number and by the value of a network prefix. The information in this group is:

- Interface Index
- Network Prefix
- Network Prefix Status

5.8.4.1.1 Network Prefix

The Network Prefix object has the value of a network prefix.

5.8.4.1.2 Network Prefix Status

The Network Prefix Status object provides an indication of the validity of a network prefix at this UNI. To configure a new network prefix, the network-side uses an ILMI SetRequest to set the Network Prefix Status object for that prefix to be valid. To delete an existing network prefix, the network-side uses an ILMI SetRequest to set the Network Prefix Status object for that prefix to be invalid.

5.8.4.2 Address Group (CR)

The Address group is required to be implemented by the UME on the network-side of the UNI. This group consists of one MIB table, the Address Table, which is indexed by the UNI port number and by the value of an ATM address. The information in this group is:

- Interface Index
- ATM Address
- ATM Address Status

5.8.4.2.1 ATM Address

The ATM Address object has the value of an ATM address.

5.8.4.2.2 ATM Address Status

The ATM Address Status object provides an indication of the validity of an ATM address at this UNI. To configure a new ATM address, the user-side uses an ILMI SetRequest to set the ATM Address Status object for that address to be valid. To delete an existing ATM address, the user-side uses an ILMI SetRequest to set the ATM Address Status object for that address to be invalid.

5.8.5 Address Registration Procedures

5.8.5.1 Network-Side UME

Upon its own restart and before sending the standard ColdStart trap, the network-side UME initializes the Address Table to be empty. It then sends the ColdStart trap and issues an ILMI message (i.e., a GetNext request) to read the first instance of the Network Prefix Status object from the user-side. If the response does not indicate that the Network Prefix table is empty, then this procedure is restarted by retransmitting the ColdStart trap.

The network-side UME also initializes the Address Table to be empty upon receipt of a ColdStart trap from the user-side UME.

In either case, the Network Prefix table will now be empty. Thus, the network-side now issues one (or more) SetRequests to register its network prefix(es), e.g.,

SetRequest { atmfNetPrefixStatus.port.prefix=valid(1) }

Upon receipt of a SetRequest setting an instance of the Address Status object to be valid, the network-side validates the referenced address. If the validation fails, it responds with a Response containing a badValue error. If the validation succeeds, if responds with a Response indicating noError, and if the address is not currently registered, it is registered and the Address table is updated.

Upon receipt of a SetRequest setting an instance of the Address Status object to be invalid, the network-side checks if the referenced address is currently registered. If not, it responds with a Response containing a noSuchName error. If the address is currently registered, then it responds with a Response indicating noError, the address is de-registered and the Address table is updated.

If at any time, the network-side wishes to register an additional network prefix, it issues an appropriate SetRequest, e.g.,

SetRequest { atmfNetPrefixStatus.port.prefix=valid(1) }

If at any time, the network-side wishes to de-register a network prefix, it issues an appropriate SetRequest, e.g.,

SetRequest { atmfNetPrefixStatus.port.prefix=invalid(2) }

If at any time, the network-side wishes to check the consistency of the set of network prefixes currently registered, it issues ILMI messages (e.g., GetNext requests) to read all instances of the Network Prefix Status object from the user-side and checks that the set of those which have a valid status is the correct set.

During operation, if the network-side receives an indication of link down, it de-registers all addresses. Procedures for declaring a link down are specified in §4.7.7.

5.8.5.2 User-Side UME

Upon its own restart and before sending the standard ColdStart trap, the user-side UME initializes the Network Prefix Table to be empty, and then sends the ColdStart trap. The user-side then issues an ILMI message (i.e., a GetNext request) to read the first instance of the ATM Address Status object from the network-side. If the response does not indicate that the ATM Address table is empty, then this procedure is restarted by retransmitting the ColdStart trap.

After ascertaining that the ATM Address table is empty, the user-side awaits one or more ILMI messages (i.e., SetRequests) from the network-side UME which set the Network Prefix Status object to valid for a particular network prefix. In the unlikely event that no such ILMI messages are received for a prolonged period (e.g., 5 seconds) after sending a ColdStart trap, the user-side should retransmit the ColdStart trap.

The user-side UME also initializes the Network Prefix Table to be empty upon receipt of a ColdStart trap from the network-side UME.

Upon receipt of a SetRequest setting an instance of the Network Prefix Status object to be valid, the user-side validates the request. If the validation fails, it responds with a Response containing the appropriate error. If the validation succeeds, it responds with a Response indicating noError, and if the network prefix is one not currently registered, the prefix is registered and the Network Prefix table is updated. For any address which the user-side wishes to register using the new prefix, it forms the address in one of two ways: by appending the ESI and SEL values to the prefix or, for non-NSAP-encoded E.164 addresses, by appending a null user part. It then issues an appropriate SetRequest, e.g.,

SetRequest { atmfAddressStatus.port.address=valid(1) }

Upon receipt of a SetRequest setting an instance of the Network Prefix Status object to be invalid, the user-side checks if the prefix is currently registered. If not, it responds with a Response containing a noSuchName error. If the prefix is currently registered, then it responds with a Response indicating noError, the prefix is de-registered and the Network Prefix table is updated.

If at any time, the user-side wishes to register an additional address having the Private ATM Address structure, it forms the address by appending the ESI and SEL values to one of the registered prefixes and issues an appropriate SetRequest, e.g.,

SetRequest { atmfAddressStatus.port.address=valid(1) }

If at any time, the user-side wishes to de-register an address, it issues an appropriate SetRequest, e.g.,

SetRequest { atmfAddressStatus.port.address=invalid(2) }

If the user-side wishes to check periodically that (at least one of) its address(es) is still registered (e.g., that it has not been unplugged from one UNI and plugged into another), then it may wish to poll the first instance of the Address Status object from the network-side and check that the retrieved instance is the one expected.

If at any time, the user-side wishes to check for consistency of all ATM addresses currently registered, it uses ILMI messages (e.g., GetNext requests) to read all instances of the Address Status object from the network-side and checks that the set of those with valid status is the correct set. (Note that when a single address is registered, this consistency check is equivalent to checking just the first address; when multiple addresses are registered, this full consistency check is expected to be needed much less frequently than checking just the first address.)

5.8.5.3 Retransmission of SetRequest Messages

For the above procedures, a UME should retransmit a SetRequest message if it does not receive a Response message within a time-out period. There are two possibilities when a Response is not received, e.g., due to a transmission error/loss, either: A) the SetRequest was processed and the generated Response was lost, or B) the SetRequest was lost before it was processed. When registering a network prefix/address, the retransmission of the appropriate SetRequest message produces the same result no matter which of A or B occurred. When de-registering a network prefix/address, the Response to a retransmission of the appropriate SetRequest message will have an error-status of either noError, noSuchName, or some other error. The noError status indicates that situation B existed, but the de-registration is now complete. The noSuchName status indicates that situation A existed, i.e., the prefix/address was already de-registered. Some other error indicates that the de-registration cannot be completed for some other reason.

5.8.6 MIB Definition

```
ATM-FORUM-ADDR-REG  DEFINITIONS ::= BEGIN

IMPORTS
        atmForumUni             FROM ATM-FORUM-MIB
        OBJECT-TYPE             FROM RFC-1212;

-- Textual Convention

-- Representations in this MIB Module of an ATM address
-- use the data type:

AtmAddress ::= OCTET STRING (SIZE (8 | 20))
```

```
-- Representations in this MIB Module of a network-prefix
-- for an ATM address use the data type:

NetPrefix ::= OCTET STRING (SIZE (8 | 13))

-- in both the AtmAddress and NetPrefix conventions, non-NSAP-encoded E.164
-- addresses are represented as 8 octets using the format specified in section 5.1.3.1.4.
-- In contrast, an NSAP-encoded address is 20 octets, and an NSAP-encoded network
-- prefix is 13 octets long.

--  New MIB Groups

atmfAddressGroup        OBJECT IDENTIFIER ::= { atmForumUni 6 }
atmfNetPrefixGroup      OBJECT IDENTIFIER ::= { atmForumUni 7 }


--              The Network Prefix Table
--
-- The Network Prefix Table is implemented by the UNI Management
-- Entity on the user-side of the UNI

atmfNetPrefixTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF AtmfNetPrefixEntry
        ACCESS      not-accessible
        STATUS      mandatory
        DESCRIPTION
                "A table implemented by the UNI Management Entity on the user-side of an
                ATM UNI port, containing the network-prefix(es) for ATM-layer addresses
                in effect on the user-side of the UNI."
        ::= { atmfNetPrefixGroup 1 }

atmfNetPrefixEntry OBJECT-TYPE
        SYNTAX      AtmfNetPrefixEntry
        ACCESS      not-accessible
        STATUS      mandatory
        DESCRIPTION
                "Information about a single network-prefix for ATM-layer addresses in effect
                on the user-side of a UNI port. Note that the index variable atmfNetPrefixPrefix
                is a variable-length string, and as such the rule for variable-length strings in
                section 4.1.6 of RFC 1212 applies."
        INDEX  { atmfNetPrefixPort, atmfNetPrefixPrefix }
        ::= { atmfNetPrefixTable 1 }
```

```
AtmfNetPrefixEntry ::=
        SEQUENCE {
                atmfNetPrefixPort       INTEGER,
                atmfNetPrefixPrefix     NetPrefix,
                atmfNetPrefixStatus     INTEGER
        }

atmfNetPrefixPort OBJECT-TYPE
        SYNTAX      INTEGER (0..2147483647)
        ACCESS      not-accessible
        STATUS      mandatory
        DESCRIPTION
                "A unique value which identifies the UNI port for which the network prefix
                for ATM addresses is in effect. The value of 0 has the special meaning of iden-
                tifying the local UNI."
        ::= { atmfNetPrefixEntry 1 }

atmfNetPrefixPrefix  OBJECT-TYPE
        SYNTAX      NetPrefix
        ACCESS      not-accessible
        STATUS      mandatory
        DESCRIPTION
                "The network prefix for ATM addresses which is in effect on the user-side of
                the ATM UNI port."
        ::= { atmfNetPrefixEntry 2 }

atmfNetPrefixStatus  OBJECT-TYPE
        SYNTAX      INTEGER { valid(1), invalid(2) }
        ACCESS      read-write
        STATUS      mandatory
        DESCRIPTION
                "An indication of the validity of the network prefix for ATM addresses on the
                user-side of the UNI port. To configure a new network prefix in this table, the
                network-side must set the appropriate instance of this object to the value
                valid(1). To delete an existing network prefix in this table, the network-side
                must set the appropriate instance of this object to the value invalid(2).

                If circumstances occur on the user-side which cause a prefix to become invalid,
                the user-side modifies the value of the appropriate instance of this object to
                invalid(2).

                Whenever the value of this object for a particular prefix becomes invalid(2),
                the conceptual row for that prefix may be removed from the table at any time,
                either immediately or subsequently."
```

```
        ::= { atmfNetPrefixEntry 3 }

--              The Address Table
--
-- The Address Table is implemented by the UNI Management Entity
-- on the network-side of the UNI

atmfAddressTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF AtmfAddressEntry
        ACCESS      not-accessible
        STATUS      mandatory
        DESCRIPTION
                "A table implemented by the network-side of an ATM UNI port, containing
                the ATM-layer addresses in effect on the user-side of the UNI."
        ::= { atmfAddressGroup 1 }

atmfAddressEntry OBJECT-TYPE
        SYNTAX      AtmfAddressEntry
        ACCESS      not-accessible
        STATUS      mandatory
        DESCRIPTION
                "Information about a single ATM-layer address in effect on the user-side of a
                UNI port. Note that the index variable atmfAddressAtmAddress is a variable-
                length string, and as such the rule for variable-length strings in section 4.1.6 of
                RFC 1212 applies."
        INDEX  { atmfAddressPort, atmfAddressAtmAddress }
        ::= { atmfAddressTable 1 }

AtmfAddressEntry ::=
        SEQUENCE {
                atmfAddressPort         INTEGER,
                atmfAddressAtmAddress   AtmAddress,
                atmfAddressStatus       INTEGER
        }

atmfAddressPort OBJECT-TYPE
        SYNTAX      INTEGER (0..2147483647)
        ACCESS      not-accessible
        STATUS      mandatory
        DESCRIPTION
                "A unique value which identifies the UNI port for which the ATM address is
                in effect. The value of 0 has the special meaning of identifying the local UNI."
        ::= { atmfAddressEntry 1 }
```

```
atmfAddressAtmAddress  OBJECT-TYPE
        SYNTAX          AtmAddress
        ACCESS          not-accessible
        STATUS          mandatory
        DESCRIPTION
                "The ATM address which is in effect on the user-side of the ATM UNI port."
        ::= { atmfAddressEntry 2 }

atmfAddressStatus  OBJECT-TYPE
        SYNTAX          INTEGER { valid(1), invalid(2) }
        ACCESS          read-write
        STATUS          mandatory
        DESCRIPTION
                "An indication of the validity of the ATM address at the user-side of the UNI
                port. To configure a new address in this table, the user-side must set the
                appropriate instance of this object to the value valid(1). To delete an existing
                address in this able, the user-side must set the appropriate instance of this
                object to the value invalid(2).

                If circumstances occur on the network-side which cause an address to become
                invalid, the network-side modifies the value of the appropriate instance of this
                object to invalid(2).

                Whenever the value of this object for a particular address becomes invalid(2),
                the conceptual row for that address may be removed from the table at any
                time, either immediately or subsequently."
        ::= { atmfAddressEntry 3 }

END
```

5.9 Signalling ATM Adaptation Layer (SAAL)

This section specifies the Signalling ATM Adaptation Layer (SAAL) for use at the UNI. The SAAL resides between the ATM layer and Q.2931. The purpose of the SAAL is to provide reliable transport of Q.2931 messages between peer Q.2931 entities (e.g., ATM Switch and host) over the ATM layer. The SAAL is composed of two sublayers, a common part and a service specific part. The service specific part is further subdivided into a Service Specific Coordination Function (SSCF), and a Service Specific Connection Oriented Protocol (SSCOP). Figure 5-47 illustrates the structure of the SAAL.

The SAAL for supporting signalling shall use the protocol structure as illustrated in Figure 5-47.

The Common Part AAL protocol provides unassured information transfer and a mechanism for detecting corruption of SDUs. AAL Type 5 Common Part protocol shall be used to

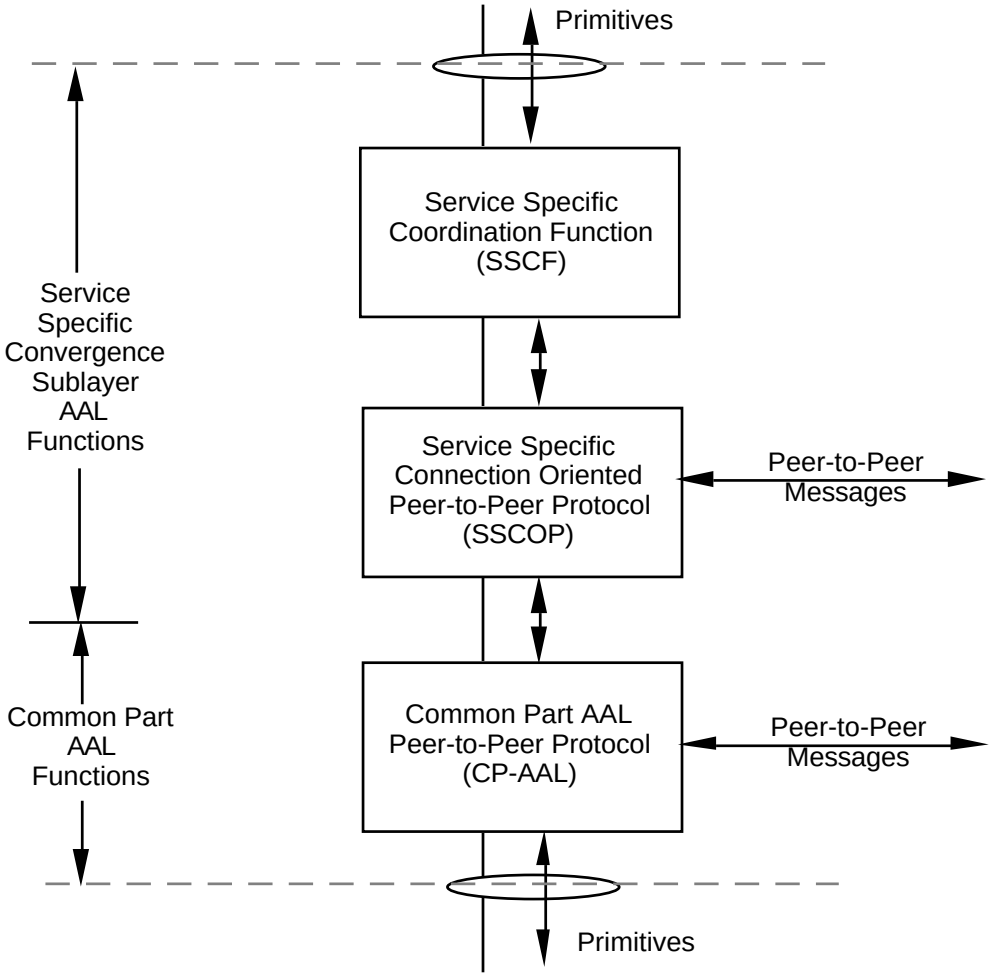
support signalling. The AAL Type 5 Common Part Protocol is specified in Draft Recommendation I.363 [30].

The Service Specific Connection Oriented Protocol (SSCOP) resides in the Service Specific Convergence Sublayer (SSCS) of the SAAL. SSCOP is used to transfer variable length Service Data Units (SDUs) between users of SSCOP. SSCOP provides for the recovery of lost or corrupted SDUs. SSCOP is specified in Q.2110 [31].

The SAAL for supporting signalling shall utilize SSCOP as specified in Q.2110 [31].

An SSCF maps the service of SSCOP to the needs of the SSCF user. Different SSCFs may be defined to support the needs of different AAL users. The SSCF used to support Q.2931 at the UNI is specified in Draft Recommendation Q.2130 [32].

The external behavior of theSAAL at the UNI shall appear as if the UNI SSCF as specified in Q.2130 [32] were implemented.



Note: the Figure represents the allocation of functions and is not intended to illustrate sub-layers as defined by OSI modeling principles.

Figure 5-47 SAAL Structure