The ATM Forum

Technical Committee

LAN Emulation over ATM Version 2 - LNNI Specification -

AF-LANE-0112.000

February, 1999

The ATM Forum
Worldwide Headquarters
2570 West El Camino Real, Suite 304
Mountain View, CA 94040-1313
Tel:+1-650-949-6700
Fax: +1-650-949-6705

Contents

List of Figures

List of Tables

1 Introduction

This document is one of two parts of LANE v2, the second revision of the LAN Emulation specification. This document describes the LAN Emulation Network Network Interface (LNNI). LNNI defines the protocols necessary to distribute the LE Configuration Servers (LECSs), LAN Emulation Servers (LESs), Selective Multicast Servers (SMSs[1]) and Broadcast and Unknown Servers (BUSs). Distribution of these servers with a standard LNNI facilitates load distribution, fault tolerance and multivendor service components supporting LANE. This document is a companion document to the LAN Emulation - LUNI v2 Specification [5]. Familiarity with [5] is assumed.

1.1 Purpose of Document

This document specifies an implementation agreement for the LAN Emulation Service. This set of protocols is referred to as the LAN Emulation Network Network Interface (LNNI) protocols.

1.2 Terminology

The following acronyms and terminology are used throughout this document:

AAL	ATM Adaptation Layer
ATM	Asynchronous Transfer Mode
B-LLI	Broadband Low Layer Information
BUS	Broadcast and Unknown Server
CA	Cache Alignment
CK	Cache Key
CSA	Client State Advertisement
CSU	Cache State Update
ELAN	Emulated Local Area Network
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
ILMI	Interim Local Management Interface
LAN	Local Area Network

[1] SMS is not mentioned explicitly in [5] but sufficient mechanisms are defined in [5] for an LE Client and SMS to interact (e.g. LE Client can obtain the ATM address of an SMS via LE_ARP and SMS learns via SCSP which LE Clients wish to receive multicast traffic).

LANE	LAN Emulation
LE Client	LAN Emulation Client
LE_ARP	LAN Emulation Address Resolution Protocol
LECID	LAN Emulation Client Identifier
LECS	LAN Emulation Configuration Server
LES	LAN Emulation Server
LLC	Logical Link Control
LNNI	LAN Emulation Network Network Interface
LUNI	LAN Emulation User Network Interface
MAC	Media Access Control
OID	(Originator Identifier) SID of the server which initially created the CSA.
OUI	Organizationally Unique Identifier
QoS	Quality of Service
RFC	Request For Comment (IETF Document Series)
SCSP	Server Cache Synchronization Protocol
SG	Server Group
SGID	Server Group ID
SID	Server Identifier uniquely identifying a server within an ELAN.
SMF	Selective Multicast (capable) Flag
SMS	Selective Multicast Server
SN	Sequence Number
TLV	Type / Length / Value Encoded Parameter
UNI	User-Network Interface
VCC	Virtual Channel Connection

This document uses normative statements throughout as follows:

Table 1-1. Normative Statements

Statement	Verbal Form2
Requirement	MUST/MUST NOT
Recommendation	SHOULD/SHOULD NOT
Permission	MAY

1.3 References

[1] The ATM Forum, ATM User-Network Interface Specification, Version 3.0, af-uni-0010.001, September 10, 1993.

[2] The ATM Forum, ATM User-Network Interface Version 3.1 (UNI 3.1) Specification, af-uni-0010.002, July 21, 1994.

[3] The ATM Forum, LAN Emulation Over ATM Version 1.0 Specification (af-0021-000), January, 1995.

[4] IETF, Server Cache Synchronization Protocol - SCSP, Luciani et al, RFC 2334, April 1998.

[5] The ATM Forum, LAN Emulation - LUNI Specification Release 2.0 , af-lane-0084.000, July 1997.

[6] The ATM Forum, ATM User-Network Interface (UNI) Signalling Specification Version 4.0, af-sig-0061.000, July 1996.

[7] The ATM Forum, Traffic Management Specification Version 4.0, af-tm-0056.000, April 1996.

1.4 ATM Network Service Assumptions

This LAN Emulation Over ATM specification is based on the ATM Forum User-Network Interface Specification, Version 3.0 [1] or later. The specification provides example Information Element codings for UNI 3.0 [1], 3.1 [2], and 4.0 [7].

[2]Verbal forms are based on ISO except for “Requirements,” where ISO uses the terms “SHALL/SHALL NOT” instead of “MUST/MUST NOT” in this document.

2 Architectural Overview

2.1 Emulated LANs

An emulated LAN (ELAN) provides the appearance of either an Ethernet or Token-Ring LAN segment over a switched ATM network. An ELAN provides applications on ATM networks with the services found on more traditional LANs, such as a broadcast medium and topologically independent MAC addressing. Applications that run over Ethernet/802.3 or Token-Ring/802.5 LANs also run over emulated LANs on ATM networks. In the same way that Ethernet and Token-Ring Network Interface Cards (NICs) provide a physical interface onto a LAN, a LAN Emulation Client provides a virtual interface onto an emulated LAN.

An ELAN is composed of a collection of LE Clients and a set of co-operating service entities. LE Clients on the same ELAN may setup direct communications to one another. LE Clients on different ELANs require an interconnection device such as a bridge or router in order to communicate.

Although LAN Emulation does not exploit all of the benefits of an ATM network, it is useful in migrating to ATM technology and in lowering network management costs. High speed ATM networks may be utilized, while software and hardware investments remain protected. Software investments are protected because application interfaces are unchanged, and hardware investments are protected with devices that bridge LAN and ATM technologies. Network management is improved because there is increased flexibility in moves, adds, or changes in the network.

2.2 LNNI Scope and LUNI Scope Contrasted

The LAN Emulation LUNI specifications [3],[5] define the protocols and interactions between LAN Emulation Clients (LE Clients) and the LAN Emulation Service. The LAN Emulation Service is itself composed of a variety of components: LAN Emulation Configuration Servers (LECSs), LAN Emulation Servers (LESs), Broadcast and Unknown Servers (BUSs), and Selective Multicast Servers (SMSs). Each LE Client connects across the LUNI to a single LES and BUS, may connect to a single LECS, and may have connections to multiple SMSs. The LE Client LUNI connections are defined in [3],[5] and are depicted in Figure 2-1. The behaviour of the LANE Service components as seen by LE Clients is completely defined in the LUNI specifications.

The LAN Emulation NNI (LNNI), this document, defines the behaviour of these LANE service components as seen by each other. The LNNI defines the procedures necessary to provide a distributed and reliable LAN Emulation Service. A single ELAN may be served by multiple LECSs, LESs, BUSs and SMSs. Each LES, BUS, and SMS serves a single ELAN, while an LECS may serve multiple ELANs. LANE service components interconnect with multiple VCCs for Configuration, Status, Database Synchronization, Control and Data forwarding. It is the purpose of the LNNI specification to provide multivendor interoperability among the components serving an ELAN so that consumers may mix and match the LANE Service implementations of different vendors.

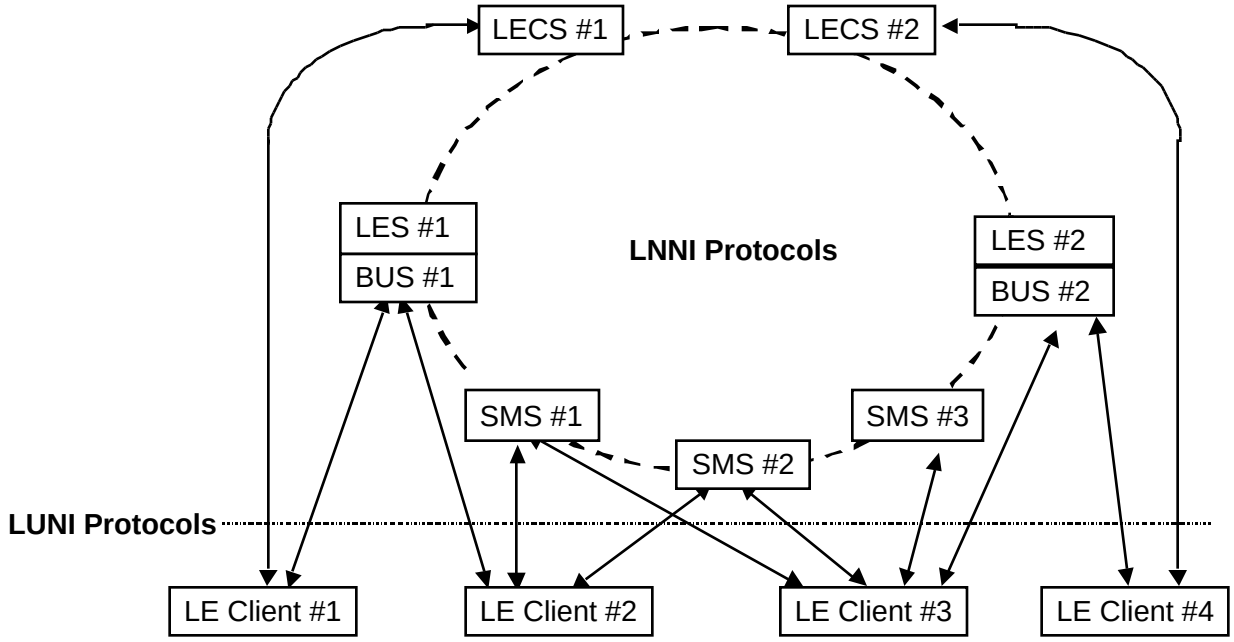

Figure 2-1 LUNI and LNNI Protocols

The LNNI is not a single interface. Each type of service component participates in different interface(s). The LNNI connections that provide the channel for these interfaces are as follows.

a) LECS LNNI Connections

- LECS Synchronization VCC(s) to other LECS
- Configuration Direct VCC(s) from one or more LES and SMS

b) LES LNNI Connections

- Configuration Direct VCC to an LECS
- Cache Synchronization VCC(s) to neighbour LES(s) or SMS(s)
- Control Coordinate VCC(s) to all other LES

c) BUS LNNI Connections

- Multicast Forward VCC(s) to all other BUSs (Note: this connection can go to LE Clients and hence also form part of the LUNI [5])

d) SMS LNNI Connections

- Configuration Direct VCC to an LECS
- Cache Synchronization VCC(s) to neighbour LES(s) or SMS(s)
- Multicast Forward VCC to other SMS(s) and BUS(s)

The general function of each LANE Service component is described in Section 2.3, and the communications between and among the service components is discussed in Section 2.4.

2.3 LAN Emulation Service Components

The LAN Emulation Service consists of four types of components: LECS(s), LES(s), BUS(s), and SMS(s). These components are described in the following sections.

2.3.1 LAN Emulation Configuration Servers

LECSs reduce the network management burden by serving as a central repository for configuration data. Each LANE service component establishes a Configuration Direct VCC to an LECS using the same procedures specified for LE Clients in the LUNI[5]. Unlike Clients however, Servers maintain these connections and send periodic Keep-Alive messages while participating in the ELAN.

Different entities on the same ELAN may connect to different LECSs. Once connected to an LECS, LANE entities may query the LECS to obtain configuration. For an LE Client, the LECS assigns the LE Client to a specific LES serving an ELAN. Additional configuration may be returned along with the ATM address of the LES. For an LES or SMS, the LECS returns the ATM addresses of neighbouring service entities, with additional configuration also possible. BUSs do not configure with the LECS. Each BUS is assumed to be logically associated with an LES, and may communicate with its LES to obtain its configuration.

An LECS may also dynamically update the configuration of other service components. However, no provision is made for an LECS to dynamically change the configuration of an LE Client directly. An LECS may trigger the reconfiguration of a particular LES, which may result in that LES terminating certain LE Clients, and those LE Clients may return to an LECS for new configuration. Thus, indirectly at least, an LECS may dynamically modify the configuration of an LE Client.

Multiple LECSs may be visible to the components of an ELAN. In order to meet the requirements imposed on the LANE Service by the LUNI specification [5], it is required that each LECS have knowledge of which LESs are currently serving which ELANs. This specification defines a protocol by which multiple LECSs may synchronize their knowledge on the status of servers within ELANs. This status information is exchanged between LECS on LECS Synchronization VCC(s). In cases where multiple LECSs are visible to the LANE entities within an ELAN, synchronization of configuration data between LECSs must be provided by means outside of this specification.

Configuration via the LECS is mandatory behaviour on the part of any LNNI compliant LANE service component.

2.3.2 LAN Emulation Servers

LESs implement the control aspects of emulated LANs. Each LES maintains a database of all registered LAN destinations and their associated ATM address(s). The LES may use this information to answer LE_ARPs, which request the ATM address associated with a particular LAN destination. Each LES is also responsible for forwarding certain control frames between different LE Clients and between LE Clients and other LESs.

Multiple LESs serving an ELAN must synchronize their databases so that every server on the ELAN has a complete and up-to-date database. An LES establishes Cache Synchronization VCC(s) to neighbour LES(s) assigned by the LECS during configuration. Database synchronization is accomplished across these connections using the Server Cache Synchronization Protocol (SCSP) defined in [4]. A brief overview of SCSP is provided in Sections 2.4.4 and 2.5, with emphasis on how SCSP applies to the LNNI protocols. More details on the use of SCSP for the LNNI are found in Sections 4.2 and 5.4.2 below.

LE Clients join with a particular LES in an ELAN by establishing a Control Direct VCC to the LES and issuing an LE_JOIN request. The LES may then establish a Control Distribute VCC back to the LE Client, and return an LE_JOIN response indicating that the LE Client has joined the ELAN. The LE Client may then register any number of LAN destinations with that LES (a single registration may have been performed in conjunction with the join). LE Clients joined with a particular LES are referred to as local LE Clients for that LES. Each LES is

responsible for control communications between local LE Clients, and for control communications between local LE Clients and other LESs. Other LESs are, in turn, responsible for control communications to their local LE Clients.

All LESs serving an ELAN must be interconnected into a full-mesh of Control Coordinate VCCs. These connections are used to forward control messages between LESs.

A single VCC can be used for both Cache Synchronization (using SCSP) and Control Coordinate communications (forwarding LANE Control messages) between two neighbour LESs.

The following example illustrates the typical control frame forwarding behaviour of an LES. When an LE Client has a frame to send to a LAN destination for which the associated ATM address is not known, the LE Client sends an LE_ARP request to its local LES. The LES may answer the LE_ARP request itself, or forward the LE_ARP request to additional LE Clients and LESs. In the latter case, the local LES is responsible for relaying the LE_ARP request from the requesting LE Client to other local LE Clients and to all other LESs, and it is responsible for forwarding the LE_ARP response back to the requesting LE Client. Additional details of LES behaviour may be found in Section 2.4, which discusses communications between an LES and other service entities. Interactions between an LES and an LE Client are defined in the LUNI specifications[3],[5].

2.3.3 Broadcast and Unknown Servers

BUSs provide centralized forwarding for broadcast frames, and for frames with destinations that have not yet been resolved to ATM addresses. BUSs also share in the responsibility for the forwarding of sustained streams of multicast data.

Provisions are made in the LUNI specification [5] and this document for both an "intelligent" and "VCC-Mapped" BUS implementation. An intelligent BUS differs from a VCC-Mapped BUS by forwarding frames based on the target destination, where a VCC Mapped BUS blindly forwards all received frames to all LE Clients. As a typical example, an intelligent BUS may forward a frame destined for a registered MAC address to only the LE Client that registered the MAC address. A VCC-Mapped BUS would forward this frame to all LE Clients, which may result in most LE Clients discarding the frame.

Each BUS is assumed to be logically paired with an LES; there is no protocol defined for LES/BUS interactions. As with the LES, the BUS has a set of local LE Clients for which it is directly responsible. During initialization, each LE Client LE_ARPs for the broadcast MAC address, and the local LES returns the ATM address of its associated BUS. The LE Client then establishes a Default Multicast Send VCC to its BUS, which in turn establishes a Multicast Forward VCC back to the client. The LE Clients which have established a Multicast Send VCC to a particular BUS are the local LE Clients for that BUS. Equivalently, the LE Clients local to a BUS are the LE Clients local to the BUS's associated LES.

An LE Client sends broadcast frames to its BUS, which must forward the frames to all local LE Clients and to all other BUSs. All BUSs serving a single ELAN must be interconnected in a full-mesh of Multicast Forward VCCs that are used to forward data frames. An LE Client may also send frames to its BUS which are destined for unresolved unicast or multicast LAN destinations. Once the LE Client resolves LAN destination to an ATM address, the LE Client transmits frames on a more direct path to the destination (using Data Direct VCCs for unicast frames and Multicast Send VCCs for multicast frames). However, the LES may resolve a multicast address to the ATM address of the BUS, requiring the BUS to forward a stream of multicast data. Whether an LE Client sends sustained streams of multicast data to its BUS or to an SMS is at the discretion of the system administrator, LANE v2 supports both possibilities.

As described in the above paragraph, the BUS must forward all frames for the broadcast address, initial frames for unresolved unicast or multicast destination, and may have to forward all frames for certain multicast addresses. These frames may be received at the BUS from local LE Clients, other BUSs, or SMSs, and then be forwarded by the BUS to both local LE Clients and other BUSs. Additional details of BUS behaviour may be found in Sections 2.4.6,

2.4.7 and 5.7, which describe BUS interactions with other service entities. Interactions between a BUS and an LE Client are defined in the LUNI specifications [3],[5].

2.3.4 Selective Multicast Servers

SMSs are designed for the single purpose of efficiently forwarding multicast frames. SMSs may be used to offload much of the multicast processing from the BUSs, which also have to forward broadcast frames and frames for unresolved LAN destinations.

SMSs establish Cache Synchronization VCC(s) to neighbour LESs assigned by the LECS during configuration. This enables an SMS to synchronize its cache database with the LES(s) using SCSP.

LE Clients following the initial version of the LUNI specification [3] are referred to as LANEv1 Clients, while LE Clients adhering to the second version [5] are referred to as LANEv2 Clients. LANEv1 Clients send all multicast data to the local BUS without LE_ARPing for the multicast destination. LANEv2 Clients treat multicast transmissions similarly to unicast transmissions, sending multicast frames to the BUS over the Default Multicast Send VCC until the multicast destination is resolved to an ATM address. The local LES answers an LE_ARP for a multicast destination by supplying the ATM address of either the local BUS or an SMS. A LANEv2 LE Client then establishes a Multicast Send VCC to that ATM address (if one does not yet exist) over which the LE Client transmits data destined for the resolved multicast address. If the LE_ARP response resolves to the local BUS, the BUS may establish a Multicast Forward back to the LE Client. A LANEv2 LE Client may be assigned to an SMS as a sender when the local LES responds to the multicast LE_ARP. If the local LES provides an SMS ATM address in the LE_ARP response, then the LANEv2 LE Client transmits data for that multicast destination to that SMS.

A LANEv2 LE Client may be assigned to an SMS as a receiver when the LE Client registers to receive frames for a multicast address. The local LES assigns the LE Client to an SMS as a receiver by including the association of (multicast destination, SMS, LEC) in its registration database, which then gets propagated to all servers. When receiving this information via SCSP, an interested SMS adds the LE Client as a leaf on a Multicast Forward VCC. Hence, an LE Client is added as leaf on a “SMS” Multicast Forward VCC as a result of registering a multicast destination.

An SMS combines a subset of LES and BUS functionality into a single entity. Each SMS maintains a copy of the registration database, participating in the SCSP with the LESs and other SMSs. A local copy of the registration database permits the SMS to know which LE Clients registered for which multicast MAC addresses. In this sense, an SMS behaves like an LES. However, LE Clients cannot join or register with an SMS, nor will an SMS forward control frames. Thus, an SMS has all of the LES information, but is free from the LANE Control Communications associated with local LE Clients. Since an SMS processes only certain multicast frames, it has no need for unicast registration data. This permits certain SCSP optimizations for LNNI as discussed in Section 2.5.

An SMS also behaves like a BUS. An LE Client connects to an SMS via a Multicast Send VCC, and an SMS connects to an LE Client with a Multicast Forward VCC. An SMS forwards frames for certain multicast destinations. However, an SMS is not required to process LE_FLUSH frames, nor is it assumed to be logically associated with an LES. An LES may assign one if its Local LE Clients to an SMS as either a sender, receiver, or both, but only the paired LES may assign an LE Client to a BUS. Unlike a BUS, an SMS has independent sets of local senders and local receivers, so LE Clients may be connected to an SMS via a Multicast Forward VCC which are not connected to the SMS via a Multicast Send VCC. The number and distribution of SMSs in a network may be scaled to the required multicast bandwidth, independent of the number of LES/BUS pairs.

Additional details of SMS behaviour may be found in Sections 2.4.2, 2.4.4, and 2.4.7, which discuss SMS interactions with other LANE Service components. Interactions between an SMS and an LE Client are defined in the LUNI specifications [3],[5].

2.4 LNNI Communications

The multiple LANE Service entities serving an ELAN need to co-operate and communicate in order to provide a distributed and reliable LAN Emulation Service. The communications required for LNNI may be partitioned as follows:

a) Control Plane

- Configuration and Status Communications (Section 2.4.2)
- LANE Control Communications (Section 2.4.5)

b) Synchronization Plane

- LECS Synchronization (Section 2.4.3)
- LES-SMS Database Synchronization (Section 2.4.4)

c) Data Plane

- BUS Data Communications (Section 2.4.6)
- SMS Data Communications (Section 2.4.7)

The above communications refer to only the communications within the LANE Services. Communications to and from LE Clients are defined in the LUNI specifications [3],[5]. The following sections overview the various communications required for LNNI.

2.4.1 Full Mesh Connectivity

In several instances it is desirable to provide *full-mesh* connectivity between a set of servers. These instances include:

- LNNI Control Coordinate VCCs between LES
- LNNI Multicast Forward VCCs between BUSs
- LECS Synchronization VCCs between LECSs

Full-mesh connectivity implies that any member node of the mesh can send a message directly to any combination of other members of the mesh, and can receive a message from any member of the mesh, without that message being relayed by any other node. Some combination of point-to-point and point-to-multipoint VCCs must be used to create a full-mesh.

The basic sender and receiver requirements for implementing a full mesh are:

a) To transmit a message to another member of the mesh, a sender must have one or more point-to-point and/or point-to-multipoint VCCs to the other members of the mesh. The choice of how many VCCs are used, and how many copies of the message are sent on those VCCs, is at the discretion of the sending node, subject to the following constraints:
   - The message must reach at least all of the other nodes of the mesh to which it is addressed and
   - The message must not be sent more than once to any other node in the mesh.

b) In order to allow senders to conform to (a), a node in the mesh must not refuse a call SETUP from any other member of the mesh that meets the requirements imposed on the call SETUP parameters, themselves. In particular, it must allow any number of connections from another member of the mesh.

c) The protocols used over a full mesh must be such that they do not require a message to be delivered to one member, but not another member, of the mesh. That is, the sender must never be required to not send a particular message to another node.

d) As an optimization, point-to-point VCCs, if bi-directional, may be used in both directions. When a node realizes that it has more than one point-to-point VCC to another node, and that the VCCs are equivalent (in terms of QoS) to each other, then it uses the LUNI rules for duplicate Data Direct VCCs (Section 8.1.13 of [5]) to select one of those VCCs for all point-to-point transmissions to that other node. The "extra" VCC is not immediately released; it may be timed out and released if it is not used for some length of time.

Under these constraints, a wide variety of VCC implementations are possible. In particular, a sender may set up a single point-to-multipoint VCC to all other members of the mesh, and send all messages on that VCC. Alternatively, the sender may set up a point-to-point VCC to every other member of the mesh, and send one copy of a message on each VCC required to reach the intended nodes.

Note that these requirements say nothing about whether these VCCs connect to other nodes not in the mesh. For some protocols and some meshes, it may optimize VCC usage to allow off-mesh and on-mesh traffic to use the same point-to-multipoint VCC.

In Figure 2-2, an example four-node full mesh is illustrated. The four nodes use four different methods for connecting to the other nodes.

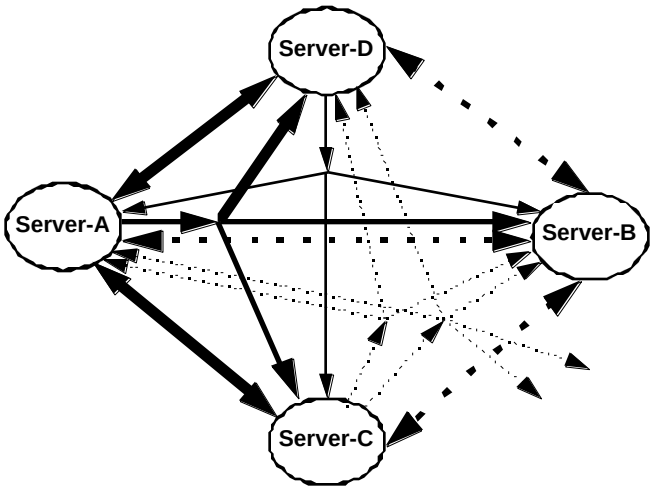

Figure 2-2: Example of Full Mesh Connectivity between Servers

Server-A (thick, solid lines) tries to minimize wasted bandwidth. It maintains both a point-to-multipoint VCC to the other nodes B, C and D, and also a point-to-point VCC to each other node. It uses the appropriate VCC to send each message to exactly the other nodes to which it needs to go. When it first was initialized, it also set up a point-to-point VCC to node B. However, the point-to-point VCC set up by B was preferable by the LUNI rules, so the A-B point-to-point VCC set up by A has timed out and been released. Both A and B use the point-to-point VCC setup by B.

Server-B (thick, dashed lines) depends on point-to-point connections to other nodes. It makes no point-to-multipoint VCCs. If it has a message to send to all three nodes A, C, and D, it must transmit three copies, one on each point-to-point VCC.

Server-C (thin, dotted lines) is trying to minimize multiple transmissions. It also has a requirement to send some messages both to destinations within the mesh, and outside the mesh. It therefore maintains two point-to-multipoint VCCs. One goes to just the other members of the mesh. It uses this VCC for traffic that goes to one or

more of the other nodes. The second goes to all other members of the mesh and to two off-mesh nodes. It uses this VCC for traffic that goes to both areas. With this scheme, C does not have to transmit twice the messages that are going both to in-mesh and off-mesh nodes.

Server-D (thin, solid lines) utilizes a single point-to-multipoint VCC to reach A, B and C. When sending, it ignores the point-to-point VCCs established to it by B and A. Every message it sends goes to all other nodes. If it has a message to send just to B, then C and D must know, by looking at the message, to ignore it.

There are many other possible configurations.

2.4.2 Configuration and Status Communications

LESs and SMSs obtain configuration information from an LECS over Configuration Direct VCCs. LECSs obtain the status of LESs and SMSs over the same connection. Figure 2-3 depicts typical Configuration Direct VCCs.

Each LES and SMS must establish a Configuration Direct VCC to an LECS and download their configuration. The configuration includes:

- the list of neighbouring LESs and SMSs to which the LES/SMS must connect and with which the LES/SMS must synchronize databases,
- a server ID (SID) uniquely identifying this server within the ELAN, and,
- for LESs, a range of LECIDs which the LES may assign to its local LE Clients.

Note that the list of neighbouring servers is not necessarily the complete list of SMSs and LESs serving the ELAN, but only those with which this particular LES or SMS must directly synchronize databases. Each server directly synchronizes its database only with a subset of the other servers, SCSP reliably propagates database synchronization across the set of servers. After synchronization, the complete list of LESs and SMSs serving the ELAN may be obtained via the registration database, which maintains a copy of all active LESs, SMSs, LE Clients, and registrations.

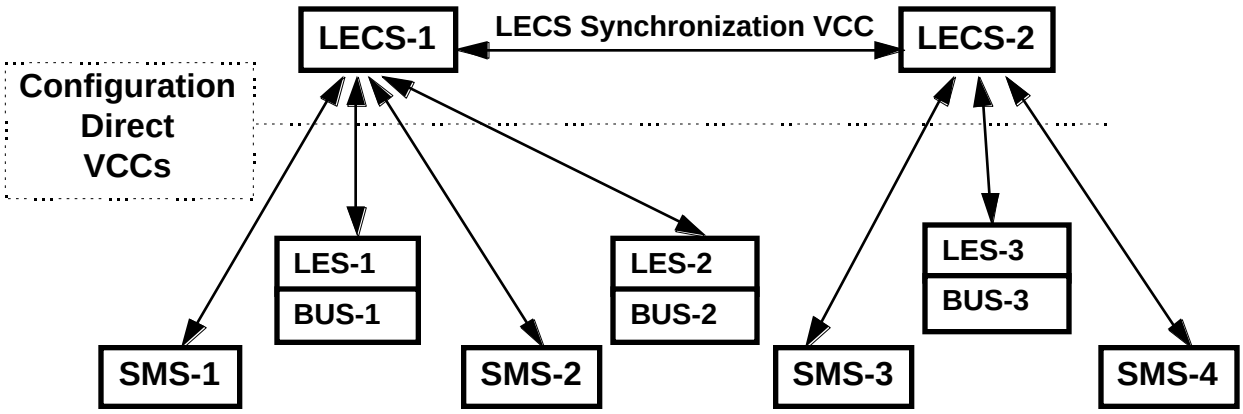

Figure 2-3 Configuration and Status Communications

Once an LES or SMS is operational, it must have established a Configuration Direct VCC to the LECS. This VCC is established using the same procedures for connecting to an LECS as defined in the LUNI specification [5]. While operational, LESs and SMSs send Keep-Alive-Notices (see Section 5.2) to the LECS over the Configuration Direct VCC. Each Keep-Alive-Notice contains information identifying a set of servers that are operational. The LECS uses this information so that LE Clients do not get assigned to non-operational LESs and to update the synchronization topology. The LECS sends a Keep-Alive-Response as an answer to each Keep-Alive-Notice, and the Keep-Alive-Response indicates the allowed interval before the next Keep-Alive-Notice must be received by the LECS. If the interval expires without the LECS receiving a new Keep-Alive-Notice identifying the server(s) the

LECS assumes that the server(s) is non-operational. The LECS will not assign an LE Client to a non-operational LES. Note that the LUNI specifications [3],[5] do not require that an LE Client maintains the Configuration Direct VCC, nor do LE Clients participate in the Keep-Alive protocol.

The Configuration Direct VCC from an LES or SMS uses the same signalling parameters as the Configuration Direct VCCs from an LE Client as defined in [5]. Co-located LANE entities may use the same VCC for their configuration and Keep-Alive-Notices. Multiple LANE components using the same Configuration Direct VCC may include multiple server identifications in a single Keep-Alive-Notice (as opposed to sending multiple Keep-Alive-Notices). For example, a device with 3 LESs and 4 SMSs may have a single Configuration Direct VCC to an LECS, and may send a single Keep-Alive-Notice to the LECS which contains information identifying all 7 servers as operational. Alternatively, the device may have 7 Configuration Direct VCCs to an LECS, and send 7 Keep-Alive-Notices to the LECS, each identifying a single server.

If a LES or SMS temporarily lose their configuration direct VCC, they may continue to maintain their connections to their peer servers and local LE Clients until their KeepAlive time expires. If a LESs or SMSs KeepAlive time expires, then it must terminate all peer server connections and return to the initial state. However, a LES can continue to serve the local LE Clients. When the LES is able to re-establish communication with the LECS, it must once again enter the configuration phase. In such an event, the LES may attempt to reuse the LECID range that was previously allocated to it by requesting the same LECID range in a future LE_CONFIGURE_REQUEST. During the configuration phase, the LES would not have any neighbours assigned since it is in the initial state. Hence no synchronization of the change is required. If an LECS receives the LECID range TLV in a LE_CONFIGURE_REQUEST from the LES, it should attempt to return a superset of the requested LECIDs to the LES.

2.4.3 LECS Synchronization

Since each LES or SMS connects to a single LECS, a particular LECS may not directly receive status from all service components. Thus, LECSs must exchange LES and SMS status information among themselves. In order to distribute this status information, all LECSs participating in an ELAN must maintains an LECS Synchronization VCC to all other LECSs in the network. This full mesh of connections between LECS can use a combination of point-to-point and point-to-multipoint connections as described in Section 2.4.1.

There are 2 ways in which an LECS may learn of the other LECSs in the network.

1. Each LECS may be configured with the ATM address of other LECSs in the network.
2. An LECS should dynamically determine the ATM addresses of other LECSs by using ILMI and querying the local ATM switch.

Upon determining that an incoming VCC is from another LECS, an LECS may attempt authentication to determine if Keep-Alive information should be shared with this neighbour. Valid types of authentication include, but are not limited to, no authentication (i.e. assume that neighbour is trusted and share information) and comparison against the configured/ILMI list of LECSs (i.e. only authenticate LECS connections if configured to do so). If duplicate VCCs are established between two LECSs, the rules for duplicate Data Direct VCCs (Section 8.1.13 of [5]) are used so that one of the VCCs ages out.

Once an LECS connects to another LECS, it periodically sends LECS Synchronization Messages on behalf of its locally attached servers. The Synchronization Message from the LECS is distinguished from a Keep-Alive from a server via the op-code of the control packet. The Synchronization Message from an LECS contains information identifying which servers are considered locally attached and operational by that LECS. Thus each LECS is informed, either directly or indirectly, of the status of all operational servers.

The LECS can cause a server to reinitiate configuration by sending the server a configuration trigger. Upon receipt of the trigger, the server must re-request its configuration from the LECS, and modify its configuration accordingly.

This mechanism is used, for example, for the LECS to inform a server that another server has become operational and that it should now attempt to connect to the other server.

2.4.4 LES-SMS Database Synchronization

LESs and SMSs use SCSP to synchronize their databases. As part of their configuration, a server receives a list of neighbour servers from the LECS. The server then establishes Cache Synchronization VCCs to their neighbours. Synchronization occurs only between neighbouring servers.

It is recommended that SCSP messages be exchanged over point-to-point Synchronization VCCs, although sending SCSP messages over point-to-multipoint VCCs is not prohibited. However, the use of point-to-multipoint dual function VCCs to carry both LANE Control and SCSP Control communications cannot result in an SMS receiving LANE Control frames

At the LECS, a set of servers for an ELAN may be configured. Each server may be configured with a set of neighbouring servers for the purpose of database synchronization. The LECS uses a combination of the configured neighbours and the active servers when returning a neighbour list to a server. In the most simple case, no servers are configured for the ELAN, and the LECS returns to each server the set of active servers for that ELAN (no configuration is required!). At the other extreme, an entire server topology (set of servers and their neighbours) is configured at the LECS and the configured information is returned, regardless of status. The LNNI protocols permit various grades of configuration versus plug-and-play under the control of a system administrator.

The LECS uses the following guidelines when returning a set of neighbour servers:

1. If a set of neighbours is configured, the configured neighbours are included in the returned neighbour list. A subset of the active servers may also be included.
2. If a set of neighbours is not configured, then a subset of the active servers is returned in the neighbour list.
3. In either case, the LECS may use configuration triggers to inform operational servers of a new server with which they should synchronize.

When an LES or SMS is becoming operational, or when it is informed of a new server with which it should synchronize, it establishes a Cache Synchronization VCC to its neighbour server(s). These VCCs are LLC-multiplexed and may be used for both SCSP (Cache Synchronization) and Control Coordinate (LE Control frame forwarding). A server's primary set of Cache Synchronization VCCs is initially established for the purpose of database synchronization, although they may also be used to carry LANE Control Communications. Through synchronization, a server may learn of more non-neighbour servers, and will establish additional Control Coordinate VCCs to these servers. These additional Control Coordinate VCCs are used only for only LANE Control Communications as discussed in Section 2.4.5. They do not carry SCSP Control Communications.

Each LES may setup any combination of point-to-point or point-to-multipoint Cache Synchronization VCCs for its Database Synchronization. LESs must accept Cache Synchronization VCCs from other LESs when at all possible. If duplicate VCCs are established between two LESs, then the rules for duplicate Data Direct VCCs (Section 8.1.13 of [5]) must be used so that one of the VCCs ages out. SCSP messages are most likely exchanged over point-to-point Cache Synchronization VCCs, although sending SCSP messages over point-to-multipoint VCCs is not prohibited. However, the use of a point-to-multipoint Cache Synchronization VCC to carry both LANE Control and SCSP Control communications cannot result in an SMS receiving LANE control frames. Refer to Figure 2-4 for an example of SCSP Control communications over Cache Synchronization VCCs.

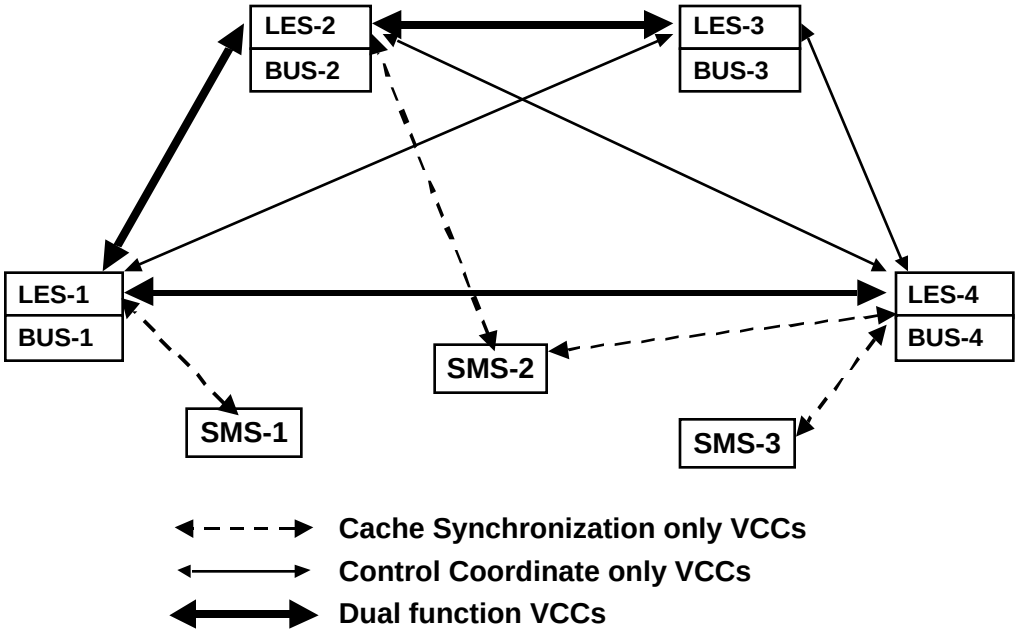

Figure 2-4 Cache Synchronization VCCs for SCSP Communications

After a Cache Synchronization VCC is established, the server synchronizes its database with its neighbour. Each server maintains a complete and up-to-date copy of the LNNI database, which includes information on all LESs, BUSs, SMSs, and LE Clients in the ELAN, and the registrations associated with all clients and servers. This information is used to answer and forward control and data frames in the appropriate manner. Since SMSs do not require unicast registration information, the following SCSP optimizations are permitted (when configured).

- SMSs may acknowledge and discard any received unicast registration information.
- SMSs may acknowledge and discard any received multicast registration information that does not pertain to the MAC addresses served by that SMS.
- LESs are not required to flood unicast registration information to neighbour SMSs.
- SMSs are not required to flood information received from one neighbour to their other neighbours.

When operating in this mode, it is the responsibility of the system administrator to ensure that removal of the SMSs would not result in partitioning of the SCSP topology into “islands”.

As one example SCSP topology, all servers may be connected in a mesh. The LECS would inform all servers, as a new server became operational, thus continuing the full mesh connectivity. As another example, the LESs may be connected in a full mesh, with the SMSs operating with connections(s) into the LES mesh. All LESs and some SMSs would be notified by the LECS when a new LES became operational. When an SMS became operational, the LECS would only notify those LESs with which it must synchronize. Many other topologies are also permitted, and a server may not assume anything about the topology except for the optimizations previously mentioned.

Each entry in the synchronized database is tagged with a lifetime for the information. Servers entering an ELAN include a server advertisement in their registration database. SCSP reliably propagates the information to all other servers that are then informed of the new server. Similarly, when LE Clients join, register, unregister, or terminate, the local LES includes the information in its local database. This information is transformed into an appropriate client advertisement, and SCSP ensures that all other servers learn the updated client information.

Server information is originated with a shorter lifetime than client information. In this way, server failures are more quickly noticed than client failures. When server information ages out of the database, all client information originated by that server is automatically purged.

The registration database synchronized by SCSP includes the following information for each server:

Server information

ATM address, SID

BUS information (for LESs)

ATM address Local LE Client information (for LESs)

LECID, primary ATM address, LE_JOIN response flags

Local LE Client registration information (for LESs)

Registrations including registered TLVs, assignments to SMS as receiver

Multicast information (for SMSs)

Served multicast addresses and serving ATM addresses

SCSP operation and general frame formats are defined in [4]. The frame formats for LNNI SCSP are defined in Section 4.1.1, and LNNI specific actions associated with the SCSP protocol are defined in Section 5.4.2.

2.4.5 LANE Control Communications

Each LES is responsible for distributing LE_ARP requests for unregistered destinations from local LE Clients to local LE Clients and to other LESs. LESs must also forward LE_ARP responses back to the originator. Additionally, LESs must be able to forward LE_FLUSH responses and LE_TOPOLOGY requests to the correct destination(s). Other LANE control frames (such as LE_JOIN, LE_REGISTER, LE_UNREGISTER, and LE_VERIFY frames) are never transmitted between servers.

Each LES may setup any combination of point-to-point or point-to-multipoint Control Coordinate VCCs to accomplish this distribution, and it may also use the Cache Synchronization VCCs originally established for SCSP communications. Figure 2-5 illustrates a typical set of Control Coordinate VCCs. LESs must accept the Control Coordinate VCCs from other LESs when at all possible. If duplicate VCCs are established between two LESs, then the rules for duplicate Data Direct VCCs (Section 8.1.13 of [5]) must be used so that one of the VCCs ages out. When deciding whether to use point-to-point VCCs or point-to-multipoint VCCs to interconnect LNNI Control and Synchronization VCCs, it is important to keep in mind the following:

1. It is recommended that SCSP messages be exchanged over point-to-point VCCs, although sending SCSP messages over point-to-multipoint VCCs is not prohibited. The use of point-to-multipoint VCCs for SCSP frames may result in unnecessary replication of frames during synchronization activities. The LNNI requires that individually addressed (**ServerId**) CSAs be sent to all interconnected servers. If these servers are interconnected via a point-to-multipoint VCC, then all servers will receive SCSP messages intended for all the other servers participating in the synchronization topology. The receiving servers will need to discard these various SCSP messages not directed to its SeverGroupId and ServerId. Additionally, if any receiving server fails to respond to the SCSP message, the sending server must resend the SCSP message,

again resulting in all interconnected servers receiving a duplicate CSU request message intended for a single server.

2. Because Control Coordinate and Cache Synchronization VCCs are LLC multiplexed, the same VCC can carry both types of traffic. However, the use of a point-to-multipoint VCC to carry both LANE Control and SCSP Control communications must not result in an SMS receiving LANE Control frames.

3. Point-to-multipoint VCCs are often selected in an effort to reduce frame replication within a particular server's implementation. Implementing the LNNI control plane with point-to-multipoint VCCs is recommended as it reduces the need to replicate a control frame for each receiving server in the control plane (frame replication would be necessary in a network of servers interconnected via point-to-point VCs). It should be noted, however, that the use of point-to-multipoint VCs in the LNNI does not eliminate the need for the LES to replicate frames under all circumstances. For example, LE_ARP frames may be transmitted on both Control Distribute VCs to locally attached LE Clients and on Control Coordinate VCCs to all other LESs, and as such the LE_ARP frame must be replicated in order to adequately satisfy this requirement.

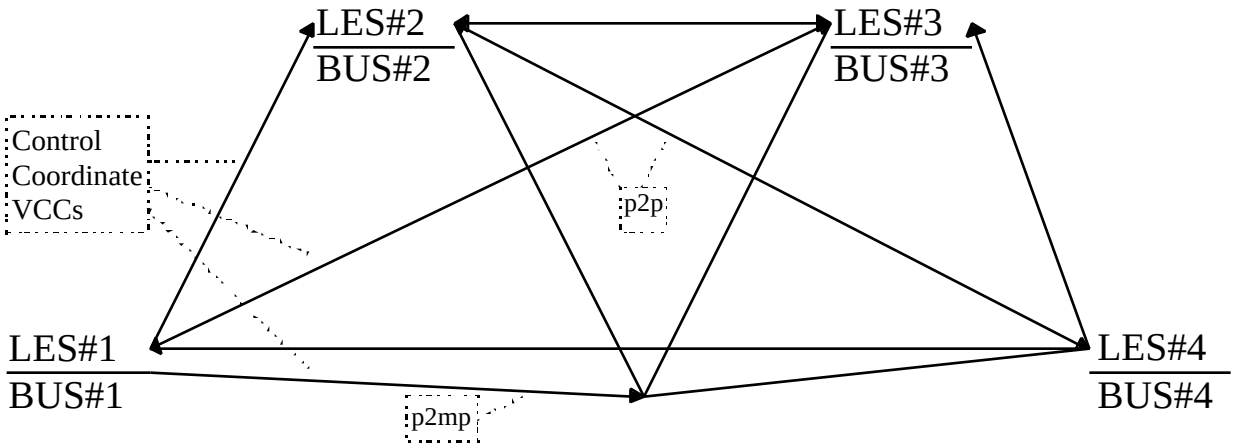

Figure 2-5 Control Coordinate VCCs for Control Communications

LESs never forward control frames from an LES to any other LES, for it is the originating LESs responsibility to distribute the frames to all required servers. Logically, the collection of LESs is a mesh, with each LES directly communicating to all other LESs.

The Control Coordinate VCCs are LLC-multiplexed VCCs and thus may also be used to carry other protocols. Servers may use any LLC-multiplexed VCC between the correct ATM addresses as Control Coordinate VCCs. The correct endpoints of the Control Coordinate VCCs may be returned in the configure response, or may be distributed as part of the registration database.

2.4.6 BUS Data Communications

Each BUS is assumed to be logically paired with an LES. No protocol is defined between a paired LES and BUS, although the BUS is assumed to have access to the registration database maintained by the LES. BUSs are responsible for forwarding broadcast, some multicast, and initial unicast data to and from their local LE Clients.

The LES database includes the ATM address of all BUSs. Each BUS is assumed to have access to this information. The collection of BUSs is logically a mesh, with each BUS directly connected to all others.

Each BUS may setup any combination of point-to-point or point-to-multipoint VCCs to accomplish its forwarding objectives. The VCCs established between BUSs are referred to as Multicast Forward VCCs. Figure 5 depicts a typical set of Multicast Forward VCCs for BUS Data Communications. BUSs must accept Multicast Forward

VCCs from other BUSs when at all possible. If duplicate Multicast Forward VCCs are established between two BUSs, then the rules for duplicate Data Direct VCCs (Section 8.1.13 of [5] must be used so that one of the VCCs ages out. The use of point-to-multipoint VCCs is constrained only by the fact the LUNIv1 LE Clients need not accept multiple Multicast Forward VCCs. This constraint may cause some frames to be replicated (sent on multiple VCCs) by the BUS.

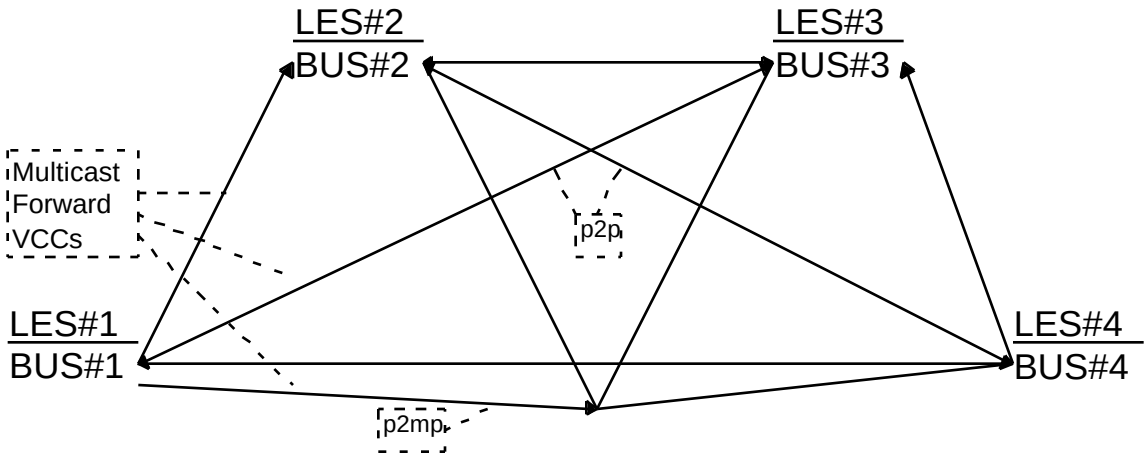

Figure 2-6 Multicast Forward VCCs for Data Communications

Multicast Forward VCCs are signalled as specified in the LUNI specification [5], and, in particular, are not LLC-multiplexed (unlike Control Coordinate VCCs). A single point-to-multipoint Multicast Forward VCC may have both BUSs and LE Clients as leafs. Such VCCs may be used to improve forwarding throughput at the cost of additional VCCs.

A BUS forwards data and LE_FLUSH requests to LE Clients and to other BUSs.

A BUS must never forward a frame received from a BUS to another BUS, for it is the originating BUS's responsibility to distribute the frame to all required BUSs. The collection of BUSs serving an ELAN is meshed.

A BUS may receive frames from an LE Client, another BUS, or an SMS. Through unspecified communications with the LES, the BUS knows the ATM address of LE Clients, BUSs, and SMSs, so it can classify an incoming VCC as belonging to one of these groups. BUS frame handling differs for frames from LE Clients, BUSs, or SMSs.

A BUS must forward broadcast frames from a local LE Client to all local LE Clients and to all other BUSs. Unicast frames from a local LE Client must be forwarded in at least the direction of the destination. This may be to a local LE Client or to another BUS.

The BUS's handling of multicast frames differs depending on how the Selective Multicast Flag was set in the LE_JOIN response to an LE Client. An LE Client for which the Selective Multicast Flag was set in the LE_JOIN response is referred to as an SMF-LE Client. Otherwise, the LE Client is referred to as non-SMF LE Client. A LUNI v1 LE Client is a non-SMF LE Client by definition. Whether an LE Client is an SMF LE Client or non-SMF LE Client is included in the registration database.

The following example illustrates typical BUS forwarding behaviour. A multicast frame received from a local LE Client must be forwarded to all local non-SMF LE Clients, local SMF LE Clients that registered for the multicast destination, and all other BUSs. A multicast frame received from a BUS must be forwarded to all local non-SMF LE Clients and to local SMF LE Clients registered for the multicast destination. A multicast frame received from an SMS must be forwarded only to local non-SMF LE Clients.

2.4.7 SMS Data Communications

A Selective Multicast Server processes frames from LUNI v2 LE Clients destined for a selected set of multicast MAC addresses. Every SMS (and LES) obtains a complete copy of the registration database for the entire ELAN via SCSP, so every SMS knows of every other SMS and BUS. An LE Client cannot distinguish a Multicast Forward established by a BUS from a Multicast Forward established by an SMS.

When a LUNI v2 LE Client LE_ARPs for a multicast address, the LES should assign the client to an SMS as a sender if an SMS is available for that destination. An SMS is configured with which multicast MAC addresses it is to serve, and the complete set of SMSs and the MAC addresses served by each is in the registration database. The LES may use this information to decide to which SMS this LE Client should be assigned as a sender. The algorithm used by the LES to make this assignment is not specified. The assignment is complete when the local LES returns the ATM address of an SMS in the LE_ARP response. If no SMS for the requested multicast address is registered, then the LES must return a BUS's ATM address in the LE_ARP response. At any later point, the LES may move the LE Client from the BUS to a particular SMS by issuing a LE_NARP request to that client.

In order to receive multicast frames, SMF LE Clients must register multicast addresses with the LES. At registration, the LES must determine to which SMS this receiver should be assigned (if any are available for the requested multicast MAC address).

All SMSs are notified that the LE Client has been assigned to a particular SMS as a receiver via the registration database. The LE Client is notified of the SMS to which it has been assigned a receiver when the SMS establishes a Multicast Forward to the LE Client.

An ELAN, and hence all the ELAN's SMSs, may operate in either distributed or stand-alone mode, as determined by the network administrator. In stand-alone mode, each SMS maintains a point-to-multipoint Multicast Forward VCC to all receiver LE Clients in the ELAN for a particular multicast address. The SMS is responsible for transmitting multicast frames directly to all SMF LE Clients registered for the MAC, and to all BUSs. The BUSs, in turn, forward the frames to all non-SMF LE Clients.

In distributed mode, each SMS maintains two sets of point-to-multipoint Multicast Forward VCCs. One set of Multicast Forward VCC(s) connects to all of the SMS's assigned SMF LE Client receivers for this MAC, to all BUSs, and to all other SMSs. The other set of Multicast Forward VCC(s) connects to all of the SMS's assigned SMF LE Client receivers for this MAC, but not to any BUS nor to any other SMS.

The SMS sends any multicast frame received from one of its own sender LE Clients to:

- all of its own receiver LE Clients
- the other SMSs (which in turn forward to their local SMF LE Clients for that MAC), and
- to all BUSs (which forward to their local non-SMF LE Clients).

The SMS sends any multicast frame received from another SMS only to its own receiver LE Clients.

Since the SMS knows at the time each incoming connection is established whether or not it is from another SMS, it can associate the incoming connection to the proper Multicast Forward VCC(s).

Note that no frame replication is required at the SMS in either mode. Stand-alone mode results in a lower multicast latency, since a multicast frame need be collected and re-transmitted in only one SMS. Distributed mode results in a lower number of VCCs if three or more SMSs are present. The total cost of establishing SMS Multicast Forward VCCs is expected to be lower for the distributed mode, because almost all of the endpoints are local to the SMS, and not distributed all over the ELAN.

Stand-alone mode for SMSs is depicted in Figure 2-7, and distributed mode in Figure 2-8. In both figures, SMF LE Clients (LEC) 1 and 2 have been assigned as receivers to SMS#1, and SMF LE Client 3 to SMS#2. Also in both figures, two BUSs have been shown. All three SMF LE Clients and Non-SMF LE Clients #1 and #2 are assigned to BUS#1, and only the Non-SMF LE Client#3 is assigned to BUS#2.

Note that Multicast Send VCCs from each LE Client are not shown. Also, note that the forwarding rules for a VCC-Mapped BUS or Intelligent-BUS must be complied with to avoid duplicate packets.

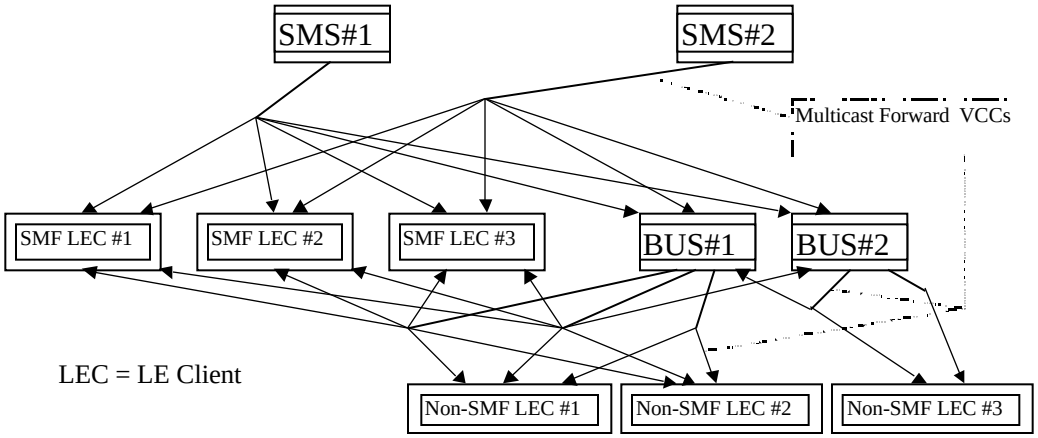

Figure 2-7: Stand-Alone Mode SMSs

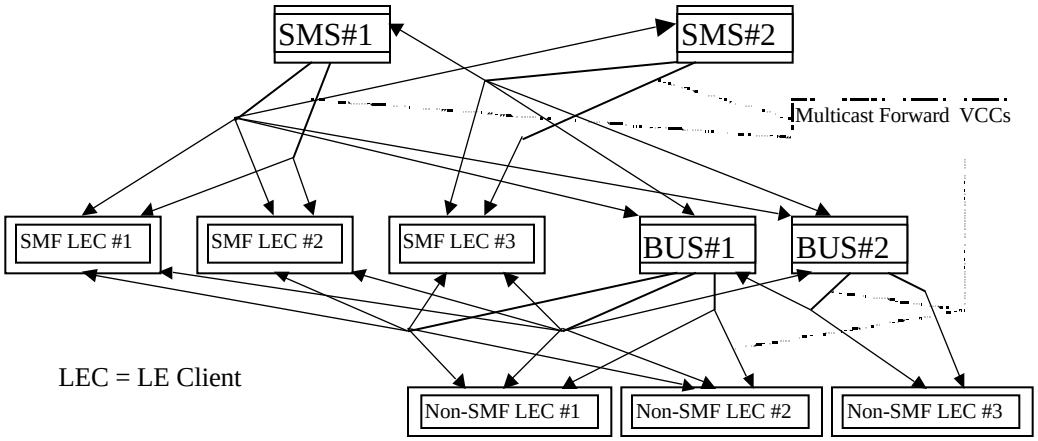

Figure 2-8 Distributed Mode SMSs

In both modes, BUS#1, which is assigned all three SMF LE Clients as well as Non-SMF LE Clients #1 and #2, needs three Multicast Forward VCCs in order to avoid duplicating frames. One VCC goes to all of its LE Clients, and only its LE Clients. This VCC is used only to forward frames received from the other BUS. The second VCC goes to all of its LE Clients, and to the other BUS, as well. This VCC is used to forward frames received from any of its own LE Clients, whether SMF or Non-SMF. The third VCC goes to only its Non-SMF LE Clients. This VCC is used only for frames received from an SMS. BUS#2 needs only two VCCs; the first and third types are equivalent to it, because it has no SMF LE Clients. The BUS's Multicast Send VCCs are not shown.

2.5 SCSP Overview

2.5.1 Server Synchronization Overview

SCSP may be considered to be composed of two sublayers: the protocol independent layer and the protocol dependent layer. The protocol independent sublayer consists of the Hello protocol, the Cache Alignment protocol and the part of the Client State Update (CSU) protocol that is independent of the type of server being synchronized. The Hello protocol maintains SCSP level connectivity between neighbouring servers. The Cache Alignment (CA) protocol allows a server to synchronize its entire cache with the cache of a neighbour server. The protocol dependent layer contains part of the CSU protocol that is specific to the type of server being synchronized. The Client State Update protocol is used by a server to flood the change in state of one or more of its cache entries throughout the group of aligned servers. The CSU protocol may only be used on a link to another server after the initial Cache Alignment has occurred between the servers on each side of that link.

When an LES or SMS establishes a Cache Synchronization VCC to a neighbouring server, it initiates the Hello Protocol with that neighbour. Control Communications may use the VCC before the Hello Protocol has recognized both servers as being operational for SCSP. Once the neighbours recognize each other via the Hello Protocol, the LES or SMS synchronizes its current registration database with that of its neighbour using the Cache Alignment protocol. After the Cache Alignment has completed, both servers have the same information in their registration database. The use of the Cache State Update in LNNI is covered in the following section.

2.5.2 Basic LNNI SCSP Operation

This section defines the protocol specific layer procedures of SCSP when that specific protocol is LNNI. Thus, the following contains procedures relative to the use of CSU Request and CSU Reply messages. CSU messages contain zero or more Client State Advertisement records (CSA records). CSA records embody the change in state information necessary to update a given cache entry. For LNNI, a CSA record may indicate that a new server has become operational, or a MAC address has been unregistered, etc. For example, if an LE Client leaves the ELAN, its local LES purges information from the LNNI database by sending an LSA for that client with a lifetime of zero; all other LES and SMS receiving this CSA delete information on this client from their local cache. A CSA record is placed in a CSU Request and is sent to another server to advertise the change in state of this shared cache entry. A server acknowledges the receipt of a CSA record in a CSU Request by placing the CSA record into a CSU Reply and sending that reply to the server from which the CSU Request was received.

A CSA record contains the following fields, among others: an Originator ID (OID), a CSA Cache Key (CSA CK), and a CSA Sequence Number (CSA SN). The server that creates a CSA assigns the OID (itself), a CSA CK, and a CSA SN to the CSA. These fields stay with the CSA as it moves from server to server and are part of the server cache entry when the advertisement is deposited in a server. These fields identify a unique instance of a cache entry which is shared throughout the LNNI. The OID and CSA CK uniquely identify the CSA, and the CSA SN determines which instance of the CSA is more recent. Whenever a server issues an updated version of a given CSA, it must increment the CSA SN for that CSA.

2.5.3 CSU Processing

There are only a few simple rules from processing CSU messages. The following algorithm explains how Cache State Advertisements are propagated between servers, and what happens to cache entries when an advertisement is received. The rudimentary procedures are as follows:

a) If a server receives a valid CSA in CSU Request then:

- the server updates its cache appropriately
- the server ACKs the CSA by sending a CSU Reply which contains the CSA to the requesting neighbour

- the server sends a CSU Request containing the CSA to each neighbouring server except the server from which the CSA was originally received

b) Else, if a server detects an invalid CSA in a CSU Request then it must not change its cache based on the CSA.

If a valid CSA is received then the server's registration database is updated only if the CSA SN in the CSA is larger than the CSA SN already associated with the matching entry being updated. Otherwise the CSA acknowledged and dropped without further flooding. An entry is matching if and only if the OID (SID) and Cache Key are the same in both the CSA and the cache entry in the registration database.

3 Connection Management Services [Normative]

The signalling requirements of the LAN Emulation LUNI are detailed in Section 3.3 of the LUNI specification [5]. Those requirements, specifically the rules for addressable components, the mandatory information elements, and the information element contents, apply to the entities and communications of this specification with exceptions as detailed in this section.

3.1 LLC-Multiplexed Virtual Channel Connections

This specification requires the use of both LLC-multiplexed and non-LLC-multiplexed VCCs. It is assumed that LLC-multiplexed VCCs perform their own connection management, including timeouts, multiplexing, etc. Multiple logical information flows may exist over a single LLC-multiplexed VCC.

3.2 Addressable Components

LE Service entities use different types of VCCs for LNNI communications. Basic procedures for distinguishing different types of VCCs are described in Section 3.3 of the LUNIv2 specification [5]. The following requirements are in addition to those procedures.

VCC types are distinguishable by the BLLI code values. The BLLI values relevant to LNNI and the associated VCCs are listed in Table 3-1.

BLLI PID Value	VCC Types
BLLI Layer 3, PID X"0001"	Configuration Direct, Control Direct, Control Distribute, LECS Synchronization
BLLI Layer 3, PID X"0004"	Ethernet Multicast Send, Ethernet Multicast Forward
BLLI Layer 3, PID X"0005"	Token Ring Multicast Send, Token Ring Multicast Forward
BLLI L2, Value 12	LLC-Multiplexed Data Direct, Control Coordinate, Cache Synchronization

Table 3-1 BLLI Values and the Associated VCCs

BLLI values and the BLLI Information Element are further discussed in Section 3.3.2.

In certain cases, multiple VCC types use the same BLLI signalling element. In such cases, either additional information is needed to distinguish the VCC types, or distinguishing the VCC types is not necessary.

VCCs using BLLI L3, PID X"0001"
An LES may assume that an incoming VCC using this BLLI IE is a Control Direct. An LECS may assume that an incoming point-to-point VCC using this BLLI is both a Configuration Direct VCC and an LECS Synchronization VCC. It is not necessary to distinguish between these two types of VCCs, the LECS processes both configuration and keep-alive frames over such a VCC. An incoming point-to-multipoint VCC to a LECS can only be a LECS Synchronization VCC.

VCCs using BLLI L3, PID X"0004" (or X"0005")
An Ethernet (Token Ring) BUS or SMS must be able to distinguish an incoming VCC using BLLI PID X"0004" (X"0005") as either originating from an LE Client, a BUS, or an SMS. The BUS must examine the calling ATM

address of the incoming VCC. If the calling ATM address of the VCC is not in the LNNI database, then the VCC must be rejected. Otherwise, the caller can be classified as either a LE Client, BUS, or SMS based on its ATM address.

3.3 SETUP and ADD_PARTY Message Contents

The information elements included in SETUP and ADD_PARTY signalling messages are exactly as specified in the LUNI document with the following exceptions.

3.3.1 ATM Adaption Layer (AAL) Parameters

The AAL parameters included in a SETUP or ADD_PARTY message for LNNI connection are exactly as specified in Section 3.3 of the LUNI specification [5] with the following exception.

- Forward Maximum CPCS-CPU Size. Exactly as specified in Section 3.3 of the LUNI specification [5] for Multicast Forward and Configuration Direct VCCs. Must be 1516 for LECS Synchronization VCCs, and at least 1516 for Cache Synchronization and Control Coordinate VCCs. The signalled value for LLC-multiplexed connections may be higher because this VCC may be shared by multiple protocols.

3.3.2 Broadband Lower Layer Information (BLLI)

The BLLI parameters included in a SETUP or ADD_PARTY message for LNNI connection are exactly as specified in Section 3.3 of the LUNI specification [5] with the following expansion. The mapping between types of LNNI VCCs and BLLI values is given in Table 3-1. Procedures for distinguishing the type of an incoming VCC are described in Section 3.2.

4 LNNI Frame Formats [Normative]

4.1 LNNI Control Frame

The Control Coordinate VCCs are all LLC encapsulated. The frame format for SCSP is defined in [4]. The frame format for LE Control is shown below:

Table 4-1. LNNI Protocol Control Frame

0	LLC = X"AAAA03"	OUI
4	OUI	FRAME-TYPE
8	ELAN-ID	
12	LANE control frame (LE_ARP, etc.)	

OUI = X"00A03E" which indicates ATM Forum. FRAME-TYPE = X"000F".

4.1.1 Control Frame Opcodes

Table 4-2. OP-CODE Summary

OP-CODE Value	OP-CODE Function
X"000b"	LNNI_CONFIGURE_TRIGGER
X"000c"	LNNI_LECS_SYNC_REQUEST
X"000d"	LNNI_KEEP_ALIVE_REQUEST
X"010d"	LNNI_KEEP_ALIVE_RESPONSE
X"000e"	LNNI_VALIDATE_REQUEST
X"010e"	LNNI_VALIDATE_RESPONSE

Additional OP-CODE Values for Control Frames can be found in Table 19 of the LUNI specification [5].

4.2 SCSP Packet Formats

LE Client information must be synchronized across all LESs and SMSs within an ELAN. The LANE Service uses SCSP for this purpose. Although SCSP solves the generalized problem of database synchronization/replication in a protocol independent way, there are protocol specific extensions that are necessary in addition to SCSP. This section defines LNNI specific extensions to SCSP.

4.2.1 Mandatory Common Part Fields

SCSP requires the definition of common part fields for LNNI.

4.2.1.1 Protocol ID

An SCSP packet contains a Protocol ID that identifies the protocol that the packet is associated with, in this case LNNI. The Protocol ID is a 16-bit field in the mandatory common part of the SCSP packet. The Protocol ID value for LNNI = 5 as defined in [4].

A LANE Service component MUST ignore any SCSP packet in which the Protocol ID is not encoded as LNNI.

4.2.1.2 Sequence Number

When a server originates or refreshes a modified CSA it must increment the Sequence Number for that CSA. A server must not increment the Sequence Number of a CSA if it has not changed. A server MUST increment the sequence number when it refreshes a CSA with its neighbor (ie when it is trying to extend the lifetime of the information).

4.2.1.3 Server Group ID

An SCSP packet contains a Server Group ID (SGID) which identifies the Server Group (SG) that the packet is associated with. All LANE Service components serving the same ELAN belong to the same Server Group[3]. For LNNI, the Server Group ID is equal to the **ServerGroupId**. The Server Group ID is a sixteen-bit field in the mandatory common part of the SCSP packet and is defined in [4].

The **ServerGroupId** is a configurable parameter for each LANE service component.

A LANE Service component MUST ignore any SCSP packet in which the Server Group ID is not equal to its **ServerGroupId**.

4.2.1.4 Sender ID

An SCSP packet contains a Sender ID field which identifies the transmitting server of an SCSP packet. The Sender ID is a variable length field in the mandatory common part of the SCSP packet defined in [4].

For LNNI the value of the Sender ID is the value of the **ServerId** (LANE Service component **ServerId**). The Sender ID Length is 2 bytes.

A LANE Service component MUST set the Sender ID field to **ServerId** and the Sender ID Len field to 2 prior to transmitting an SCSP packet.

4.2.1.5 Receiver ID

An SCSP packet contains a Receiver ID field which identifies the intended destination server of an SCSP packet. The Receiver ID is a variable length field in the mandatory common part of the SCSP packet defined in [4].

For LNNI the value of the Receiver ID is the **ServerId** corresponding to one of the servers listed in the SynchronizationPeerServerList. The Receiver ID Length is 2 bytes.

ServerId values which correspond with synchronization peers are obtained via the SCSP Hello protocol[4].

[3] Each **ServerGroupID** is uniquely mapped to a single ELAN ID. The ELAN ID can not double as the Server Group ID because the LUNI Specification [5] and RFC 2334 define the sizes of each field to be four octets and two octets, respectively.

A LANE Service component MUST set the Receiver ID field to the **ServerId** corresponding to one of the neighbours listed in the **SynchronizationPeerServerList** and set the Receiver ID Len field to 2 prior to transmitting an SCSP packet.

4.2.1.6_Originator ID

Every CSA record contained in an SCSP packet contains an Originator ID (OID) which identifies the LANE Service component advertising the associated CSA record and is obtained through configuration. For LNNI, the OID is 4 octets in length and is equivalent to 2 zero octets followed by the 2 octet **ServerId**.

A LANE Service component MUST set the Originator ID field of any CSA it originates to its **ServerId**.

4.2.2 Client/Server LNNI Specific Part of a Cache Entry

CSA Records in SCSP contain a "Client/Server Protocol Specific Part" which contains the protocol specific information for a given server's cache entry. Note that for each CSA, the **ServerId** of the originator of the CSA may be obtained from the originator-id of the CSA. Fields which are indicated as “must be zero,” must be zeroed on transmit and ignored on receipt.

4.2.2.1_Cache Key

A CSA record contains a Cache Key Length and Cache Key. The Cache Key is set to a unique value that identifies the data to the originating server, the length of the Cache Key is 4 bytes.

An LE Server MUST set the Cache Key field to a value which uniquely identifies the piece of data being synchronized and set the Cache Key Length field to 4 bytes prior to transmitting the SCSP packet.

The Cache Key must uniquely identify the CSA to the originating server. Certain fields in the CSA can change without requiring a new Cache Key, but other fields must not change without changing the Cache Key. The fields which cannot change in a CSA without requiring a change to the Cache Key are marked in Sections 4.2.2.3 through 4.2.2.8

4.2.2.2_LNNI CSA Identifier Values

LNNI defines the following CSA types.

1. LES Refresh. Originated by an LES to indicate that the LES is expected to be operational for the given lifetime.
2. SMS Refresh. Originated by an SMS to indicate that the SMS is expected to be operational for the given lifetime.
3. LE Client Join. Originated by an LES to indicate that an LE Client has joined locally and is expected to be operational for the given lifetime.
4. LE Client Unicast Registration. Originated by an LES to indicate that an LE Client has registered a particular unicast LAN destination, and that registration is expected to be valid for the given lifetime.
5. LE Client Multicast Registration. Originated by an LES to indicate that an LE Client has registered a particular multicast LAN destination, and that registration is expected to be valid for the given lifetime
6. SMS Multicast Registration. Originated by an SMS to indicate that the SMS is serving the particular MAC address and is expected to do so for the given lifetime.

4.2.2.3_LES Refresh CSA Record Format

Table 4-3. LES Refresh CSA Record Format

0	CSA Identifier = 1	Number of TLVs	Operational Status
4	Remaining Lifetime		
8	LES ATM Address		
28	BUS ATM Address		
48	TLV Type		
52	TLV Length	TLV Value	
etc..			

CSA Identifier - Identifies an LNNI CSA type. Equals 1 for LES Refresh CSAs

Operational Status - Identifies whether the LES is resource starved (e.g. memory or VC). Equals 1 if starved else 0.

Remaining Lifetime - The number of seconds before the CSA is considered invalid.

LES ATM Address[4] - The LLC-multiplexed ATM address of the advertising LE Server (**ServerMuxedAtmAddress**), used by other servers to build the LE Server Control Plane topology (full mesh of Control Coordinate VCCs). The **ServerId** of this LES may be obtained from the Originator ID of the SCSP packet.

BUS ATM Address[5] - ATM address of the BUS used to build a full mesh of Multicast Forward VCCs.

The LEC-ID Range TLV(s) returned in the latest configuration response should be included.

4.2.2.4_SMS Refresh CSA Record Format

Table 4-4. SMS Refresh CSA Record Format

0	CSA Identifier = 2	Number of TLVs	Operational Status
4	Remaining Lifetime		
8	TLV Type		
12	TLV Length	TLV Value	
etc..			

CSA Identifier - identifies an LNNI CSA type. Equals 2 for SMS Refresh CSA

Operational Status - Identifies whether the LES is resource starved (e.g. memory or VC). Equals 1 if starved else 0.

[4] Cannot change without changing Cache Key.
[5] Cannot change without changing Cache Key.

Remaining Lifetime - The number of seconds before the CSA is considered invalid.

The **ServerId** of the SMS being refreshed may be obtained from the OriginatorID of the CSA.

There are no TLVs currently defined for this CSA.

4.2.2.5 LE Client Join Record

Table 4-5. LE Client Join Record

0	CSA Identifier = 3	Number of TLVs		Must be Zero
4	Remaining Lifetime			
8	LE Client ID		Response Flags	Must be Zero
12	LE Client Primary ATM Address			
32	TLV Type			
36	TLV Length	TLV Value		
etc..				

CSA Identifier - Identifies an LNNI CSA type. Equal to 3 for an LE Client Join CSA.

Remaining Lifetime - The number of seconds before the CSA is considered invalid.

LE Client ID[6] - LECID of the registering LE Client

Number of TLVs - the number of TLVs found in this CSA

Response Flags[7] – the flag field of the JOIN_RESPONSE sent to the LE Client.

LE Client Primary ATM Address[8] -Primary ATM address of the LE Client, used by LE Servers for duplicate ATM address detection

TLV Type - Type of TLV as defined in the LUNI specification [5]. All TLVs that were included in the LE_JOIN_REQUEST MUST be included in this record.

TLV Length - Length of the TLV Value.

TLV Value - Value of TLV.

Note that the LE Client Join CSA has a variable length due to the variable number of TLVs.

[6] Cannot change without changing Cache Key.
[7] Cannot change without changing Cache Key.
[8] Cannot change without changing Cache Key.

4.2.2.6 LE Client Unicast Registration CSA Record Format

Table 4-6. LE Client Unicast Registration CSA Record Format

0	CSA Identifier = 4	Number of TLVs	Must be Zero
4	Remaining Lifetime		
8	LE Client ID		Must be Zero
12	Registered LAN Destination		
20	LE Client ATM Address		
40	TLV Type		
44	TLV Length	TLV Value	
etc..			

CSA Identifier - identifies an LNNI CSA type. Equal to 3 for Unicast Registration CSA.

Remaining Lifetime - The number of seconds before the CSA is considered invalid.

LE Client ID[9] - LECID of the registering LE Client

Number of TLVs - the number of TLVs found in this CSA

Registered LAN Destination[10] - Unicast LAN Destination registered by the LE Client, used by the LESs for duplicate LAN Destination detection

LE Client ATM Address[11] - ATM address of the LE Client, used by the LE Servers for duplicate ATM address detection and in forwarding certain control frames (e.g. LE_ARP_RESPONSE)

TLV Type - Type of TLV as defined in the LUNI specification [5]. All TLVs that were included in the LE_REGISTER_REQUEST MUST be included in this record.

TLV Length - Length of the TLV Value.

TLV Value - Value of TLV.

[9] Cannot change without changing Cache Key.
[10] Cannot change without changing Cache Key.
[11] Cannot change without changing Cache Key.

4.2.2.7 LE Client Multicast Registration CSA Record Format

Table 4-7. LE Client Multicast Registration CSA Record Format

0	CSA Identifier = 5	Number of TLVs	Must be Zero
4	Remaining Lifetime		
8	LE Client ID		SMS Server ID
12	Registered LAN Destination		
20	LE Client ATM Address		
40	TLV Type		
44	TLV Length	TLV Value (variable length)	
etc…			

CSA Identifier - identifies an LNNI CSA type. Equals 5 for LE Client Multicast Registration CSA.

Remaining Lifetime - The number of seconds before the CSA is considered invalid.

LE Client ID[12] - LECID of the registering LE Client.

Number of TLVs - The number of TLVs found in this CSA.

Registered LAN Destination[13] - multicast MAC Address registered by the LE Client.

SMS ServerId - When SMSs are in distributed mode, identifies the SMS to which the advertising LE Server has assigned the LE Client. Must be zero otherwise.

LE Client ATM Address[14] - The ATM address of the LE Client that registered this multicast MAC Address

TLV Type - Type of TLV as defined in the LUNI specification [5].

TLV Length - Length of the TLV Value.

TLV Value - Value of TLV.

[12] Cannot change without changing Cache Key.
[13] Cannot change without changing Cache Key.
[14] Cannot change without changing Cache Key.

4.2.2.8 SMS Multicast Registration CSA Record Format

Table 4-8. SMS Multicast Registration CSA Record Format

0	CSA Identifier = 6	Number of TLVs	Must be Zero
4	Remaining Lifetime		
8	Registered LAN Destination		
16	SMS Non-multiplexed ATM Address		
36	TLV Type		
40	TLV Length	TLV Value (variable length)	
etc..			

CSA Identifier - Identifies an LNNI CSA type. Equals 6 for SMS Multicast Registration CSA.

Remaining Lifetime - The number of seconds before the CSA is considered invalid.

Number of TLVs - The number of TLVs found in this CSA

Registered LAN Destination[15] - multicast MAC Address served by the advertising SMS

SMS Non-multiplexed ATM Address[16] - ATM address that the advertising SMS has assigned for the multicast MAC address. Used by the LE Servers to answer LE ARP Requests for the multicast MAC address

TLV Type - Type of TLV as defined in the LUNI specification [5].

TLV Length - Length of the TLV Value.

TLV Value - Value of TLV.

4.3 Data Frame Formats

LNNI Data Frame Formats are defined in Section 4.1 of the LUNI specification [5].

[15] Cannot change without changing Cache Key.
[16] Cannot change without changing Cache Key.

5 LNNI Protocols & Procedures [Normative]

5.1 Server Configuration

This section describes the configuration of LE Service components. LE Service configuration is composed of a collection of configuration variables for each component, and the method by which these variables may be configured. All configuration variables are assumed to start in a valid initial state. The setting of the initial state of the variables is determined by means outside the scope of this specification.

Certain of the configuration parameters contain a "valid range" or "default" values. A variable MUST NOT be set to an invalid value. Most ATM Emulated LANs, if composed entirely of LAN Emulation components compliant with this specification, and whose components' variable values are set to the default values, should operate correctly. Altering the values away from the defaults may optimize the behaviour of specific configurations, but such optimization is beyond the scope of this specification. Values outside the specified minima and maxima are likely to result in an emulated LAN that functions poorly or not at all for many applications.

5.1.1 LECS Configuration

5.1.1.1 LECS Configuration Variables

The following LNNI configuration parameters apply to each LE Configuration Server.

5.1.1.1.1 ObtainPeerListViaIlmi

Valid values: TRUE, FALSE.

Default value: TRUE.

Description: When TRUE, the LECS uses ILMI to obtain the list of LECSs in the network. The LECS then uses this list (minus its own ATM address) as its list of peer LECS for the purpose of LECS Synchronization Communications. When FALSE, the LECS uses the **LocallyConfiguredPeerLecsList** as its list of peer LECSs. It is the administrator's responsibility to ensure that each LECS has the ATM addresses of all peer LECSs in its neighbour list.

5.1.1.1.2 LocallyConfiguredPeerLecsList

Valid values: Any collection of ATM addresses

Default value: Empty.

Description: When **ObtainPeerListViaIlmi** is FALSE, this variable contains the list of peer LECSs for this LECS. When **ObtainPeerListViaIlmi** is TRUE, this variable is ignored. It is the administrator's responsibility to ensure that each LECS has the ATM addresses of all peer LECSs in its peer LECS list.

5.1.1.1.3 PeerConnectRetryTimeout

InitialRetryTimeout

Valid Values: 10…30 seconds.

Default Value: 10 seconds

RetryMultiplier

Valid Values: 2…5

Default Value: 2

MaxRetryTimeout

Valid Values: 30…300 seconds

Default Value: 120 seconds.

Description: Servers must establish and maintain connection to peer servers. **InitialRetryTimeout** specifies the time in seconds that a server may wait before it retries after a VCC setup fails. **RetryMultiplier** specifies the factor by which the **PeerConnectRetryTimeout** value must be multiplied before retrying subsequent failed VCC setup. **MaxRetryTimeout** specifies the maximum time in seconds that a server may wait before it retries a setup to the peer server.

5.1.1.1.4 IdleLaneControlVccTimeout

Valid values: 10…3600 seconds (1 hour)

Default value: 120 seconds (2 minutes)

Description: If the server detects that a LNNI control VCC has been idle for more than **IdleLaneControlVccTimeout** seconds, then the server SHOULD release that flow to recover from situations where the remote entity has disappeared or where one of a set of duplicate VCCs is ageing out. If the VCC is not used by other entities, then the VCC must be released..

5.1.1.1.5 ConfigurationTriggerTimeout

Valid values: 10…300 seconds

Default value: 30 seconds

Description: The maximum time that an LECS should wait for a configuration request before retrying a LNNI_CONFIGURE_TRIGGER.

5.1.1.1.6 KeepAliveTime

Valid values: 1…300 seconds

Default value: 30 seconds

Description: The LECS selects this time and sends it to servers in Keep-Alive responses and to other LECSs in synchronization frames within an Operational Server TLV.

5.1.1.2 Identifying Peer LECSs

An LECS MUST determine its set of peer LECSs either via ILMI or by using a locally configured set. If **ObtainPeerListViaIlmi** is TRUE, then the LECS MUST obtain its peer list via ILMI as described in the LUNI specification[5]. If **ObtainPeerListViaIlmi** is FALSE, the neighbour list MUST be obtained from the configuration variable **LocallyConfiguredPeerList**.

5.1.2 LES, BUS and SMS Configuration

5.1.2.1 LES Configuration Variables

The following LNNI configuration variables apply to each LES.

5.1.2.1.1 ElanName

Valid Values: Any SNMPv2 Display String of length 0-32 characters.

Default value: Unspecified.

Description: The name of the ELAN served by this server.

5.1.2.1.2 ElanId

Valid Values: Any non-zero unsigned 32-bit integer.

Default value: None.

Description: An unique identifier for the ELAN. ELAN IDs are used to demultiplex LLC-multiplexed VCCs, so the ELAN ID must be unique across all ELANs which may have entities sharing an LLC-multiplexed VCC.

5.1.2.1.3 SynchronizationPeerServerList

Valid Values: Any collection of ATM addresses.

Default value: Empty.

Description: The servers identified in this list are the servers with which the local server must synchronize databases using SCSP.

5.1.2.1.4 ServerGroupId

Valid Values: Any non-zero unsigned 16-bit integer.

Default value: None.

Description: The **ServerGroupId** uniquely identifies a group of LANE server components operating within an ELAN. Each ELAN within the ATM Network must be configured with a unique **ServerGroupId.** The **ServerGroupId** is used in control and synchronization frames to identify the Server Group (i.e. ELAN) of the originated frame.

5.1.2.1.5 ServerId

Valid Values: Any non-zero unsigned 16-bit integer.

Default value: None.

Description: The **ServerId** uniquely identifies this server within the set of servers for the ELAN. The **ServerId** is NOT globally unique across ELANs. The **ServerId** is used in control and synchronization frames to identify the originator of a request or data within an ELAN.

5.1.2.1.6 LecidRanges

Valid Values: A collection of (LowerBound, UpperBound) pairs where 1 <= LowerBound <= UpperBound <= 65279, and where each pair in the collection represents a disjoint range.

Default value: (X ”1”, X ”FEFF” (65,279)). (see variable C14 in [3])

Description: Each LE Client in an ELAN must receive a unique LECID. In an ELAN with distributed services, each LES in the ELAN must be configured with a disjoint set of LECIDs to return to its local LE Clients. An LES can only assign an LECID to a local LE Client if it falls within one of the LECID ranges.

5.1.2.1.7 PeerConnectRetryTimeout

InitialRetryTimeout

Valid Values: 10…30 seconds.

Default Value: 10 seconds

RetryMultiplier

Valid Values: 2…5

Default Value: 2

MaxRetryTimeout

Valid Values: 30…300 seconds

Default Value: 120 seconds.

Description: Servers must establish and maintain connection to peer servers. **InitialRetryTimeout** specifies the time in seconds that a server may wait before it retries after a VCC setup fails. **RetryMultiplier** specifies the factor by which the **PeerConnectRetryTimeout** value must be multiplied before retrying subsequent failed VCC setup. **MaxRetryTimeout** specifies the maximum time in seconds that a server may wait before it retries a setup to the peer server.

5.1.2.1.8 SMSModeOfOperation

Valid Values: DISTRIBUTED, STAND_ALONE

Default value: STAND_ALONE.

Description: Determines the mode of operation of the SMSs in the ELAN. This variable must be consistent across all servers in the ELAN.

5.1.2.1.9 ServerRefreshLifetime

Valid values: 5…300 seconds.

Default value: 30 seconds.

Description: A server must refresh its server (LES/SMS) refresh CSA periodically to ensure that other servers do not age out the CSA. The **ServerRefreshLifetime** is the lifetime value used by this server in its originated server (LES/SMS) refresh CSAs.

5.1.2.1.10 ClientJoinLifetime

Valid values: 10…3600 seconds.

Default value: 60 seconds.

Description: A server must refresh the LE Client Join CSAs of local LE Clients periodically to ensure that other servers do not age out the CSAs. The **ClientJoinLifetime** is the lifetime value used by this server in its originated LE Client Join CSAs.

5.1.2.1.11 RegistrationLifetime

Valid values: 30…43200 seconds.

Default value: 300 seconds.

Description: A server must refresh the registration CSAs of local LE Clients periodically to ensure that other servers do not age out the CSAs. The **RegistrationLifetime** is the lifetime value used by this server in its originated registration CSAs.

5.1.2.1.12 ServerMuxedAtmAddress

Valid values: any ATM address.

Default value: none.

Description: The ATM Address of the server used for LNNI LLC Multiplexed connections.

5.1.2.1.13 IdleLaneControlVccTimeout

Valid values: 10…3600 seconds (1 hour)

Default value: 120 seconds (2 minutes)

Description: If the server detects that a LNNI control VCC has been idle for more than **IdleLaneControlVccTimeout** seconds, then the server SHOULD release that flow to recover from situations where the remote entity has disappeared or where one of a set of duplicate VCCs is ageing out. If the VCC is not used by other entities, then the VCC must be released..

5.1.2.1.14 ServerReinitDelayMin

Valid values: 1 ms…**ServerReinitDelayMax**

Default value: 1 ms

Description: If the server must return to the initial state, it delays a random amount of time before trying to reinitialize. The minimum delay is given by **ServerReinitDelayMin**.

5.1.2.1.15 ServerReinitDelayMax

Valid values: **ServerReinitDelayMin**…10 seconds

Default value: 5 seconds

Description: If the server must return to the initial state, it delays a random amount of time before trying to reinitialize. The maximum delay is given by **ServerReinitDelayMax**.

5.1.2.1.16 ConfigurationRequestTimeout

Valid values: 10…300 seconds

Default value: 30 seconds

Description: The maximum time that an LES should wait for a configuration response before retrying a configuration request to the LECS.

5.1.2.1.17 SyncHopCount

Valid Values: 1..20

Default: 10

Description: For each CSA that is advertised by a LNNI server (originator), SCSP requires a hop count field to be set in the CSAS record to indicate how many hops the CSU message needs to be forwarded before it is dropped. This is required in order to avoid loops. **SyncHopCount** determines the number of hops the CSU message can traverse before it is dropped.

5.1.2.1.18 ControlRequestRetries

Valid Values: 1-10

Default: 3

Description: The maximum number of times that the server will send a control frame.

5.1.2.2 BUS Configuration Variables

There are no LNNI configuration parameters for the BUS.

5.1.2.3 SMS Configuration Variables

The following LNNI configuration variables apply to each SMS.

5.1.2.3.1 ElanName

Valid Values: Any SNMPv2 Display String of length 0-32 characters.

Default value: None.

Description: The name of the ELAN served by this server.

5.1.2.3.2 ElanId

Valid Values: Any non-zero unsigned 32-bit integer.

Default value: None.

Description: An unique identifier for the ELAN. ELAN IDs are used to demultiplex LLC-multiplexed VCCs, so the ELAN ID must be unique across all ELANs which may have entities sharing an LLC-multiplexed VCC.

5.1.2.3.3 SynchronizationPeerServerList

Valid Values: Any collection of ATM addresses.

Default value: Empty.

Description: The servers identified in this list are the servers with which the local server must synchronize databases using SCSP.

5.1.2.3.4 ServerGroupId

Valid Values: Any non-zero unsigned 16-bit integer.

Default value: None.

Description: The **ServerGroupId** uniquely identifies a group of LANE server components operating within an ELAN. Each ELAN within the ATM Network must be configured with a unique **ServerGroupId.** The **ServerGroupId** is used in control and synchronization frames to identify the Server Group (i.e. ELAN) of the originated frame.

5.1.2.3.5 ServerId

Valid Values: Any non-zero unsigned 16-bit integer.

Default value: None.

Description: The **ServerId** uniquely identifies this server within the set of servers for the ELAN. The **ServerId** is NOT globally unique across ELANs. The **ServerId** is used in control and synchronization frames to identify the originator of a request or data within an ELAN.

5.1.2.3.6 PeerConnectRetryTimeout

InitialRetryTimeout

Valid Values: 10…30 seconds.

Default Value: 10 seconds

RetryMultiplier

Valid Values: 2…5

Default Value: 2

MaxRetryTimeout

Valid Values: 30…300 seconds

Default Value: 120 seconds.

Description: Servers must establish and maintain connection to peer servers. **InitialRetryTimeout** specifies the time in seconds that a server may wait before it retries after a VCC setup fails. **RetryMultiplier** specifies the factor by which the **PeerConnectRetryTimeout** value must be multiplied before retrying subsequent failed VCC setup. **MaxRetryTimeout** specifies the maximum time in seconds that a server may wait before it retries a setup to the peer server.

5.1.2.3.7 OptimizedSmsTopology

Valid Values: TRUE, FALSE

Default value: FALSE.

Description: If TRUE, an SMS may discard unicast registration data received via SCSP, and is not required to flood synchronization data received on one port out its other ports. If TRUE, then a LES is not required to synchronize unicast registration data to neighbour SMSs. If FALSE, an SMS must maintain a complete registration database including unicast information, and must flood new data out all ports as specified by the synchronization protocol. If FALSE, then an LES must synchronize unicast registration data with neighbour SMSs. When TRUE, it is the responsibility of the system administrator to ensure that removal of the SMSs would not result in partitioning of the SCSP topology into "islands".

5.1.2.3.8 SmsModeOfOperation

Valid Values: DISTRIBUTED, STAND_ALONE

Default value: STAND_ALONE.

Description: Determines the mode of operation of the SMSs in the ELAN. This variable must be consistent across all servers in the ELAN.

5.1.2.3.9 SmsMuxedAtmAddress

Valid values: any ATM address.

Default value: none.

Description: The ATM Address of the SMS used for LNNI LLC Multiplexed connections.

5.1.2.3.10 SmsNonmuxedAtmAddress

Valid values: any ATM address.

Default value: none.

Description: The ATM Address of the SMS used for non-LLC multiplexed connections.

5.1.2.3.11 IdleLaneControlVccTimeout

Valid values: 10…3600 seconds (1 hour)

Default value: 120 seconds (2 minutes)

Description: If the server detects that a LNNI control VCC has been idle for more than **IdleLaneControlVccTimeout** seconds, then the server SHOULD release that flow to recover from situations where the remote entity has disappeared or where one of a set of duplicate VCCs is ageing out. If the VCC is not used by other entities, then the VCC must be released.

5.1.2.3.12 MAC Multicast Addresses

Valid values: list of multicast MAC addresses

Default value: none

Description: Multicast MAC addresses assigned to the SMS. An SMS may support more than one multicast MAC address.

5.1.2.3.13 ServerRefreshLifetime

Valid values: 5…300 seconds.*Default value*: 30 seconds.

Description: A server must refresh its server (LES/SMS) refresh CSA periodically to ensure that other servers do not age out the CSA. The **ServerRefreshLifetime** is the lifetime value used by this server in its originated server (LES/SMS) refresh CSAs.

5.1.2.3.14 RegistrationLifetime

Valid values: 30…43200 seconds.*Default value*: 300 seconds.

Description: A server must refresh the registration CSAs of multicast addresses it serves periodically to ensure that other servers do not age out the CSAs. The **RegistrationLifetime** is the lifetime value used by this server in its originated registration CSAs.

5.1.2.3.15 ServerReinitDelayMin

Valid values: 1 ms…**ServerReinitDelayMax**

Default value: 1 ms

Description: If the server must return to the initial state, it delays a random amount of time before trying to reinitialize. The minimum delay is given by **ServerReinitDelayMin**.

5.1.2.3.16 ServerReinitDelayMax

Valid values: **ServerReinitDelayMin**…10 seconds

Default value: 5 seconds

Description: If the server must return to the initial state, it delays a random amount of time before trying to reinitialize. The maximum delay is given by **ServerReinitDelayMax**.

5.1.2.3.17 ConfigurationRequestTimeout

Valid values: 10…300 seconds

Default value: 30 seconds

Description: The maximum time that an SMS should wait for a configuration response before retrying a configuration request to the LECS.

5.1.2.3.18 SyncHopCount

Valid Values: 1..20

Default: 10

Description: For each CSA that is advertised by a LNNI server (originator), SCSP requires a hop count field to be set in the CSAS record to indicate how many hops the CSU message needs to be forwarded before it is dropped. This is required in order to avoid loops. **SyncHopCount** determines the number of hops the CSU message can traverse before it is dropped.

5.1.2.3.19 ControlRequestRetries

Valid Values: 1-10

Default: 3

Description: The maximum number of times that the server will send a control frame.

5.1.2.4 Returning to Initial State

Whenever this specification indicates that a server must return to its initial state, the server MUST reinitialize all LNNI configuration variables to their original initial state, MUST implement a random delay that is selected from the uniform distribution over the interval [**ServerReinitDelayMin, ServerReinitDelayMax]**, and MUST restart the configuration phase by reacquiring the location of the LECS(s).

5.1.2.5 Configuration Exchange – Requestor's View

5.1.2.5.1 Locating and Connecting to an LECS

LESs and SMSs locate LECSs using the same methods specified for LE Clients in the LUNI specification [5]. LESs and SMSs MUST connect to an LECS to obtain initial configuration. Configuration Direct VCCs are signalled as specified in the LUNI specification [5].

Unlike LE Clients, LNNI compliant LESs and SMSs MUST NOT bypass LECS configuration. Co-located LE Clients, LESs, and SMSs MAY share a common Configuration Direct VCC.

5.1.2.5.2 Configuration Request

Each LES and SMS MUST transmit an LE_CONFIGURE_REQUEST to the LECS over a Configuration Direct VCC. The configuration request MUST contain the ATM address of the server in the SOURCE-ATM-ADDRESS field. The configuration request MUST contain the ELAN name, ELAN type, and maximum frame size if these are configured at the server (the ATM address, ELAN type, and maximum frame size configuration variables are defined as **S1**, **S2**, and **S3** in the LUNI specifications[5], respectively). The format of the configuration request is the same as the format of a configuration request from an LE Client with the exception that the IS_LE_SERVER or IS_MCAST_SERVER flag is set in the request. LESs set the IS_LE_SERVER flag; SMSs set the IS_MCAST_SERVER flag. If the LES already has operational LE Clients, then it MAY specify a range(s) LECIDs in the LECID range TLV(s). The format of the Configuration Request is given in Table 5-1. The server MUST include the Operational Server TLV in the configuration request.

5.1.2.5.3 Unsuccessful Configuration Response

If the server receives an unsuccessful configuration response (status not equal to "Success") with the appropriate IS_LE_SERVER or IS_MCAST_SERVER flag set, then the server MUST return to its initial state. If the server receives an unsuccessful configuration response with the appropriate flag cleared, then it MUST attempt to connect to the next LECS in order. If there are no more LECSs with which to attempt a connection, the server MUST return to its initial state. The latter condition (appropriate flag cleared in response) may occur if the server connected to a non-LNNI capable LECS.

If the server receives a successful configuration response (status equal to "Success") with the appropriate IS_LE_SERVER or IS_MCAST_SERVER flag cleared, then it MUST ignore this response and attempt to connect to the next LECS in order. If there are no more LECS with which to attempt a connection, the server MUST return to its initial state.

5.1.2.5.4 Resending the Configuration Request

If a configuration response is not received by the server, the server MUST keep resending a configuration request every **ConfigurationRequestTimeout** seconds until a configuration response is received, the **ControlRequestRetries** count is exceeded, or the Configuration Direct VCC is released. If **ControlRequestRetries** is exceeded, the server should return to initialization state.

5.1.2.5.5 Successful Configuration Response

If the server receives a successful configuration response (status equal to "Success") with the appropriate IS_LE_SERVER or IS_MCAST_SERVER flag set, then the server MUST either copy the parameters provided in the response into its local variables, or return to its initial state. The parameters to be copied are the ELAN name, ELAN type, maximum frame size, and all TLV values. The provision permitting the server to return to its initial state provides a possible course of action in case of misconfiguration, where the server is not capable of acting as directed by the LECS. If the LES receives the LECID range TLV(s) in the response, then the LES MUST terminate all operational local LE Clients whose LECIDs are not included in the TLV(s). The server MUST use the value returned in the Operational Server TLV as its initial Keep-Alive time as described in Section 5.2.1.1.

5.1.2.5.6 Configuration Response TLVs

The TLVs that can be returned in successful configuration response are given in Table 5-2.

Servers must ignore User-defined TLVs that can not be decoded. Servers should ignore returned TLVs which are not supported by the server type. For example, a Client Join Lifetime TLV returned to an SMS.

5.1.2.5.7 Releasing Configuration Direct VCCs

An LE Service component MUST NOT release the Configuration Direct VCC after completing its configuration. This VCC is used for Keep-Alive exchanges with the LECS as described in Section 5.2.

5.1.2.6 Configuration Exchange – LECS' View

5.1.2.6.1 Accepting Configuration Direct VCCs

An LECS MUST accept all incoming connections satisfying the conditions of a Configuration Direct VCC as specified in the LUNI specification[5].

5.1.2.6.2 Releasing Configuration Direct VCCs

An LECS MUST NOT release a Configuration Direct VCC after transmitting a configuration response. An LECS SHOULD release a Configuration Direct VCC if it has not been used to receive or transmit data for **IdleLaneControlVccTimeout** seconds.

5.1.2.6.3 Server Configuration Requests

An LECS MUST recognize the IS_LE_SERVER and IS_MCAST_SERVER flags to distinguish server requests from client requests. The LECS identifies the server by the combination of the SOURCE-ATM-ADDRESS and ELAN name in the configuration request.

5.1.2.6.4 Configuration Response

If the LECS cannot determine the ELAN for the server, then the LECS MUST return an unsuccessful configuration response with status "No configuration". Otherwise, the LECS returns a successful configuration response. The IS_LE_SERVER and IS_MCAST_SERVER flags MUST be set in the configuration response if they were set in the corresponding configuration request.

5.1.2.6.5 Unsuccessful Configuration Response

The LECS MUST NOT include any TLV information in an unsuccessful configuration response.

5.1.2.6.6 Successful Configuration Response

The LECS MUST set the ELAN name, ELAN type, and maximum frame size in a successful configuration response. The returned ELAN type and maximum frame size MUST be compatible with the ELAN type and maximum frame size specified in the request, where compatible is defined as in the LUNI specification. The LECS MUST return a **ServerGroupId** for the ELAN and a **ServerId** TLV uniquely identifying this server within the assigned ELAN. If the server is an LES and there may be multiple LESs serving the ELAN, then the LECS MUST return a disjoint set of LECIDs to each LES using an LECID Range TLV(s). If the LECS receives a configuration request with a LECID range TLV(s), then it SHOULD return a superset of the requested LECID ranges in its response if the requested LECIDs are not already assigned to another LES. If there may be multiple servers for this ELAN, then the LECS MUST return at least one neighbour server using the Synchronization Neighbour TLV. The LECS MAY return additional servers depending on the desired level of synchronization connectivity. The method used by the LECS to determine which neighbouring servers to return is left unspecified. However, we mention several possibilities:

- Return a static set of neighbours to each server so that the graph of synchronized servers is connected.
- Return a static set of neighbours to each server so that the graph of synchronized LESs is connected, and so SMSs are connected to at least one LES. In this case, SMSs SHOULD be notified that they can discard unicast registrations via the Optimized SMS Topology TLV in the configuration response.
- Return a dynamic set of neighbours to each server based upon which servers are currently serving the ELAN. In this case, the LECS MUST make use of the configuration trigger protocol to notify existing servers that a newly configured server can be expected to synchronize with them.

The LECS MAY return the Synchronization Hop Count TLV. If it is returned, it MUST be set to at least the depth of the graph of synchronized servers with the server (LES or SMS) as the root. This value may vary for different servers based on the graph.

The LECS MUST include an Operational Server TLV in any successful configuration response to a LES or SMS. The Operational Server TLV includes a Keep-Alive time which the server can use as its initial Keep-Alive interval for the Keep-Alive protocol. The ATM address in the TLV must equal the ATM address in the SOURCE-ATM-ADDRESS field of the configuration response.

5.1.2.7 Configuration Frame Formats

The format of a configuration frame is given in Table 5-1. The list of TLVs which are valid in a configuration request are given in Table 5-2. The list of TLVs which are valid in a successful configuration response to a server are given in Table 5-3.

Table 5-1 Server Configuration Frame Format

Offset	Size	Name	Function
0	2	MARKER	Control Frame = X”FF00”
2	1	PROTOCOL	ATM LAN Emulation protocol = X”01”
3	1	VERSION	ATM LAN Emulation protocol version = X”01”
4	2	OP-CODE	Type of request. See Table 4-2
6	2	STATUS	Always X”0000” in requests. See the LUNI specifications for a list of supported values in a response.
8	4	TRANSACTION-ID	Arbitrary value supplied by the requester and returned by the responder.
12	2	REQUESTER-LEC-ID	Always X"0000" in requests, ignored on response.
14	2	FLAGS	X”0400” IS_LE_SERVER flag set by an LES. X”0800” IS_MCAST_SERVER flag set by an SMS.
16	8	SOURCE-LAN-DESTINATION	Always “not present” when sent, ignored on receipt.
24	8	TARGET-LAN-DESTINATION	Always “not present” when sent, ignored on receipt.
32	20	SOURCE-ATM-ADDRESS	Non LLC-muxed ATM address of prospective server for which information is requested.
52	1	LAN-TYPE	X”00” Unspecified[17] X”01” Ethernet/IEEE 802.3 X”02” IEEE 802.5
53	1	MAXIMUM-FRAME-SIZE	X”00” Unspecified X”01” 1516 X”02” 4544 X”03” 9234 X”04” 18190 X”05” 1580
54	1	NUMBER-TLVS	Number of Type/Length/Value elements encoded in Request/Response.
55	1	ELAN-NAME-SIZE	Number of octets in ELAN-NAME (may be 0).
56	20	TARGET-ATM-ADDRESS	X'00' when sent, ignored on receipt.
76	32	ELAN-NAME	Name of emulated LAN[18].
108	4	ITEM_1-TYPE	Three octets of OUI, one octet identifier.
112	1	ITEM_1-LENGTH	Length in octets of VALUE field. Minimum=0.

[17] If the LAN-TYPE is “unspecified,” then the Ethernet/IEEE 802.3 MAC address format MUST be used.
[18] SNMPv2 DisplayString

113	Variable	ITEM_1-VALUE	
		Etc.	

Table 5-2 : Supported TLVs in Configuration Request

Operational Server TLV
LECID Range TLV (may be multiple)

Table 5-3 Supported TLVs in Configuration Response

TLV Items	**May be Included in Responses as Indicated by Server Flags**
ServerId	IS_LE_SERVER or IS_MCAST_SERVER
ServerGroupId	IS_LE_SERVER or IS_MCAST_SERVER
Synchronization Peer Server (may be multiple)	IS_LE_SERVER or IS_MCAST_SERVER
LECID Range (may be multiple)	IS_LE_SERVER
Control Timeout (S4)	IS_LE_SERVER
Maximum Frame Age (S5)	IS_LE_SERVER
Local Segment ID (S8)	IS_LE_SERVER
Disconnect Timeout (S9)	IS_LE_SERVER
ELAN ID	IS_LE_SERVER or IS_MCAST_SERVER
Peer Reconnect Retry Timeout	IS_LE_SERVER or IS_MCAST_SERVER
Server Refresh Lifetime	IS_LE_SERVER IS_MCAST_SERVER
Client Join Lifetime	IS_LE_SERVER
Registration Lifetime	IS_LE_SERVER
Idle LANE Control VCC Timeout	IS_LE_SERVER or IS_MCAST_SERVER
May Discard Unicast Registrations	IS_MCAST_SERVER
Mode of Operation	IS_MCAST_SERVER
Operational Server TLV	IS_LE_SERVER or IS_MCAST_SERVER
Synchronization Hop Count	IS_LE_SERVER or

	IS_MCAST_SERVER
Multicast MAC Address (may be multiple)	IS_MCAST_SERVER

5.1.3 Configuration Trigger

The LECS uses the configuration trigger message to force a server to reacquire its configuration. This feature is used to enact dynamic configuration changes. The configuration trigger can be used to notify operational servers that a new server is joining the ELAN, and to disable an operating server. Many other uses are possible.

5.1.3.1 Configuration Trigger – LECS' View

5.1.3.1.1 Instigating Server Reconfiguration

The LECS MAY send a LNNI_CONFIGURE_TRIGGER message to any locally attached active server at any time. The message MUST be sent over the Configuration Direct VCC used for the most recent status or configuration exchange with that server. An LECS MUST NOT send a configuration trigger to a non-locally attached server through another LECS. It is the responsibility of the other LECS to instigate configuration triggers for its local servers.

5.1.3.1.2 Identifying the Target Server

The LECS identifies the target server by putting the appropriate ATM address in the TARGET-ATM-ADDRESS field of the LNNI_CONFIGURE_TRIGGER, and the ELAN name in the ELAN-NAME field. The appropriate ATM address to trigger an LES is the non-multiplexed ATM address (**S1**). The appropriate ATM address to trigger an SMS is the non-multiplexed ATM address (**SmsNonmuxedAtmAddress**). The format of the LNNI_CONFIGURE_TRIGGER is given in Table 5-4.

5.1.3.1.3 No Response from Target Server

If the target server of LNNI_CONFIGURE_TRIGGER does not issue a configuration request within **ConfigurationTriggerTimeout** seconds, the LECS MAY transmit another the LNNI_CONFIGURE_TRIGGER with a new TRANSACTION-ID. Retrying the configuration trigger may be necessary if the original configuration trigger or the configuration request was lost. If the LECS is unable to trigger a server into configuration, it MAY discontinue acknowledging the Keep-Alive requests from the server.

5.1.3.2 Configuration Trigger – LES/SMS View

5.1.3.2.1 Receiving Configuration Trigger

If a LNNI_CONFIGURE_TRIGGER is received with a TARGET-ATM-ADDRESS equal to the non LLC-multiplexed ATM address of the server and an ELAN-NAME equal to the ELAN name of the server, then the server MUST issue a configuration request on the Configuration Direct VCC over which the trigger was received. The format of the configuration trigger is given in Table 5-4. If the TARGET-ATM-ADDRESS or ELAN name is not that of the server, then the server MUST ignore the trigger request.

5.1.3.2.2 Unsuccessful Configuration Response

If the server receives an unsuccessful configuration response, then it MUST terminate all client and peer connections and return to its initial state.

5.1.3.2.3 Successful Configuration Response

If the server receives a successful configuration response, then it MUST attempt to integrate this new configuration into its current configuration. If only the set of neighbour servers changes, then the configuration is compatible and

the server MUST change its neighbour set without affecting local LE Clients. This change will most likely result in peer connections being dropped and added. If any other configuration variable changes, the server MUST terminate all client and peer connections and return to the initial state.

5.1.3.2.4 No Configuration Response

5.1.3.2.5 Resending the Configuration Request

If a configuration response is not received by the server, the server MUST keep resending a configuration request every **ConfigurationRequestTimeout** seconds until a configuration response is received, the **ControlRequestRetries** count is exceeded, or the Configuration Direct VCC is released. If **ControlRequestRetries** is exceeded, the server should return to initialization state.

5.1.4 Configuration Trigger Frame Formats

The format of the LNNI_CONFIGURE_TRIGGER message is given in Table 5-4. No TLVs are supported in LNNI_CONFIGURE_TRIGGER messages.

Table 5-4 : Configuration Trigger Frame Format

Offset	Size	Name	Function
0	2	MARKER	Control Frame = X”FF00”
2	1	PROTOCOL	ATM LAN Emulation protocol = X”01”
3	1	VERSION	ATM LAN Emulation protocol version = X”01”
4	2	OP-CODE	Type of request. See Table 4-2.
6	2	STATUS	Always X”0000” in requests. See the LUNI specifications for a list of supported values in a response.
8	4	TRANSACTION-ID	Arbitrary value supplied by the requester and returned by the responder.
12	2	REQUESTER-LEC-ID	Always X"0000" in requests, ignored on response.
14	2	FLAGS	X”0000” when sent, ignored on receipt.
16	8	SOURCE-LAN-DESTINATION	Always “not present” when sent, ignored on receipt.
24	8	TARGET-LAN-DESTINATION	Always “not present” when sent, ignored on receipt.
32	20	SOURCE-ATM-ADDRESS	X”00” when sent, ignored on receipt.
52	1	LAN-TYPE	X”00” when sent, ignored on receipt.
53	1	MAXIMUM-FRAME-SIZE	X”00” when sent, ignored on receipt.
54	1	NUMBER-TLVS	Number of Type/Length/Value elements encoded in Request/Response. There are no TLVs currently defined for Configuration Triggers.
55	1	ELAN-NAME-SIZE	Number of octets in ELAN-NAME (may be 0).
56	20	TARGET-ATM-ADDRESS	Non LLC-multiplexed ATM address of prospective server for which trigger is targeted.
76	32	ELAN-NAME	Name of Emulated LAN[19].
108	4	ITEM_1-TYPE	Three octets of OUI, one octet identifier.
112	1	ITEM_1-LENGTH	Length in octets of VALUE field. Minimum=0.
113	Variable	ITEM_1-VALUE	
		Etc.	

5.2 Status Communications

Status communications provide the method for servers to notify LECSs that they are operational, and for LECSs to share information on locally attached operational servers. The status communications and configuration communications between a particular server and an LECS use the same VCC. This VCC is initially established for configuration, and must remain for status communications and possibly for configuration triggers.

[19]SNMPv2 DisplayString

5.2.1 Status Communications – LES/SMS View

5.2.1.1 Initial Keep-Alive Request

As part of its configuration, a server receives an Operational Server TLV which supplies its initial Keep-Alive time.

The Keep-Alive request contains an Operational-Server TLV(s) identifying the LES(s) indicating operational status to the LECS. The status can be normal or resource starved (e.g. memory or VCC).

5.2.1.2 Periodic Keep-Alive Requests

Each server MUST send periodic Keep-Alive requests to an LECS. The server MUST send a request before one-third of the Keep-Alive time has expired. If the request is unacknowledged, another request MUST be sent before two-thirds of the Keep-Alive time has expired. The server MAY choose to send Keep-Alive requests more frequently.

If a Keep-Alive response is not received before the Keep-Alive time expires, the server MUST clear the Configuration Direct VCC if the VCC is not being shared by another LNNI entity and the server MUST attempt to establish a new Configuration Direct VCC to one of the LECSs that it learned about during initialization (see 5.1.2.5.1) and with which it has not yet tried to connect to with a Configuration Direct VCC. If the server has attempted to establish (successfully or unsuccessfully) a Configuration Direct VCC with all of the LECSs that it knows about, then the server MUST return to the initial state. An LES SHOULD continue to service clients until it is forced to return to the initial state.

5.2.1.3 Collective Keep-Alive Requests

Since multiple servers can use the same VCC for configuration and status, Keep-Alive requests from multiple co-located servers SHOULD be combined into a single Keep-Alive request. A Operational-Server TLV for each server is required in the Keep-Alive request. Each Operational Server TLV contains the non-multiplexed ATM address of a server.

5.2.1.4 Keep-Alive Responses

Each Keep-Alive response from the LECS includes the same set of Operational-Server TLVs as the corresponding request. The LECS provides a Keep-Alive time in each Operational-Server TLV. This time indicates the number of seconds that the LECS will assume the server is alive in the absence of further Keep-Alive requests for the server.

5.2.1.5 Release of Configuration Direct VCC

If a Configuration Direct VCC is released, the SMS or LES MUST attempt to establish a new Configuration Direct VCC to one of the LECSs that it learned about during initialization (see 5.1.2.5.1) and with which it has not yet tried to connect to with a Configuration Direct VCC. If the Configuration Direct VCC is released, the server SHOULD not cancel a Keep Alive Timer if one is running. If a new Configuration Direct VCC is established before the expiration of the Keep Alive Timer the SMS or LES SHOULD not send a configuration but SHOULD continue to operate with its existing configuration.

If the SMS or LES has attempted to establish (successfully or unsuccessfully) a Configuration Direct VCC with all of the LECSs that it knows about, then the SMS or LES MUST return to the initial state.

The SMS or LES SHOULD continue to service clients until it is forced to return to the initial state.

5.2.1.6 Expiration of Keep-Alive Time

The current Keep-Alive Time for an server is the Keep-Alive Time indicated in the Operational Server TLV received in most recent Keep-Alive response from the LECS.

If an server does not receive a Keep-Alive response from the LECS for Keep-Alive Time seconds, , then the server MUST terminate all peer server connections and return to its initial state. An LES (and its associated BUS) SHOULD continue to service local clients as if it were the only LES on the ELAN. The server MUST reconfigure upon re-establishing communications with the LECS. A SMS SHOULD terminate all LE Client connections if its Keep-Alive time expires.

5.2.2 Status Communications – LECS View

5.2.2.1 Maintaining Server Status

Each LECS MUST maintain status on each LES to which an LE Client may be assigned.

5.2.2.2 Using Server Status

LE Clients MUST NOT be assigned to servers that are not known to be operational via status communications. LECSs MAY use the operational status of server to update the **SynchronizationPeerServerList** of other servers when the status of an server changes.

5.2.2.3 Responding to Keep-Alive Requests

When an LECS is not waiting for a configuration request from an LES in response to a configuration trigger sent to the server, the LECS MUST respond to each Keep-Alive request with a Keep-Alive response. The LECS MUST assign a Keep-Alive time to each server identified in the request. The LECS MUST consider the server to be operational for the interval specified by the Keep-Alive time.

When an LECS is waiting for a configuration request from an server in response to a configuration trigger sent to the server, the LECS MUST ignore all Keep-Alive requests from the server. The LECS MUST consider the server to be operational while waiting for the configuration request.

5.2.3 Keep-Alive Frame Formats

The format of the Keep-Alive request and response is given in Table 5-5. The TLVs supported in Keep-Alive frames are given in Table 5-6.

Table 5-5 Keep-Alive Frame Format

Offset	Size	Name	Function
0	2	MARKER	Control Frame = X”FF00”
2	1	PROTOCOL	ATM LAN Emulation protocol = X”01”
3	1	VERSION	ATM LAN Emulation protocol version = X”01”
4	2	OP-CODE	Type of request. See Table 4-2.
6	2	STATUS	Always X”0000” in requests. See the LUNI specifications for a list of supported values in a response.
8	4	TRANSACTION-ID	Arbitrary value supplied by the requester and returned by the responder.
12	2	REQUESTER-LEC-ID	Always X"0000" in requests, ignored on response.
14	2	FLAGS	X”0000” when sent, ignored on receipt.
16	8	SOURCE-LAN-DESTINATION	Always “not present” when sent, ignored on receipt.
24	8	TARGET-LAN-DESTINATION	Always “not present” when sent, ignored on receipt.
32	20	SOURCE-ATM-ADDRESS	X'00' when sent, ignored on receipt.
52	1	LAN-TYPE	X”00” when sent, ignored on receipt.
53	1	MAXIMUM-FRAME-SIZE	X”00” when sent, ignored on receipt.
54	1	NUMBER-TLVS	Number of Type/Length/Value elements encoded in Request/Response.
55	1	ELAN-NAME-SIZE	X”00” when sent, ignored on receipt.
56	20	TARGET-ATM-ADDRESS	The ATM address of the target LECS.
76	32	ELAN-NAME	X”00” when sent, ignored on receipt.
108	4	ITEM_1-TYPE	Three octets of OUI, one octet identifier.
112	1	ITEM_1-LENGTH	Length in octets of VALUE field. Minimum=0.
113	Variable	ITEM_1-VALUE	
		Etc.	

Table 5-6 TLVs Supported in Keep-Alive Frames

Operational-Server TLV

5.3 LECS Synchronization Communications

LECS synchronization is the process by which an LECS notifies other LECSs of locally attached operational servers. Each LECS must connect to its peer LECSs to form a mesh of LECSs. Each LECS is responsible for notifying other LECSs of the locally attached servers.

5.3.1 Connecting to Peer LECSs

The LECS MUST attempt to connect to each LECS in its peer list. If the connection to a neighbour fails, then the LECS must continuously retry the connection based on the logic in section 5.1.1.1.3. The connection is signalled using the signalling parameters for a Configuration Direct VCC as described in the LUNI specification[5] with the exception that the LECS MAY signal the connection as a point-to-multipoint connection. Use of a point-to-multipoint connection to neighbouring LECSs reduces the number of data transmissions by an LECS.

5.3.2 Releasing Duplicate Connections

If duplicate point-to-point connections are established between two LECSs, then the LECSs MUST use the rules for ageing duplicate VCCs as specified in Section 8.1.13 of the LUNI specification [5] to ensure that one of the VCCs ages out.

5.3.3 Authenticating Peer LECSs

An LECS MAY choose to perform authentication before accepting a new LECS as a peer. Peer LECSs are first identified when the local LECS receives an LECS synchronization message from them. Once identified, authentication MAY be performed. The method of authentication is outside the scope of this document, but we mention one possibility. The local LECS could compare the ATM address of the far end of the VCC over which the synchronization message was received against the peer list. If the ATM address is not on the peer list, then the LECS could discard the synchronization message and all future synchronization messages over that VCC. The LECS could also release the VCC.

5.3.4 Transmitting LECS Synchronization Messages

Each LECS MUST notify all other LECSs of each locally attached LES via an LECS Synchronization Request. Synchronization requests MUST be transmitted before one-third of the **KeepAliveTime** elapses since the previous LECS Synchronization Request was sent.

5.3.5 Receiving LECS Synchronization Messages

Each received LECS synchronization message identifies a set of servers and indicates the length of time that each server should be considered operational. An LECS MUST consider each identified server to be operational with the defined operational status for the specified interval.

5.3.6 LECS Synchronization Keep-Alive Timeout

There are no immediate consequences for an LECS which fails to receive an LECS synchronization message from one of its peers within **KeepAliveTime**. Sychronized LES status information is removed from each LECS based on the expiration of the LES's associated Keep-Alive time. LE clients MUST NOT be assigned to servers that are not known to be operational. LE clients MUST NOT be assigned to servers that are known to be resource starved.

5.3.7 Release of Peer Connection

If a LECS Synchronization VCC to a peer LECS is released, the LECS MUST attempt to establish a new LECS Synchronization VCC to the peer. Until connectivity is established, an LECS MUST retry continuously based on the logic in section 5.1.1.1.3.

5.3.8 LECS Synchronization Frame Format

The format of LECS Synchronization frames is given in Table 5-7. The TLVs supported in LECS synchronization frames are listed in Table 5-8.

Table 5-7 Format of LECS Synchronization Frame

Offset	Size	Name	Function
0	2	MARKER	Control Frame = X”FF00”
2	1	PROTOCOL	ATM LAN Emulation protocol = X”01”
3	1	VERSION	ATM LAN Emulation protocol version = X”01”
4	2	OP-CODE	Type of request. See Table 4-2.
6	2	STATUS	Always X”0000” in requests. See the LUNI specifications for a list of supported values in a response.
8	4	TRANSACTION-ID	Arbitrary value supplied by the requester and returned by the responder.
12	2	REQUESTER-LEC-ID	Always X"0000" in requests, ignored on response.
14	2	FLAGS	X”0000” when sent, ignored on receipt.
16	8	SOURCE-LAN-DESTINATION	Always “not present” when sent, ignored on receipt.
24	8	TARGET-LAN-DESTINATION	Always “not present” when sent, ignored on receipt.
32	20	SOURCE-ATM-ADDRESS	The ATM address of the source LECS.
52	1	LAN-TYPE	X”00” when sent, ignored on receipt.
53	1	MAXIMUM-FRAME-SIZE	X”00” when sent, ignored on receipt.
54	1	NUMBER-TLVS	Number of Type/Length/Value elements encoded in Request/Response.
55	1	ELAN-NAME-SIZE	X”00” when sent, ignored on receipt.
56	20	TARGET-ATM-ADDRESS	X'00' when sent, ignored on receipt.
76	32	ELAN-NAME	X”00” when sent, ignored on receipt.
108	4	ITEM_1-TYPE	Three octets of OUI, one octet identifier.
112	1	ITEM_1-LENGTH	Length in octets of VALUE field. Minimum=0.
113	Variable	ITEM_1-VALUE	
		Etc.	

Table 5-8 TLVs Supported in LECS Synchronization Frames

Operational-Server TLV

5.4 Database Synchronization

LESs and SMSs must have synchronized databases to ensure proper responses to LE Client requests. LES/SMS Synchronization Communications take place over Cache Synchronization VCCs. Cache Synchronization VCCs are LLC-muxed VCCs. A server may elect to use the same VCC for Synchronization Communications as it uses for Control Communications. Cache synchronization is accomplished via SCSP[4].

5.4.1 LES/SMS Cache Synchronization VCCs

This section discusses the establishment and maintenance of Cache Synchronization VCCs.

5.4.1.1 Establishing Cache Synchronization VCCs

Upon completion of configuration, each server MUST attempt to establish Cache Synchronization VCC to each peer in its **SynchronizationPeerServerList** for which a Cache Synchronization VCC does not already exist.

The Cache Synchronization VCC setup MAY be performed by opening a new point-to-point VCC to the neighbouring server or as a leaf to a point-to-multipoint call at the choice of the server implementation.

5.4.1.2 B-LLI Codepoint for Cache Synchronization VCC

Cache Synchronization VCCs MUST be LLC-multiplexed VCCs. All other call parameters signalled by the LE Server are those defined in Section 3"Connection Management Services".

5.4.1.3 Handling Cache Synchronization VCC Setup Failure

If the Cache Synchronization VCC setup fails, the calling server MUST retry continuously based on the logic in section 5.1.2.1.7 until a connection is successfully established or until such time as the LECS notifies the calling server of a configuration change removing the called neighbouring server from its neighbour **SynchronizationPeerServerList**.

5.4.1.4 Accepting the Cache Synchronization VCC

A server MUST accept a UNI signalling request to establish an LLC-multiplexed VCC to its **ServerMuxedAtmAddress**, with the assumption that this VCC will be used as a Cache Synchronization or Control Coordinate VCC. A server MAY refuse a UNI signalling request if it contains call parameters other than as defined in Connection Management Services (Section 3).

The server MAY release this connection if, upon completion, no valid synchronization or control communications use the VCC within the **IdleLaneControlVccTimeout** AND the calling party address does not match a configured server address in its **SynchronizationPeerServerList** .

5.4.1.5 Handling Duplicate Cache Synchronization VCCs

When an server receives an incoming Cache Synchronization connection request from an ATM address to which it already has a connection, it SHOULD accept the request and apply the rules for duplicate Data Direct VCCs found in Section 8.1.13 of [5] for ageing out the appropriate duplicate VCC.

5.4.1.6 Handling Release of Cache Synchronization SVC

If a Cache Synchronization SVC is released by a neighbouring server, the local server MUST attempt to re-establish connection to the server which initiated the release. If the setup request fails, the calling server MUST retry the connection as described in Sections 5.1.2.1.7 and 5.1.2.3.6 until a connection is successfully established, or until such time as the LECS notifies the calling server of a configuration change removing the called neighbouring server from its **SynchronizationPeerServerList**.

5.4.1.7 Server Reconfiguration

During reconfiguration of the server, the server MUST suspend inter-server communication until a determination of required topology changes is made. If a reconfiguration results in no changes relative to neighbouring servers, inter-server communication and synchronization may be resumed.

If a server is removed as a peer of the local server during reconfiguration, then the local server MAY release the Cache Synchronization VCC, or MAY simply stop using the VCC for synchronization communications. If the VCC is being used for control communications or for some other LLC-muxed protocol, then the server MUST simply stop using the VCC for synchronization communications. Otherwise, the Cache Synchronization VCC MAY be immediately released or MAY simply stop being used, in which case it MUST age as an idle connection.

If a server is added as a peer of the local server during reconfiguration, then the local server SHOULD use an existing LLC-multiplexed VCC to the new peer (if one exists) as a Cache Synchronization VCC. Otherwise, a new Cache Synchronization VCC MUST be established to the new peer server. If a server is removed as a peer of the local server during reconfiguration, then the local server MUST cease synchronization communications with the other server. The Cache Synchronization VCC SHOULD NOT be released as it may be used for Control Communications as well. The VCC MUST age as an idle connection.

5.4.2 SCSP

Each LES has a set of local clients. SCSP is used to create a single distributed database, the LNNI Database, from the information local to the individual LESs. The LNNI database is composed of a collection of *cache entries*, where each cache entry refers to a single piece of information, such the registration information for a particular MAC address.

SCSP synchronizes the LNNI Database among the collection of servers. Corresponding to each cache entry is a *Client State Advertisement (CSA)* record. SCSP distributes CSAs among the servers, and the servers map the CSAs to entries in the LNNI database. CSAs are objects in the distribution protocol, cache entries are objects in the LNNI Database. Certain information on the CSA that caused the most recent update of a cache entry, such as the Originator ID, Cache Key, Sequence Number, and Remaining Lifetime, is assumed to be available in the local copy of the LNNI Database.

5.4.2.1 Flooding

For the purposes of SCSP, a server is a neighbour of another if the server is listed in the **SynchronizationPeerServerList** of the other. This definition of neighbour is used throughout Section 5.4.2. When a server is required to flood a CSA to its neighbours, the set of neighbours can be obtained from the **SynchronizationPeerServerList** variable**.** Flooding occurs as a result of a change to the local copy of the LNNI Database. Whenever the local copy of the database is changed, the local server must flood that change to all of its neighbours *except* the neighbour that transmitted the CSA that caused the change if the hop count in the CSA Summary Record is greater than 1. Note that when the LNNI Database changes as a result of a local event (ie a LE Client joining the ELAN), then the resulting CSA must be flooded to all neighbours. If the CSU Request is in response to a CSU Solicit (during the cache alignment phase), the hop count field in the CSAS frame MUST be set to one. For cache state updates, this field MUST be set to **SyncHopCount**.

When a server updates (refresh of altered data, purge or data update) an existing database entry, it must originate a CSA with the same OID and cache key, but with a sequence number that is greater than the previously used value.

5.4.2.2 CSA Identification and Instance Identification

To implement flooding procedures, each CSA, and each instance of a given CSA, must be uniquely identified. The information used for identifying a CSA is:

- Originator ID, in CSA Summary record
- Cache Key, in CSA Summary record

CSAs with identical Originator IDs and Cache Keys are considered different instances of the same CSA. The additional information required to distinguish one instance of a CSA from another instance of the same CSA is:

- CSA Sequence Number, in CSA Summary record
- CSA Remaining Lifetime, in client/server protocol specific part of the CSA record

5.4.2.3 Comparison of CSA instances

Two CSAs having the same Originator ID and Cache Key but different CSA sequence numbers are different instances of the same CSA. The comparison of two instances of the same CSA to determine which is the more recent is done in the following way:

1. The CSA instance having the larger CSA Sequence Number is considered to be more recent. Otherwise,
2. The CSA with the smaller CSA Remaining Lifetime is considered to be more recent. Otherwise,
3. The CSAs are considered identical.

If a more recent CSA is detected, the LANE Service component MUST validate the contents of the CSA as described in Sections 4.2, 5.4.2.10.3, 5.4.2.10.7, 5.4.2.10.11, 5.4.2.10.15, 5.4.2.10.19, and 5.4.2.10.23.

A server MUST ignore the contents of a less recent CSA.

5.4.2.4 Expiration of CSA's Lifetime

A CSA lifetime expires in two ways:

1. A LANE Service component purges one of its originated CSAs. Purging is frequently caused by the information in the CSA becoming invalid due to client termination and deregistration.
2. A CSA in the local copy of the LNNI Database naturally ages out as described in Sections 5.4.2.6 and 5.4.2.8.

5.4.2.5 Removing an Invalid CSA prior to Lifetime Expiration

If at any time a LANE Service component determines that a CSA that it has originated becomes invalid due to local client changes, the originating LANE Service component MAY remove the CSA from the LNNI Database (this is known as a cache entry purge). To remove an invalid CSA from the LNNI Database, the LANE Service component MUST set the Remaining Lifetime field of the CSA equal to 0 and flood the CSA to each neighbour server.

If a server choose not to initiate a cache entry purge, then the entry will age out normally from the databases of other servers.

5.4.2.6 Ageing of a CSA

LANE Service components MUST maintain the age of CSAs in the LNNI Database. Each LANE Service component MUST adjust the CSA Remaining Lifetime to reflect time elapsed since the CSA was entered in the LNNI Database.

If a LANE Service component detects that the Remaining Lifetime reaches 0 the LANE Service MUST remove the CSA from the LNNI Database. The server MUST NOT flood the CSA to each neighbour server. Each neighbour will age the entry in its own database.

5.4.2.7 Receipt of a Cache Entry Purge

Receiving a CSA record containing a Remaining Lifetime value of 0 indicates that the cache entry is being deleted from the LNNI Database.

If a LANE Service component receives a CSA with the Remaining Lifetime field equal to 0, then the cache entry MUST be removed from the LNNI Database and MUST be flooded to all neighbour servers. Additional steps may be necessary as described in Sections 5.4.2.10.4, 5.4.2.10.8, 5.4.2.10.12, 5.4.2.10.16, 5.4.2.10.20, and 5.4.2.10.24.

5.4.2.8 Computing the Remaining Lifetime of a CSA

The remaining lifetime of a CSA is determined based on the value of the Remaining Lifetime in the CSA when it was installed in the LNNI Database and the elapsed time since its installation. If the initial value of the Remaining Lifetime is less than or equal to the elapsed time, then the Remaining Lifetime in the CSA MUST be set to 0 and procedures for Ageing a CSA MUST be followed. Otherwise, the value of the Remaining Lifetime is computed by subtracting the elapsed time from the initial value of the Remaining Lifetime in the CSA.

5.4.2.9 Receiving a LNNI CSA

If a server receives a CSA from a neighbour server, the server MUST first determine if this CSA is a less recent instance of an existing CSA as described in Section 5.4.2.3.

If the LNNI CSA is valid and results in an update or change to the database, the LE Server MUST synchronize (flood) the CSA with neighbouring servers as described in Section 5.4.2.1.

5.4.2.10 Synchronizing with Neighbouring Servers

Servers establish Cache Synchronization VCCs with each neighbour as described in Section 5.4.1.1. These VCCs are used for synchronizing LE Client data using SCSP. A server MUST follow the cache synchronization procedures defined below.

5.4.2.10.1 Originating a LES Refresh CSA

The LE Server Control Plane topology is generated from database information received in the LNNI Synchronization Plane. Each LES advertises its Server ID and LLC-muxed ATM address (**ServerMuxedAtmAddress**) using the LES Refresh CSA Record.

An LES MUST refresh its own LES Refresh CSA records with each neighbour server in the **SynchronizationPeerServerList** by originating an LES Refresh CSA at least once every (**ServerRefreshLifetime/2)** seconds and flooding it to each neighbour server.

An LES MUST refresh its own LES Refresh CSA records with each neighbour server in the **SynchronizationPeerServerList** by originating an LES Refresh CSA immediately that its status changes from “not suffering resource starvation” to “suffering resource starvation”.

5.4.2.10.2 Purging a LES Refresh CSA

An LES may purge all of its associated information from the LNNI database by synchronizing a LES Refresh CSA with a Remaining Lifetime of 0 with its neighbor servers. An LES MUST return to the initial state after synchronizing this CSA with its neighbors.

5.4.2.10.3 Validating a LES Refresh CSA

An LES Refresh CSA is valid if

1. any existing LES cache entries with the same ATM address have an Originator ID less than or equal to that of the CSA, and
2. any cache entries with an equal ATM address and OID have the same Cache Key as the CSA.

If the CSA is valid, then the server MUST remove all LES cache entries with a strictly smaller Originator ID. If there exists an entry with a matching OID, ATM address, and Cache Key, then the server MUST update the Sequence Number, Remaining Lifetime, Operational Status and TLV set of the cache entry. If there is no LES cache entry with a matching OID, ATM address and Cache Key, then the server MUST install a new cache entry.

The LEC-ID Range TLV(s) can be used to detect configuration errors (i.e. overlapping LEC-ID range allocation).

If the LES Refresh CSA received indicates a status change to “operational” or “not resource starved”, a LES must terminate all local clients which list the LES which originated the CSA as their preferred LES.

5.4.2.10.4 LES Refresh Cache Entry Removal

When a server removes a LES cache entry from its copy of the LNNI Database, it MUST remove all LE Client Join cache entries associated with that LES. The removal of LE Client Join CSAs MUST NOT result in the flooding of additional CSAs to neighbor servers.

5.4.2.10.5 Originating a SMS Refresh CSA

Each SMS must advertise its operational status via the SMS Refresh CSA. This SMS information is valid as long as the Remaining Lifetime value in the CSA is NOT 0.

An SMS MUST refresh its own SMS Refresh CSA records with each neighbour server in the **SynchronizationPeerServerList** by originating an SMS Refresh CSA at least once every (**ServerRefreshLifetime/2**) seconds and flooding it to each neighbour server.

An SMS MUST refresh its own SMS Refresh CSA records with each neighbour server in the **SynchronizationPeerServerList** by originating an SMS Refresh CSA immediately that its status changes from “not suffering resource starvation” to “suffering resource starvation”.

5.4.2.10.6 Purging a SMS Refresh CSA

An SMS may purge all of its associated information from the LNNI database by synchronizing a SMS Refresh CSA with a Remaining Lifetime of 0 with its neighbor servers. An SMS MUST return to the initial state after synchronizing this CSA with its neighbors.

5.4.2.10.7 Validating a SMS Refresh CSA

An SMS Refresh CSA is valid if any SMS cache entries with an equal OID have the same Cache Key.

If there exists a SMS cache entry with a matching OID and Cache Key, then the server MUST update the Sequence Number, Remaining Lifetime, Operational Status and TLV set of the cache entry. If there is no SMS cache entry with a matching OID and Cache Key, then the server MUST install a new cache entry.

5.4.2.10.8 SMS Cache Entry Removal

When a server removes a SMS cache entry from its copy of the LNNI Database, it MUST remove all SMS Multicast Registration cache entries associated with that SMS. The removal of the SMS Multicast Registration CSAs MUST NOT result in the flooding of additional CSAs to neighbor servers. When a SMS cache entry is

removed from the a LES's LNNI database and the SMSs in distributed mode (**SmsModeOfOperation**), the LES MUST reassign all local LE Clients that were assigned to that SMS as receivers to a new (operating) SMS. This can be done by refreshing relevant LE Client Multicast Registration CSAs (Section 5.4.2.10.17) with the new SMS **ServerId**.

5.4.2.10.9 Originating an LE Client Join CSA

If a local LE Client successfully joins the ELAN, the LE Server MUST originate an LE Client Join CSA and synchronize this CSA with each neighbouring LE Server.

An LES MUST refresh its LE Client Join CSA records with each neighbour server in the **SynchronizationPeerServerList** by originating an LE Client Join CSA at least once every (**ClientJoinLifetime/2**) seconds and flooding it to each neighbour server.

5.4.2.10.10 Purging an LE Client Join CSA

If a local LE Client terminates, the LE Server MUST remove the LE Client Join cache entry from its local copy of the LNNI Database, originate an LE Client Join CSA for the LE Client with the Remaining Lifetime field set to 0, and synchronize this CSA with each neighbouring server.

5.4.2.10.11 Validating an LE Client Join CSA[20]

An LE Client Join CSA is valid if

1. any existing LE Client Join cache entries with the same ATM address have an Originator ID less than or equal to that of the CSA, and
2. any cache entries with an equal ATM address and OID have the same Cache Key and LECID as the CSA.

If the CSA is valid, then the server MUST remove all LE Client Join cache entries with a strictly smaller Originator ID. If there exists an entry with a matching OID, ATM address, LECID, and Cache Key, then the server MUST update the Sequence Number, Remaining Lifetime, and TLV set of the cache entry. If there is no LE Client Join cache entry with a matching OID, ATM address, LECID, and Cache Key, then the server MUST install a new cache entry.

If a Join Cache Entry is removed that corresponds to a local LE Client, the LES MUST terminate this LE Client.

If the LE Client Join CSA is not valid, the LES MUST NOT make any changes to the cache based on the CSA.

5.4.2.10.12 LE Client Join Cache Entry Removal

When a server removes a LE Client Join cache entry from its copy of the LNNI Database, it MUST remove all LE Client Unicast Registration cache entries and LE Client Multicast Registration cache entries associated with this LE Client. The removal of the registration cache entries MUST NOT result in the flooding of additional CSAs to neighbour servers.

[20] Section 6.1.2.11 in [5] indicates that when there is a unicast LAN destination register conflict, the first LE Client to register is "the winner." The rules in this section implement a concept of "first" that is a combination of timing and the lower numerical value of the LES identifier (OID).

5.4.2.10.13 Originating an LE Client Unicast Registration CSA

If a local LE Client successfully registers a unicast MAC address, either during the Join Phase or using the LE_REGISTER protocol, the LE Server MUST originate an LE Client Unicast Registration CSA and synchronize this CSA with each neighbouring LE Server.

An LES MUST refresh the LE Client Unicast Registration CSA records with each neighbour server in the **SynchronizationPeerServerList** by originating an LE Client Unicast Registration CSA at least once every (**RegistrationLifetime/2**) seconds and flooding it to each neighbour server.

5.4.2.10.14 Purging an LE Client Unicast Registration CSA

If a local LE Client successfully unregisters a unicast MAC address, the LE Server MUST originate an LE Client Unicast Registration CSA for the unicast MAC address with the Remaining Lifetime field set to 0 and synchronize this CSA with each neighbouring server.

5.4.2.10.15 Validating an LE Client Unicast Registration CSA

An LE Client Unicast Registration CSA is valid if

1. any existing LE Client Unicast Registration cache entries with the same target ATM address have an Originator ID less than or equal to that of the CSA, and
2. any existing LE Client Unicast Registration cache entries with the same target LAN destination have an Originator ID less than or equal to that of the CSA, and
3. any cache entries with an equal ATM address, LAN destination, and OID have the same Cache Key and LECID as the CSA.

If the CSA is valid, then the server MUST remove all LE Client Unicast Registration cache entries with a matching ATM address or LAN destination, and a strictly smaller Originator ID. If there exists an entry with a matching OID, ATM address, LAN destination, LECID, and Cache Key, then the server MUST update the Sequence Number, Remaining Lifetime, and TLV set of the cache entry. If there is no cache entry with a matching OID, ATM address, LAN destination, LECID, and Cache Key, then the server MUST install a new cache entry.

If a Unicast LNNI Database Entry is removed that corresponds to a local LE Client, a LES MUST terminate this LE Client. Termination is required because there is no way to asynchronously inform an LE Client that a previously successful registration has become invalid.

If the LE Client Unicast Registration CSA is not valid, the server MUST NOT make any changes to the cache based on the CSA.

5.4.2.10.16 Removal of LE Client Unicast Registration Cache Entries

The removal of registration cache entries MUST NOT result in other changes to the local copy of the LNNI Database.

5.4.2.10.17 Originating an LE Client Multicast Registration CSA

If an LE Client successfully registers a multicast MAC address using the LE_REGISTER protocol, the LE Server MUST originate an LE Client Multicast Registration CSA and synchronize this CSA with each neighbouring LE Server.

Additionally, if there is an SMS serving the registered multicast MAC address and the SMSs are operating in distributed mode, then the LES MUST assign the LE Client to an SMS before synchronizing its registration data, and MUST include this assignment in the multicast registration CSA. The algorithm used by LESs in assigning

LE Clients to SMSs is outside the scope of this specification, but the assignment must satisfy (a) the assigned SMS must be operational (i.e. a SMS Refresh cache entry exists), and (b) the assigned SMS must be serving the registered MAC address (i.e. a SMS Multicast Registration cache entry exists for the SMS/MAC combination).

An LES MUST refresh the LE Client Multicast Registration CSA records with each neighbour server in the **SynchronizationPeerServerList** by originating an LE Client Unicast Registration CSA at least once every (**RegistrationLifetime/2**) seconds and flooding it to each neighbour server. A LES MAY alter the assignment of the multicast registration from one SMS to another in subsequent refreshes.

5.4.2.10.18 Purging an LE Client Multicast Registration CSA

If a local LE Client successfully unregisters a multicast MAC address, the LE Server MUST originate an LE Client Multicast Registration CSA for the multicast MAC address with the Remaining Lifetime field set to 0 and synchronize this CSA with each neighbouring LE Server.

5.4.2.10.19 Validating an LE Client Multicast Registration CSA

An LE Client Multicast Registration CSA is valid if

1. any existing LE Client Multicast Registration cache entries with the same target ATM address have an Originator ID less than or equal to that of the CSA, and
2. any cache entries with an equal ATM address and OID have the same Cache Key and LECID as the CSA.

If the CSA is valid, then the server MUST remove all LE Client Multicast Registration cache entries with a matching ATM address and a strictly smaller Originator ID. If there exists an entry with a matching OID, ATM address, LECID, and Cache Key, then the server MUST update the Sequence Number, Remaining Lifetime, and TLV set of the cache entry. If there is no cache entry with a matching OID, ATM address, LECID, and Cache Key, then the server MUST install a new cache entry.

If a Multicast LNNI Database Entry is removed that corresponds to a local LE Client, a LES MUST terminate this LE Client. Termination is required because there is no way to asynchronously inform an LE Client that a previously successful registration has become invalid.

If the LE Client Multicast Registration CSA is not valid, the server MUST NOT make any changes to the cache based on the CSA.

Upon detecting a multicast address not yet served by another SMS, a LES SHOULD determine whether local LE Clients which have been assigned to the BUS for that multicast address should be reassigned to the SMS. If the LES chooses to reassign a local LEC, it MUST issue a LE_NARP to that client indicating that the multicast address is now served by the SMS's ATM address. A LES MAY perform such a determination at any point to reassign LE Clients to particular SMSs.

5.4.2.10.20 Removal of LE Client Multicast Registration Cache Entries

The removal of registration cache entries MUST NOT result in other changes to the local copy of the LNNI Database.

5.4.2.10.21 Originating an SMS Multicast Registration CSA

To advertise a multicast MAC address that it serves, an SMS MUST originate an SMS Multicast Registration CSA and synchronize this CSA with each neighbouring server.

An SMS MUST refresh its SMS Multicast Registration CSA records with each neighbour server in the **SynchronizationPeerServerList** by originating an SMS Multicast Registration CSA at least once every (**RegistrationLifetime/2**) seconds and flooding it to each neighbour server.

5.4.2.10.22 Purging a SMS Multicast Registration CSA

A SMS may advertise that it is no longer serving a particular multicast address by originating a SMS Multicast Registration CSA with Remaining Lifetime equal to 0.

5.4.2.10.23 Validating an SMS Multicast Registration CSA

An SMS Multicast Registration CSA is valid if

1. any existing SMS Multicast Registration cache entries with the same target ATM address have an Originator ID less than or equal to that of the CSA, and
2. any cache entries with an equal ATM address and OID have the same Cache Key as the CSA.

If the CSA is valid, then the server MUST remove all SMS Multicast Registration cache entries with a matching ATM address and a strictly smaller Originator ID. If there exists an entry with a matching OID, ATM address, and Cache Key, then the server MUST update the Sequence Number, Remaining Lifetime, and TLV set of the cache entry. If there is no cache entry with a matching OID, ATM address, and Cache Key, then the server MUST install a new cache entry.

If the LE Client Multicast Registration CSA is not valid, the server MUST NOT make any changes to the cache based on the CSA.

5.4.2.10.24 Removal of SMS Multicast Registration Cache Entries

The removal of registration cache entries MUST NOT result in other changes to the local copy of the LNNI Database.

5.5 Control Communications

Each LES must be able to forward certain LAN Emulation control frames to all others LESs in an ELAN, as described in Section 2.4.5. These frames are forwarded over Control Coordinate connections. Each LES establishes a Control Coordinate connection to each other LES servicing a given ELAN, providing a fully connected mesh between an ELAN's LESs. Control Coordinate connections utilize LLC-multiplexed VCCs, which may be shared with other LLC-multiplexed traffic, namely with Cache Synchronization connections, described below.

5.5.1 Establishing Control Coordinate VCCs

Each LES MUST establish and maintain a Control Coordinate connection to each other LES in the ELAN (i.e. each LES in the LNNI Database).

For each Control Coordinate connection to a peer LES, the local LES SHOULD use an existing LLC-multiplexed VCC to this peer LES if such a connection already exists or is in the process of being established. Otherwise, a new LLC-multiplexed VCC MUST be established.

For Control Coordinate connections which require the creation of a new VCC, the new VCC MAY be set up as either a new point-to-point VCC to the peer LES or as a leaf in a point-to-multipoint call at the choice of the LES implementation.

5.5.2 Handling VCC Setup Failure

If the LLC-multiplexed VCC connection attempt fails, the calling LES MUST attempt to establish the retry continuously based on the logic in section 5.1.2.1.7 until either a connection is successfully established or until such time as the LES is removed from the LNNI Database.

5.5.3 B-LLI Codepoint for Control Coordinate VCCc

Control Coordinate connections MUST use LLC-multiplexed VCCs. All other call parameters signalled by the LES when creating new VCCs for Control Coordinate connections are those defined in Section 3 (Connection Management Services).

5.5.4 Accepting Incoming Control Coordinate Connections

An LES MUST accept a UNI signalling request to establish an LLC-multiplexed VCC to its LLC-multiplexed ATM address (**ServerMuxedAtmAddress**) with the assumption that this VCC will be used for Cache Synchronization and/or Control Coordinate traffic. An LES MAY refuse a UNI signalling request if it contains call parameters other than as defined in Section 3 (Connection Management Services).

The server MAY release this connection if, upon completion, no valid synchronization or control communications use the VCC within the **IdleLaneControlVccTimeout** AND the calling party address does not match a configured server address in its **SynchronizationPeerServerList**.

5.5.5 Handling Duplicate Control Coordinate Connections

When an LES receives an incoming LLC-multiplexed VCC setup request from an ATM address to which it already has a connection, it SHOULD accept the request and apply the ageing rules for duplicate Data Direct VCCs found in Section 8.1.13 of [5] for ageing out the appropriate duplicate VCC.

5.5.6 Handling Release of Control Coordinate Connections

If an LLC-multiplexed VCC is released by a peer LES, resulting in the loss of all Control Coordinate connectivity to the peer LES, the local LES MUST attempt to establish a new LLC-multiplexed VCC to the LES which initiated the release. If the setup request fails, the calling server MUST retry continuously based on the logic in section 5.1.2.1.7 until either a connection is successfully established, or such time as the called server is removed from the LNNI Database.

5.5.7 Removal of an LES

If a peer LES is removed from the LNNI Database , the local LES MUST stop sending any Control Coordinate traffic to that LES. If the VCC which was being used for Control Coordinate traffic to this LES is still being used for synchronization communications or for some other LLC-muxed protocol, then the LES MUST stop using the VCC for control communications. Otherwise, the VCC MAY immediately be released.

An LES can remove itself from the ELAN by synchronizing an LES Refresh CSA with a Remaining Lifetime of zero to each of its neighbours. Once this is complete it MUST stop sending any control or synchronization traffic to any peer LES. See Section 5.4.2.10.2.

5.6 Control Frame Processing

5.6.1 LE Client Join and Terminate Protocol

LE Client Joins and Terminates are detected by the ingress LES across the LUNI. If the Join or Terminate operation is valid, the LES updates its cache and this change is synchronized with the other LESs, using SCSP.

5.6.1.1 LE Client Join

When an LES receives an LE_JOIN_REQUEST from an LE Client, the LES validates the join request according to the LUNI specification[5]. If an error is detected, the LES MUST send an unsuccessful LE_JOIN_RESPONSE to the LE Client as specified in[5]. If no error is detected, the LES MUST add the LE Client's state to its LNNI Database (including any implicit registration data). The LES MUST then send a successful LE_JOIN_RESPONSE to the LE Client and synchronize the change in its cache with the other LESs.

When an LES receives an LE_JOIN_REQUEST which contains a Preferred LES TLV, and that preferred LES is active and not resource starved as signified by the presence of a LES Refresh cache entry identifying that LES, the LES MUST send an unsuccessful LE_JOIN_RESPONSE (i.e. STATUS = Access Denied) to the LE Client.

5.6.1.1.1 Join Validation – LES View

An LES MAY validate LE Clients with the LECS prior to allowing the LE Client to join an ELAN. When an LES is validating LE Clients then, when an LES receives an LE_JOIN_REQUEST from an LE Client, the LES MUST

- Generate a LNNI_VALIDATE_REQUEST. The LNNI_VALIDATE_REQUEST is identical to the LE_JOIN_REQUEST except for:
  - the OPCODE value and,
  - the LES's non-multiplexed ATM address is placed in the TARGET-ATM-ADRESS field
- Send it to the LECS.

Upon receiving a successful LNNI_VALIDATE_RESPONSE , the LES MUST continue to process the LE_JOIN_REQUEST as defined in [5]. If the STATUS of the LNNI_VALIDATE_REQUEST is "Access Denied", the LES MUST send the LE Client a unsuccessful LE_JOIN_RESPONSE (i.e. STATUS = "Access Denied").

5.6.1.1.2 Join Validation – LECS View

Upon receiving an LE_VALIDATE_REQUEST the LECS MUST check to see if the LE Client that is attempting to join the ELAN is one which is permitted to do so via the sending LES.

The LECS responds by sending an LE_VALIDATE_RESPONSE with the STATUS field set to:

- "Success" if the ATM address is acceptable for an LE Client which wishes to join the ELAN via that LES or
- "Access Denied" which indicates that the LE Client has failed the validation

The LECS MUST only validate that the LE Client is permitted to join the ELAN via that LES. Any other processing related to the LE_JOIN_REQUEST will be performed by the LES as described in the LUNI specifications [5].

5.6.1.1.3 Join Validation – Frame Format

Table 5-9 Join Validation Frame Format

Offset	Size	Name	Function
0	2	MARKER	Control Frame = X"FF00"
2	1	PROTOCOL	ATM LAN Emulation protocol = X"01"
3	1	VERSION	ATM LAN Emulation protocol version = X"01"
4	2	OP-CODE	Type of request. See Table 4-2
6	2	STATUS	Always X"0000" in requests. See the LUNI specifications for a list of supported values in a response.
8	4	TRANSACTION-ID	Arbitrary value supplied by the requester and returned by the responder.
12	2	REQUESTER-LEC-ID	Always X"0000" in requests, ignored on response.
14	2	FLAGS	Copy from LE_JOIN_REQUEST for request , ignored on response.
16	8	SOURCE-LAN-DESTINATION	Copy from LE_JOIN_REQUEST for request , ignored on response.
24	8	TARGET-LAN-DESTINATION	Always "not present" when sent, ignored on receipt.
32	20	SOURCE-ATM-ADDRESS	Copy from LE_JOIN_REQUEST for request , ignored on response.
52	1	LAN-TYPE	Copy from LE_JOIN_REQUEST for request , ignored on response.
53	1	MAXIMUM-FRAME-SIZE	Copy from LE_JOIN_REQUEST for request , ignored on response.
54	1	NUMBER-TLVS	Copy from LE_JOIN_REQUEST for request , ignored on response.
55	1	ELAN-NAME-SIZE	Copy from LE_JOIN_REQUEST for request , ignored on response.
56	20	TARGET-ATM-ADDRESS	Non-multiplexed ATM Address of the LES (**S1** in [5]).
76	32	ELAN-NAME	Copy from LE_JOIN_REQUEST for request , ignored on response.
108	4	ITEM_1-TYPE	Copy from LE_JOIN_REQUEST for request , ignored on response.
112	1	ITEM_1-LENGTH	Copy from LE_JOIN_REQUEST for request , ignored on response.
113	Variable	ITEM_1-VALUE	Copy from LE_JOIN_REQUEST for request , ignored on response.
		Etc.	

5.6.1.2 LE Client Terminate

An LE Client may voluntarily terminate its connection to the LES or an LES may force an LE Client to terminate, as described in the following two subsections.

5.6.1.2.1 LE Client Initiated Terminate

When an LES detects the termination of an LE Client, as defined in [5], the LES MUST remove the client and all associated registration information from its LNNI Database as described in Section 5.4.2.10.12. The LES MUST then synchronize the change in its cache with the other LESs in the ELAN.

5.6.1.2.2 Server Initiated Terminate

When an LES receives an LES Refresh CSA identifying an LES not currently in the LNNI Database with a Remaining Lifetime greater than zero, the LES MUST compare the ATM Address of the LES that originated the CSA with the Preferred LES Address of each local LE Client. For each local LE Client whose Preferred LES Address matches the CSA originator's address, the Control Direct VCC to that LE Client MUST be released, and the LES MUST remove all associated registration information from its LNNI Database. The LES MUST then synchronize the change in its cache with the other LESs.

5.6.2 LE Client Register and Unregister Protocol

Client Registers and Unregisters are sent to the ingress LES across the LUNI. If a Register or Unregister operation is valid, the LES updates its cache and this change is synchronized with the other LESs, using SCSP.

5.6.2.1 LE Client Register

When an LES receives an LE_REGISTER_REQUEST from an LE Client, the LES validates the register request according to[5]. If an error is detected, the LES MUST send an unsuccessful LE_REGISTER_RESPONSE to the LE Client as specified in [5].If the registration is for a multicast address and the SMSs are operating in distributed mode, the LES MUST select an SMS which is not resource starved to serve that LE Client directly.

If no error is detected, the LES MUST add the LE Client's state to its LNNI Database. The LES MUST send a successful LE_REGISTER_RESPONSE to the LE Client and synchronize the change in its cache with the other LESs.

5.6.2.2 LE Client Unregister

When an LES receives an LE_UNREGISTER_REQUEST from an LE Client, the LES MUST remove all associated registration information from its LNNI Database. The LES MUST then synchronize the change in its cache with the other LESs.

5.6.3 Verification Protocol

5.6.3.1 Receiving an LE_VERIFY_REQUEST

When an LES receives an LE_VERIFY_REQUEST from an LE Client, it MUST NOT forward it to any other LES. Based on its LNNI Database, it sends an LE_VERIFY_RESPONSE which either confirms or denies the ATM Address from the request as a valid BUS or SMS ATM address for the ELAN.

5.6.4 LE Client Address Resolution: ARP Requests and Responses

5.6.4.1 Receiving an LE_ARP_REQUEST across the LUNI

When an LES receives an LE_ARP_REQUEST across the LUNI, and the target LAN destination is registered, the LES may choose to answer the request itself or to forward the request on.

- If the LES chooses to answer the request, then it MUST transmit a successful LE_ARP_RESPONSE to at least the requesting LE Client
- If the LE_ARP_REQUEST is for a multicast destination, the LES MUST respond with the ATM Address of an operational SMS serving that MAC address that is not resource starved (if one exists), or the ATM address of a BUS.
- If the LES chooses not to answer an LE-ARP_REQUEST for a unicast destination, then:

- If the target LAN destination corresponds to a local LE Client the LES MUST forward the LE_ARP_REQUEST to at least that local LE Client.
- If the LAN destination corresponds to a non-local LE Client, then the LES MUST forward the LE_ARP_REQUEST to at least the LES at which the target LAN destination is registered.

If the target LAN destination is not registered, then the LES MUST forward the LE_ARP_REQUEST to all other LESs and to at least all local proxy LE Clients.

If the LES receives a Targetless LE_ARP_REQUEST, then the LES MUST forward the LE_ARP_REQUEST to all local LE Clients and to all LE Servers.

5.6.4.2 Receiving an LE_ARP_REQUEST across the LNNI

When an LES receives an LE_ARP_REQUEST from another LES, and the target LAN destination is registered by a local LE Client, the LES may choose to answer the request itself or to forward the request on.

If the LES chooses to answer the request, then it MUST transmit a successful LE_ARP_RESPONSE to at least the requesting LES. If the LES chooses not to answer the request, then the LES MUST forward the LE_ARP_REQUEST to at least the local LE Client that registered that target LAN Destination.

The LES MUST NOT answer the request if the target LAN destination is registered by a non-local LE Client.

If the target LAN destination is not registered, then the LES MUST forward the LE_ARP_REQUEST to at least all local proxy LE Clients.

If the LES receives a Targetless LE_ARP_REQUEST, then the LES MUST forward the request to all local LE Clients.

5.6.4.3 Receiving an LE_ARP_RESPONSE across the LUNI

When an LES receives an LE_ARP_RESPONSE from an LE Client, and the requesting LE Client (as identified by the LECID), is local, then the LES MUST forward the LE_ARP_RESPONSE to at least the requesting LE Client. Otherwise, the LES MUST forward the LE_ARP_RESPONSE to at least the LES for which the requesting LE Client is local.

5.6.4.4 Receiving an LE_ARP_RESPONSE across the LNNI

When an LES receives an LE_ARP_RESPONSE from another LES, and the requesting LE Client (as identified by the LECID) is local, then the LES MUST forward the LE_ARP_RESPONSE to at least the requesting LE Client. Otherwise, the LES MAY forward the LE_ARP_RESPONSE to any set of local LE Clients or may discard the response.

5.6.5 LE Client NARP Requests

LE_NARP_REQUEST forwarding between LESs is done on the Control Coordinate VCCs.

5.6.5.1 Receiving an LE_NARP_REQUEST across the LUNI

When an LES receives an LE_NARP_REQUEST across the LUNI, the LES MUST forward the LE_NARP_REQUEST to all local LE Clients and to all other LESs.

5.6.5.2 Receiving an LE_NARP_REQUEST across the LNNI

When an LES receives an LE_NARP_REQUEST from another LES, the LES MUST forward the LE_NARP_REQUEST to all local LE Clients.

5.6.6 LE Client Topology Change

LE_TOPOLOGY_REQUEST forwarding between LESs is done on the Control Coordinate VCCs.

5.6.6.1 Receiving an LE_TOPOLOGY_REQUEST across the LUNI

When an LES receives an LE_TOPOLOGY_REQUEST across the LUNI, the LES MUST forward the LE_TOPOLOGY_REQUEST to all local LE Clients and to all other LE Servers.

5.6.6.2 Receiving an LE_TOPOLOGY_REQUEST across the LNNI

When an LES receives an LE_TOPOLOGY_REQUEST from another LES, the LES MUST forward the LE_TOPOLOGY_REQUEST to all local LE Clients.

5.6.7 Flush Protocol

5.6.7.1 Flush Protocol - LES View

LE_FLUSH_RESPONSE forwarding between LESs is done on the Control plane.

5.6.7.1.1 Receiving an LE_FLUSH_RESPONSE across the LUNI

When an LES receives an LE_FLUSH_RESPONSE across the LUNI and the LECID in the LE_FLUSH_RESPONSE is known and indicates a local LE Client, then the LES MUST forward the LE_FLUSH_RESPONSE to at least the requesting LE Client.

If the LECID is known and indicates a non-local LE Client, then the LES MUST forward the LE_FLUSH_RESPONSE to at least the LES for which the requesting LE Client is local.

If the LECID is unknown, then the LES MUST forward the LE_FLUSH_RESPONSE to all other LESs and to at least all local proxy LE Clients.

5.6.7.1.2 Receiving an LE_FLUSH_RESPONSE across the LNNI

When an LES receives an LE_FLUSH_RESPONSE from another LES and the LECID in the LE_FLUSH_RESPONSE is known and indicates a local LE Client, then the LES MUST forward the LE_FLUSH_RESPONSE to at least the requesting LE Client.

If the LECID is known and indicates a non-local LE Client, then the LES MAY discard the response or MAY forward the LE_FLUSH_RESPONSE to any set of local LE Clients.

If the LECID is unknown, then the LES MUST forward the LE_FLUSH_RESPONSE to at least all local proxy LE Clients.

5.6.7.2 Flush Protocol - BUS View

5.6.7.2.1 Receiving an LE_FLUSH_REQUEST across the LUNI

When a BUS receives an LE_FLUSH_REQUEST across the LUNI and the target LAN destination corresponds to a local LE Client, the BUS MUST forward the LE_FLUSH_REQUEST to at least the destination LE Client. If the LAN destination corresponds to a non-local LE Client, the BUS MUST forward the LE_FLUSH_REQUEST to at least the BUS for which the LAN destination corresponds to a local LE Client.

If the TARGET-LAN-DESTINATION is present in that LE_FLUSH_REQUEST the BUS MUST distribute the LE_FLUSH_REQUEST on the VCC that would be used to forward data frames (see Section 5.7.2) to the target LAN Destination specified in the LE_FLUSH_REQUEST frame.

When a BUS receives an LE_FLUSH_REQUEST across the LUNI with an unspecified target LAN destination, then the BUS MUST forward the LE_FLUSH_REQUEST to at least the BUS for which the target ATM address in the LE_FLUSH_REQUEST corresponds to a local LE Client.

5.6.7.2.2 Receiving an LE_FLUSH_REQUEST across the LNNI

When a BUS receives an LE_FLUSH_REQUEST from another BUS, the BUS MUST NOT forward the LE_FLUSH_REQUEST to any other BUS.

When a BUS receives an LE_FLUSH_REQUEST from another BUS and the target LAN destination corresponds to a local LE Client, the BUS MUST forward the LE_FLUSH_REQUEST to at least the destination LE Client. If the LAN destination corresponds to a non-local LE Client, the BUS MAY discard the request or forward it to any set of local LE Clients.

When a BUS receives an LE_FLUSH_REQUEST from another BUS with an unspecified target LAN destination, and the target ATM address corresponds to a local LE Client, then the BUS MUST forward the LE_FLUSH_REQUEST to at least the destination LE Client. If the target LAN destination is not a local LE Client, the BUS MAY discard the request or forward it to any set of local LE Clients.

5.7 BUS Data Communications

The primary responsibility of the Broadcast and Unknown Server (BUS) is to forward data frames between clients who have yet to establish a Data Direct VCC, and to forward multicast frames which are not handled by any SMS. In an ELAN with multiple BUSs, the BUSs must forward frames to and from other BUSs as well as to and from clients. As defined in Section 2, a BUS's *local clients* are those clients directly attached to the BUS via a Multicast Send VCC. A BUS forwards frames between its local clients and other BUSs, which in turn forward to their local clients.

5.7.1 BUS Connections

BUSs must be interconnected with a logical mesh of VCCs to accomplish their forwarding objectives. The VCCs between BUSs are referred to as Multicast Forward VCCs. This section defines how Multicast Forward VCCs are signalled, established, and maintained.

5.7.1.1 Learning About a Neighbour BUS

BUS information, specifically the BUS ATM address, is distributed as part of the LNNI Database. When an LES learns of a new or failed BUS serving the ELAN, it MUST notify its associated BUS of the new/failed neighbour. The method of notification is outside the scope of this specification. Upon notification, the BUS MUST attempt to

connect to the newly discovered BUS as described in Section 5.7.1.2, and it MUST discontinue connection attempts to any failed BUS as described in Section 5.7.1.7.

5.7.1.2 Establishing Multicast Forward VCCs

When a BUS is informed of a new BUS serving the ELAN, it MUST attempt to establish a Multicast Forward VCC to that BUS using the signalling parameters described in Section 5.7.1.3.

5.7.1.3 BLLI Codepoint and Signalling Parameters

The signalling parameters for a BUS-BUS Multicast Forward VCC are defined in Section 3. In particular, BUS-BUS Multicast Forward VCCs do not use LLC encapsulation

5.7.1.4 Number of Multicast Forwards

A BUS MAY establish multiple Multicast Forward VCCs to accomplish its forwarding objectives in an optimized manner. This may result in a single Multicast Forward VCC to all other BUSs, in multiple Multicast Forward VCCs to distinct sets of BUSs, in multiple overlapping Multicast Forward VCCs, or in any other combination of Multicast Forward VCCs.

5.7.1.5 Accepting Multiple Multicast Forwards

A BUS MUST accept multiple Multicast Forward VCCs from another BUS. If the calling address of an incoming Multicast Forward VCC SETUP is not the ATM address of a BUS or SMS in the LNNI Database, the SETUP MUST be rejected.

5.7.1.6 Handling Connection Failures

If the establishment of a Multicast Forward VCC to another BUS fails, then the BUS MUST retry the SETUP as described in Section 5.1.2.1.7.. The BUS MUST periodically attempt to establish the connection until either the connection succeeds, or until the neighbour BUS is removed from the distributed SCSP database.

5.7.1.7 Handling Released Data Connection

If a Multicast Forward VCC is released, the BUS MUST attempt to re-establish the connection. The BUS MUST handle failed connection attempts as described in Section 5.7.1.6.

5.7.1.8 Releasing Data Connections

When a BUS terminates its service of an ELAN, it MUST release its BUS-BUS Multicast Forward VCCs. The VCCs MUST be released with UNI cause “Normal, unspecified.” If a BUS has a Multicast Forward VCC to another BUS which is removed from the LNNI Database, then the BUS MUST release the Multicast Forward to the other BUS with UNI cause “Normal, unspecified.”

5.7.1.9 Rejecting Data Connections

A BUS MAY reject an incoming Multicast Forward VCC from a calling ATM address which is not listed as the ATM address of a BUS or SMS in the LNNI Database. The rejection MUST be signalled with UNI cause “Call rejected.”

5.7.2 Data Forwarding Rules

This section describes the forwarding rules governing the BUS’ handling of data frames.

5.7.2.1 Minimal Forwarding Rules

In this section, we define the minimal set of destinations to which a BUS must forward a received frame and the minimal set of destinations to which a BUS must not forward a frame. BUSs may forward frames any destination set in this range. The forwarding rules are phrased in terms of the destination MAC address and the entity that forwarded the frame to the BUS.

5.7.2.1.1 General Forwarding Rules

5.7.2.1.1.1 Frame Duplication

A BUS MUST NOT forward a frame in such a way that any destination receives multiple copies of the frame.

5.7.2.1.1.2 BUS to BUS Forwarding

A BUS MUST NOT forward a frame received from an SMS or a BUS to another BUS.

5.7.2.1.1.3 BUS to SMS Forwarding

A BUS MUST NOT forward frames to an SMS.

5.7.2.1.2 Unicast Data Forwarding

5.7.2.1.2.1 Receiving Unicast Data across LUNI

When a BUS receives a unicast frame from an LE Client *and* the LAN destination is registered and corresponds to a local LE Client, the BUS MUST forward the unicast frame to at least the destination LE Client. If the LAN destination is registered but corresponds to a non-local LE Client, then the BUS MUST forward the unicast frame to at least the BUS for which the LAN destination corresponds to a local LE Client.

If the LAN destination is not registered, then the BUS MUST forward the unicast frame to at least all other BUSs and all local proxy LE Clients.

5.7.2.1.2.2 Receiving Unicast Data from BUS

When a BUS receives a unicast frame from another BUS and the LAN destination is registered and corresponds to a local LE Client, then the BUS MUST forward the unicast frame to at least the destination LE Client. If the LAN destination is registered but corresponds to a non-local LE Client, the BUS MAY discard the frame or it MAY forward it to any set of local LE Clients.

If the LAN destination is not registered, then the BUS MUST forward the unicast frame to at least all local proxy LE Clients.

5.7.2.1.2.3 Receiving Unicast Data from SMS

Receiving a unicast frame from an SMS is an error condition, which the BUS may or may not be able to detect. If a BUS receives a unicast frame from the SMS, it MAY discard the frame or forward it to any set of local LE Clients.

5.7.2.1.3 Multicast Data Forwarding

5.7.2.1.3.1 Receiving Multicast Data across LUNI

When a BUS receives a multicast frame from an LE Client, the BUS MUST forward the frame to all BUSs and to at least the local non-SMF LE Clients and the local SMF-LE Clients registered for that multicast destination.

5.7.2.1.3.2 Receiving Multicast Data from BUS

When a BUS receives a multicast frame from another BUS, the BUS MUST forward the multicast frame to at least the local non-SMF LE Clients and the local SMF-LE Clients registered for that multicast destination.

5.7.2.1.3.3 Receiving Multicast Data from SMS

When a BUS receives a multicast frame from an SMS, the BUS MUST forward the multicast frame to all local non-SMF LE Clients. The BUS MUST NOT forward the frame to any SMF LE Clients.

5.8 SMS Data Communications

SMSs may operate in either stand-alone or distributed mode.

5.8.1 SMS Connections

A stand-alone SMS connects to every LE Client that registers for a MAC address served by the SMS, and must connect to every BUS. A distributed SMS must be connected to every LE CLIENT assigned to that SMS for a served MAC address, and to all BUSs and to all other SMSs serving the multicast destination. The VCCs established by an SMS for data forwarding are referred to as Multicast Forward VCCs. An ELAN is served by either stand-alone or distributed mode SMSs, it cannot be served by both types of SMSs simultaneously. This section defines how the SMS Multicast Forward VCCs are signalled, established, and maintained.

5.8.1.1 Learning About Destinations

LE Client, BUS, and SMS information is distributed as part of the LNNI Database. SMSs learn of new or failed LE Clients, BUSs, and SMSs from this database.

5.8.1.2 Establishing Multicast Forward VCCs

When an SMS is informed of a new BUS serving the ELAN, it MUST attempt to establish a Multicast Forward VCC to that BUS using the signalling parameters described in Section 5.8.1.3.

5.8.1.2.1 Stand-alone SMS

A stand-alone SMS MUST NOT attempt to establish a Multicast Forward VCC to any other SMS. When a stand-alone SMS learns of a new SMF LE Client registering for a multicast address served by the SMS, it MUST attempt to establish a Multicast Forward VCC to that LE Client using the signalling parameters described in Section 5.8.1.3.

5.8.1.2.2 Distributed SMS

When a distributed SMS learns of a new SMS serving the same multicast MAC address in the ELAN, it MUST attempt to connect to the newly discovered SMS using the signalling parameters described in Section 5.8.1.3. When a distributed SMS learns of a new SMF LE Client being assigned to that SMS as a receiver for a particular multicast address, it MUST attempt to establish a Multicast Forward VCC to that LE Client using the signalling parameters described in Section 5.8.1.3.

5.8.1.3 BLLI Codepoint and Signalling Parameters

The signalling parameters for an SMS Multicast Forward VCC are identical to those of the signalling parameters for a BUS-BUS Multicast Forward as defined in Section 5.7.1.3. In particular, SMS Multicast Forward VCCs use LANE encapsulation and the LANE BLLI codepoint, and may be either point-to-point or point-to-multipoint connections. Multicast Forward VCCs MUST NOT be LLC-multiplexed.

5.8.1.4 Number of Multicast Forwards

An SMS MAY establish multiple Multicast Forward VCCs to accomplish its forwarding objectives in an optimized manner.

5.8.1.5 Accepting Multiple Multicast Forwards

A distributed SMS MUST accept multiple Multicast Forward VCCs from another SMS.

5.8.1.6 Handling Connection Failures

If the establishment of a Multicast Forward VCC fails, then the SMS MUST retry the SETUP continuously based on the logic in section 5.1.2.3.6 until either the connection succeeds, or until the destination is removed from the LNNI Database.

5.8.1.7 Handling Released Data Connection

If a Multicast Forward VCC is released, the SMS MUST attempt to re-establish the connection. The SMS MUST handle failed connection attempts as described in Section 5.8.1.6.

5.8.1.8 Releasing Data Connections

When an SMS terminates its service of an ELAN, it MUST release its Multicast Forward VCCs. The VCCs MUST be released with UNI cause "Normal, unspecified." If an SMS has a Multicast Forward VCC to a destination which is removed from the LNNI Database, then the SMS MUST release the Multicast Forward with UNI cause "Normal, unspecified." If an SMF LE Client unregisters a MAC address which results in that LE Client having no registered MAC addresses served by that SMS, the SMS MUST release the Multicast Forward VCC to the LE Client with cause "Normal, unspecified."

5.8.1.9 Rejecting Data Connections

A distributed SMS MAY reject an incoming Multicast Forward VCC from a calling ATM address which is not listed as the ATM address of an SMS in the LNNI Database. The rejection MUST be signalled with UNI cause "Call rejected." A stand-alone SMS SHOULD reject all incoming Multicast Forward VCCs with UNI cause "Call rejected."

5.8.2 Data Forwarding Rules

This section describes the forwarding rules governing the SMS' handling of data frames.

5.8.2.1 Forwarding in Stand-Alone Mode

This section defines the forwarding rules for a stand-alone SMS.

5.8.2.1.1 Receiving Multicast Data across LUNI

A stand-alone SMS MUST forward frames received from LE Clients to all SMF LE Clients registered for the destination MAC address and to all BUSs.

5.8.2.2 Forwarding in Distributed Mode

This section defines the forwarding rules for distributed SMSs.

5.8.2.2.1 Receiving Multicast Data across LUNI

A distributed SMS MUST forward frames received from LE Clients to all SMF LE Clients which are both registered for the destination MAC address and assigned to the SMS as a receiver. Additionally, a distributed SMS must forward multicast frames to all BUSs and to all other SMSs serving the multicast destination.

5.8.2.2.2 Receiving Multicast Data from SMS

A distributed SMS MUST forward frames received from SMSs to all SMF LE Clients which are both registered for the destination MAC address and assigned to the SMS as a receiver. The SMS MUST NOT forward frames received from an SMS to other SMSs or to BUSs.

6 Appendix A: Implementing a VCC-Mapped BUS

The data forwarding rules for the BUS detailed in Section 5.7.2.1 are stated in terms that imply the contents of the frame, at least the destination MAC address, are analyzed by the BUS. A BUS which inspects a frame's content is referred to as an *Intelligent BUS* (I-BUS). In fact, the forwarding rules can be satisfied without inspecting the contents of any frame, with forwarding decisions based only on the VCC that is used to deliver the frame to the BUS. A BUS that makes forwarding decisions based on only the inbound VCC is referred to as a *VCC-Mapped BUS*.

There are many ways that a VCC-Mapped BUS can be implemented and meet the rules of Section 5.7.2.1. A simple example of a BUS would contain one point-to-multipoint VCC to all the SMF-LE Clients, another point-to-multipoint VCC to all the non-SMF clients and another point-to-multipoint VCC to all the BUSs in the ELAN. While this results in a lower number of VCCs, the multicast latency is relatively high. In another approach, a BUS can make a distinction between LANEv1 non-SMF LE Clients and LANEv2 non-SMF LE Clients. This distinction is beneficial because LANEv1 LE Clients accept only one (the default) Multicast Forward VCC while the LANEv2 LE Clients accept multiple Multicast Forward VCCs, even in the non-SMF mode. With the distinction of a separate Multicast Forward VCC for only LANEv1 LE Clients, additional optimizations to improve latency are possible.

The examples in this appendix show several possible VCC-Mapped BUS implementations in various ELAN configurations. ELAN configurations with LANEv1 LE Clients only, LANEv2 LE Clients only, and a mix of LANEv1 and LANEv2 LE Clients are provided.

6.1 VCC-Mapped BUS Inputs and Outputs

BUS inputs are uniquely classified as VCCs originating from a local LE Client, another BUS, or an SMS. These VCCs carry data and or control traffic that must be distributed to the VCC-Mapped BUS outputs. VCC-Mapped BUS outputs are uniquely classified as point-to-multipoint VCCs to one or more of the following LNNI entities: LANEv1 LE Clients, non-SMF LANEv2 LE Clients , SMF LE Clients, and BUSs.

6.2 VCC-Mapped BUS Forwarding Rules

There are many ways that a VCC-Mapped BUS can be implemented and satisfy the rules stated in Section 5.7.2. Since a VCC-Mapped BUS forwards frames based solely on the type of VCC a frame was received on, the rules governing VCC-Mapped BUS behaviour are summarized below.

- Frames received from local LE Clients must be forwarded to all local LE Clients and to all BUSs and not to any SMS.
- Frames received from a BUS must be forwarded to all local LE Clients and not to any other BUS or SMS.
- Frames received from an SMS must be forwarded to all non-SMF LE Clients and not to any SMF LE Clients, any other BUS or any other SMS.
- Each LE Client and BUS in the ELAN must receive only one copy of each frame transmitted to the VCC-Mapped BUS.
- LANEv1 LE Clients accept only one Multicast Forward VCC (i.e. reside as a leaf on only one Multicast Forward VCC).

6.3 VCC Optimized VCC-Mapped BUS

The simplest VCC-Mapped BUS maintains 3 Multicast Forward VCCs: one to local SMF LE Clients, one to local non-SMF LE Clients (including LANEv1 LE Clients), and one to other BUSs. All normative statements in this section refer only to BUSs implementing this VCC-Mapped BUS. This strategy minimizes the number of leaves on the BUS Multicast Forward VCCs. However, the BUS is required to replicate frames over multiple VCCs. The forwarding rules of this section are depicted in Figure 6-1.

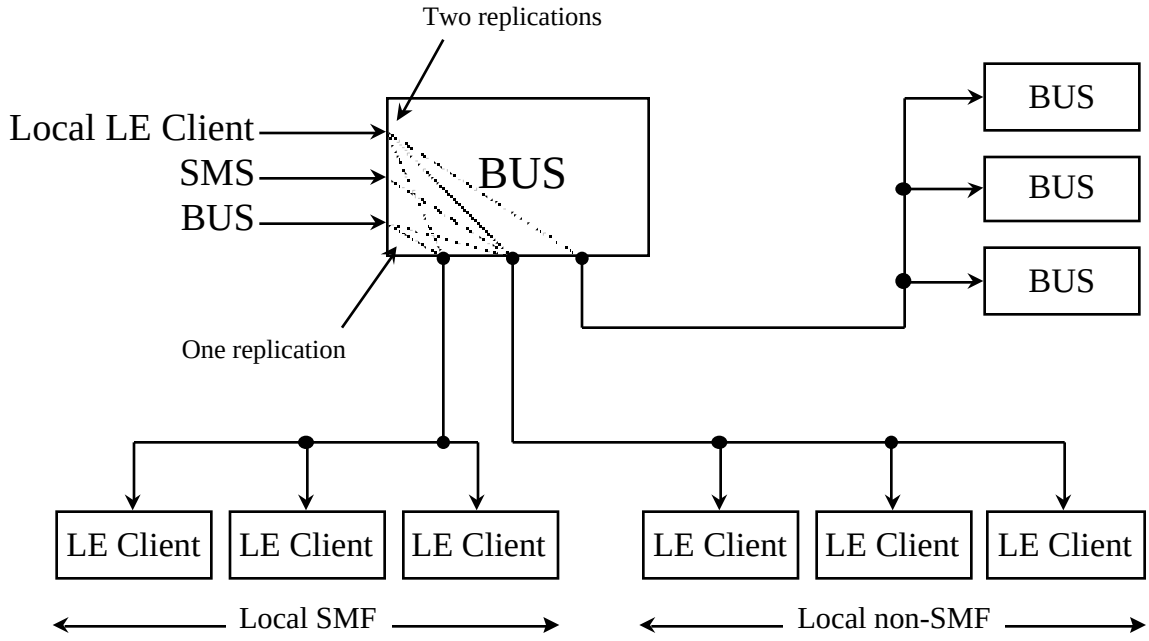

Figure 6-1 Basic VCC-Mapped BUS Example

6.3.1 Receiving Data from LE Clients

A VCC-Mapped BUS MUST map a Multicast Send VCC from a local LE Client to all three Multicast Forward VCCs. Frames received from local LE Clients MUST be forwarded over each of the three Multicast Forward VCCs.

6.3.2 Receiving Data from BUSs

A VCC-Mapped BUS MUST map a Multicast Forward VCC from another BUS to the two Multicast Forward VCCs to local LE Clients. Frames received from BUSs MUST be forwarded over each of the two Multicast Forward VCCs.

6.3.3 Receiving Data from SMSs

A VCC-Mapped BUS MUST map a Multicast Forward VCC from an SMS to the Multicast Forward VCC to local non-SMF LE Clients. Frames received from SMSs MUST be forwarded over this Multicast Forward VCC.

6.4 Replication Optimized VCC-Mapped BUS with no LANEv2 LE Clients

The BUS strategy in Section 6.3 is optimized for VCC usage. The remainder of the examples present BUS strategies that are optimized for frame replication. For this section, a simplifying assumption is made in that there are no LANEv2 LE Clients. This assumption is removed in a later section. The BUS has two Multicast Forward VCCs, one to all local (LANEv1) LE Clients, and the other to all BUSs. The BUS only has to replicate frames from local LE Clients. Normative statements in this section refer only to BUSs implementing this VCC-Mapped BUS strategy. The forwarding rules of this section are depicted in Figure 6-2.

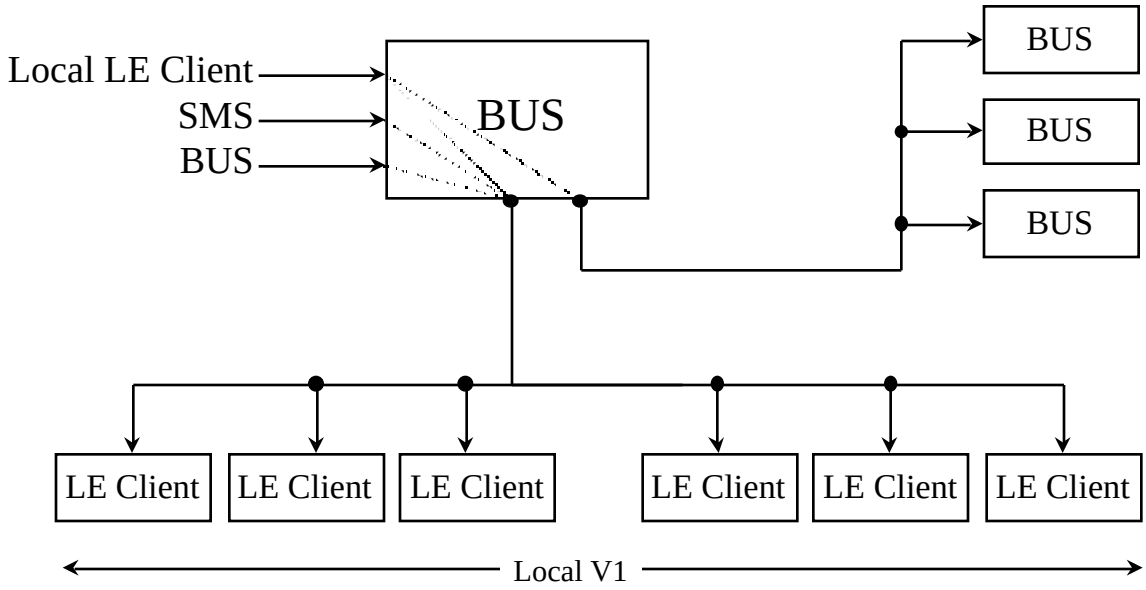

Figure 6-2 VCC-Mapped BUS Example with no LANEv2 LE Clients

6.4.1 Receiving Data from LE Clients

A BUS MUST map a Multicast Send VCC from a local LE Client to both Multicast Forward VCCs. Frames received from local LE Clients MUST be forwarded over each of these Multicast Forward VCCs.

6.4.2 Receiving Data from BUSs

A VCC-Mapped BUS MUST map a Multicast Forward VCC from another BUS to the Multicast Forward VCC to local LE Clients. Frames received from BUSs MUST be forwarded over this Multicast Forward VCC.

6.4.3 Receiving Data from SMSs

A VCC-Mapped BUS MUST map a Multicast Forward VCC from an SMS to the Multicast Forward VCC to local LE Clients. Frames received from SMSs MUST be forwarded over this Multicast Forward VCC.

6.5 Replication Optimized VCC-Mapped BUS with no LANEv1 LE Clients

The BUS strategy in Section 6.4 assumes that there are no LANEv2 LE Clients. In this section, the assumption is that there are no LANEv1 LE Clients. This assumption is removed in a later section. Again, the BUS is optimized for frame replication. The BUS has three Multicast Forward VCCs, one to all local non-SMF LE Clients, one to all local LE Clients, and the other to all local LE Clients and all BUSs. Note that each local LE Client is a leaf on multiple Multicast Forwards VCCs. Using this strategy, the BUS does not perform any frame replication, each incoming frame is forwarded on a single Multicast Forward VCC. Normative statements in this section refer only to BUSs implementing this VCC-Mapped BUS strategy. The forwarding rules of this section are depicted in Figure 6-3.

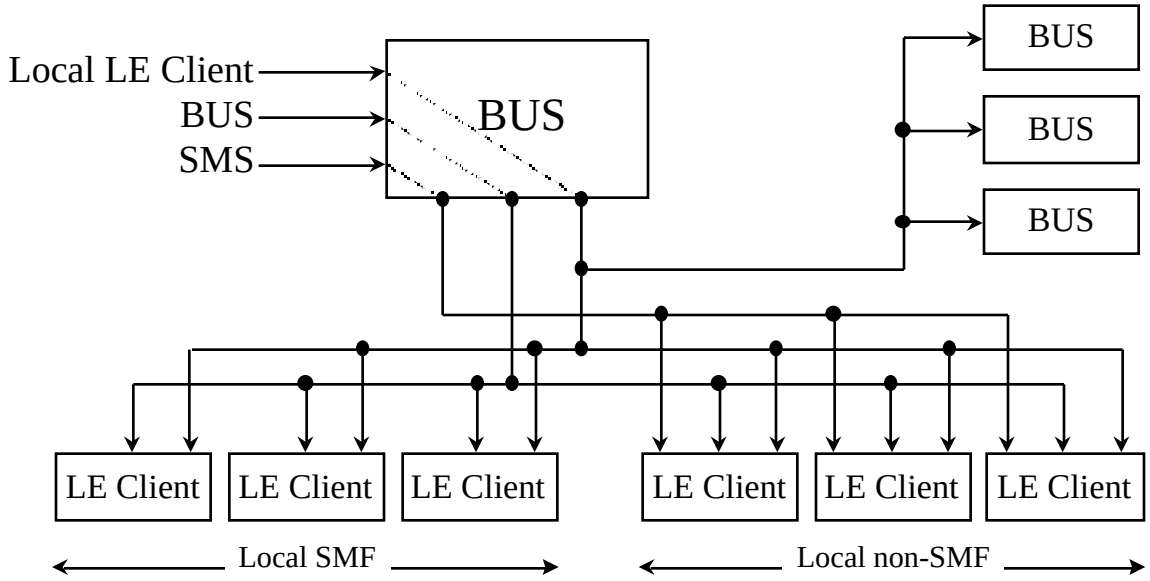

Figure 6-3 VCC-Mapped BUS Example with no LANEv1 LE Clients

6.5.1 Receiving Data from LE Clients

A VCC-Mapped BUS MUST map a Multicast Send VCC from a local LE Client to the Multicast Forward VCC to local LE Clients and BUSs. Frames received from local LE Clients MUST be forwarded over each of this Multicast Forward VCC.

6.5.2 Receiving Data from BUSs

A VCC-Mapped BUS MUST map a Multicast Forward VCC from another BUS to the Multicast Forward VCC to all local LE Clients. Frames received from BUSs MUST be forwarded over this Multicast Forward VCC.

6.5.3 Receiving Data from SMSs

A VCC-Mapped BUS MUST map a Multicast Forward VCC from an SMS to the Multicast Forward VCC to local non-SMF LE Clients. Frames received from SMSs MUST be forwarded over this Multicast Forward VCC.

6.6 Replication Optimized VCC-Mapped BUS with LANEv1 and LANEv2 LE Clients

In this section, we present a VCC-Mapped BUS that has both LANEv1 and LANEv2 LE Clients, and which optimizes for frame replication by eliminating multiple transmissions where possible. This BUS maintains 4 Multicast Forward VCCs: one to all local LANEv1 LE Clients, one to local non-SMF LE Clients, one to all local LANEv2 LE Clients, and one to all local LANEv2 LE Clients and all BUSs. This VCC-Mapped BUS implementation segregates LANEv1 LE Clients from all other LNNI BUS destinations, since LANEv1 LE Clients accept only a single Multicast Forward VCC.

The forwarding rules of this section are depicted in Figure 6-4.

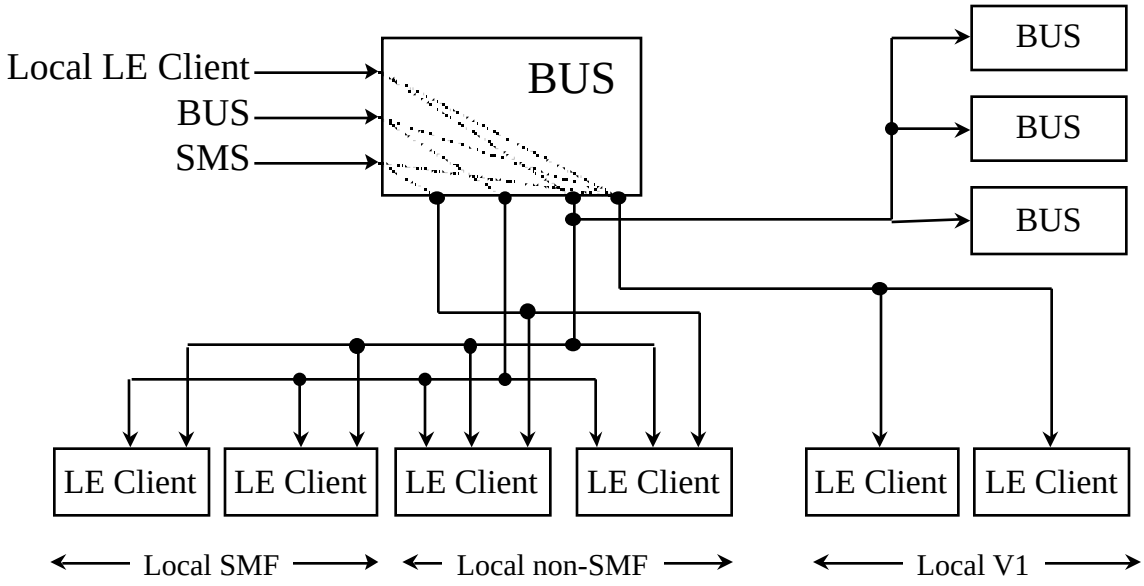

Figure 6-4 VCC-Mapped BUS Example with v1 & v2 LE Clients

6.6.1 Receiving Data from LE Clients

A VCC-Mapped BUS MUST map a Multicast Send VCC from a local LE Client to the Multicast Forward VCC to local LANEv1 LE Clients and to the Multicast Forward VCC to local LANEv2 LE Clients and BUSs. Frames received from local LE Clients MUST be forwarded over each of these Multicast Forward VCCs.

6.6.2 Receiving Data from BUSs

A VCC-Mapped BUS MUST map a Multicast Forward VCC from another BUS to the Multicast Forward VCC to local LANEv1 LE Clients and to the Multicast Forward VCC to local LANEv2 LE Clients. Frames received from BUSs MUST be forwarded over each of these Multicast Forward VCCs.

6.6.3 Receiving Data from SMSs

A VCC-Mapped BUS MUST map a Multicast Forward VCC from an SMS to the Multicast Forward VCC to local LANEv1 LE Clients and to local non-SMF LANEv2 LE Clients. Frames received from SMSs MUST be forwarded over each of these Multicast Forward VCC.

7 Appendix B – LNNI TLVs [Normative]

This appendix provides the specifications for all of the TLVs which are defined for LNNI. The Version column indicates that the TLV was defined for LANEv2 LNNI. Other LANE TLVs are specified in other ATM Forum approved specifications and therefore are not defined in this document, but may be included in LNNI defined control frames. If a LNNI TLV is received whose length is less than that as defined in Table 7.1, then the TLV MUST be ignored. The TLV MUST be processed if the length is greater than or equal to the length given in Table 7.1.

Figure 7-1 : LNNI TLVs

Item	Type	Len	Description
ServerId	00-A0-3E-14	2	Unique identifier for a server within an ELAN.
ServerGroupId	00-A0-3E-15	2	Uniquely correlates to an ELAN-ID. Required for SCSP.
SynchronizationPeerServer	00-A0-3E-16	20	Multiplexed ATM Address of LES or SMS to synchronize databases using SCSP.
LecIdRange	00-A0-3E-17	4	Format of TLV is xxxxyyyy where: xxxx = Two octet lower LE Client ID Bound yyyy = Two octet upper LE Client ID Bound The LECID range must constitute a unique non-overlapping set of LECIDs for the ELAN.
OptimizedSmsTopology	00-A0-3E-18	1	If set to zero, SMS must maintain complete registration database of unicast information and LESs must synchronize unicast registraton data with neighbor SMSs. Else, if set to one, the optimizations desribed in Section 2.4.4 are permitted.
SmsModeOfOperation	00-A0-3E-19	1	Indicates SMS operational mode. 0 = STAND_ALONE 1 = DISTRIBUTED
InitialRetryTimeout	00-A0-3E-1A	2	the time in seconds that a server may wait before it retries after a VCC setup fails
RetryMultiplier	00-A0-3E-1B	2	specifies the factor by which the **PeerConnectRetryTimeout** value must be multipled before retrying subsequent

			failed VCC setup
MaxRetryTimeout	00-A0-3E-1C	2	specifies the maximum time in seconds that a server may wait before it retries a setup to the peer server.
OperationalServer	00-A0-3E-2E	24	First twenty octets contain the non-multiplexed ATM Address of an operational server. The next two octets represent the Keep-Alive time, which is the time in seconds the LECS will assume the LES is alive in the absence of further Keep-Alive Requests. This Keep-Alive time is ignored by LECS on Keep-Alive Requests, but should be set to 0 by the LES. The last two bytes represent the the Operational-Status of the serverand should be set to 0 for normal operation and set to 1 when suffering resource starvation. This Operational-Status is ignored by LES on Keep-Alive Responses, but should be set to 0 by the LECS. The TLV is also included in LECS Synchronization Frames.
ControlTimeout	00-A0-3E-2F	2	Two-octet value representing the LE Server Control Timeout (S4).
MaximumFrameAge	00-A0-3E-30	1	Single-octet value specifying the Maximum Frame Age (S5) of the BUS.
LocalSegmentId	00-A0-3E-31	2	Two-octet value for the emulated LAN representing the IEEE 802.5 source routing segment ID (S8).
SendDisconnectTimeout	00-A0-3E-32	2	Two octet value specifying the BUS Send Disconnect Timeout (S9).
ServerRefreshLifetime	00-A0-3E-33	2	Two-octet value representing the lifetime value used by the server in its originated refresh CSAs.
ClientJoinLifetime	00-A0-3E-34	2	Two-octet value representing the lifetime value used by the server in its originated LE Client Join CSAs.
RegistrationLifetime	00-A0-3E-35	2	Two-octet value representing the lifetime value used by the

			server in its originated registration CSAs.
IdleLaneControlVccTimeout	00-A0-3E-36	2	Two-octet value representing the time a VCC must remain idle before releasing the VCC.
SyncHopCount	00-A0-3E-37	1	Set to atleast the depth of the graph of synchronized servers with this server as the root of the graph.
ServerReinitDelayMin	00-A0-3E-38	2	The minimum amount of time delay before a server tries to reinitialize.
ServerReinitDelayMax	00-A0-3E-39	2	The maximum amount of time delay before a server tries to reinitialize.
ConfigurationRequestTimeout	00-A0-3E-3A	2	The maximum time that an LES should wait for a configuration response before retrying a configuration response from the LECS.
ControlRequestRetries	00-A0-3E-3B	2	The maximum number of times that the LES will send the initial Keep-Alive request.
MulticastMACAddress	00-A0-3E-3C	6	Multicast MAC Address assigned to SMS.

where 00-A0-3E is the ATM Forum OUI.

8 Appendix C - State Machines

This appendix presents a state machine model of the LNNI.

The purpose of this appendix is to help ensure the correctness and completeness of the specification and to increase its clarity. This is important to assure interoperability among LNNI components and LE Clients when implemented by different vendors. This model should be viewed as a possible interpretation of the LNNI specification. Note that some of the transitions are implementation dependent, either optional or determined by local policy.

Each state machine is modelled with a set of states and a set of transitions. Each transition interconnects a pair of states. The behaviour of a state machine is based on events and actions: if an event happens when the machine is in a particular state, then the action is performed and a state transition takes place (potentially back to the same state). An event triggers at most one transition. An event that is not specified in the state machine is ignored.

8.1 LES Configuration State Machine

Each LES and SMS in the ELAN runs a Configuration State Machine. This state machine covers the functions of obtaining a list of LECSs, finding and connecting to an operational LECS, obtaining configuration information from the LECS, sending Keep-Alive Requests to the LECS, and obtaining new configuration information from the LECS.

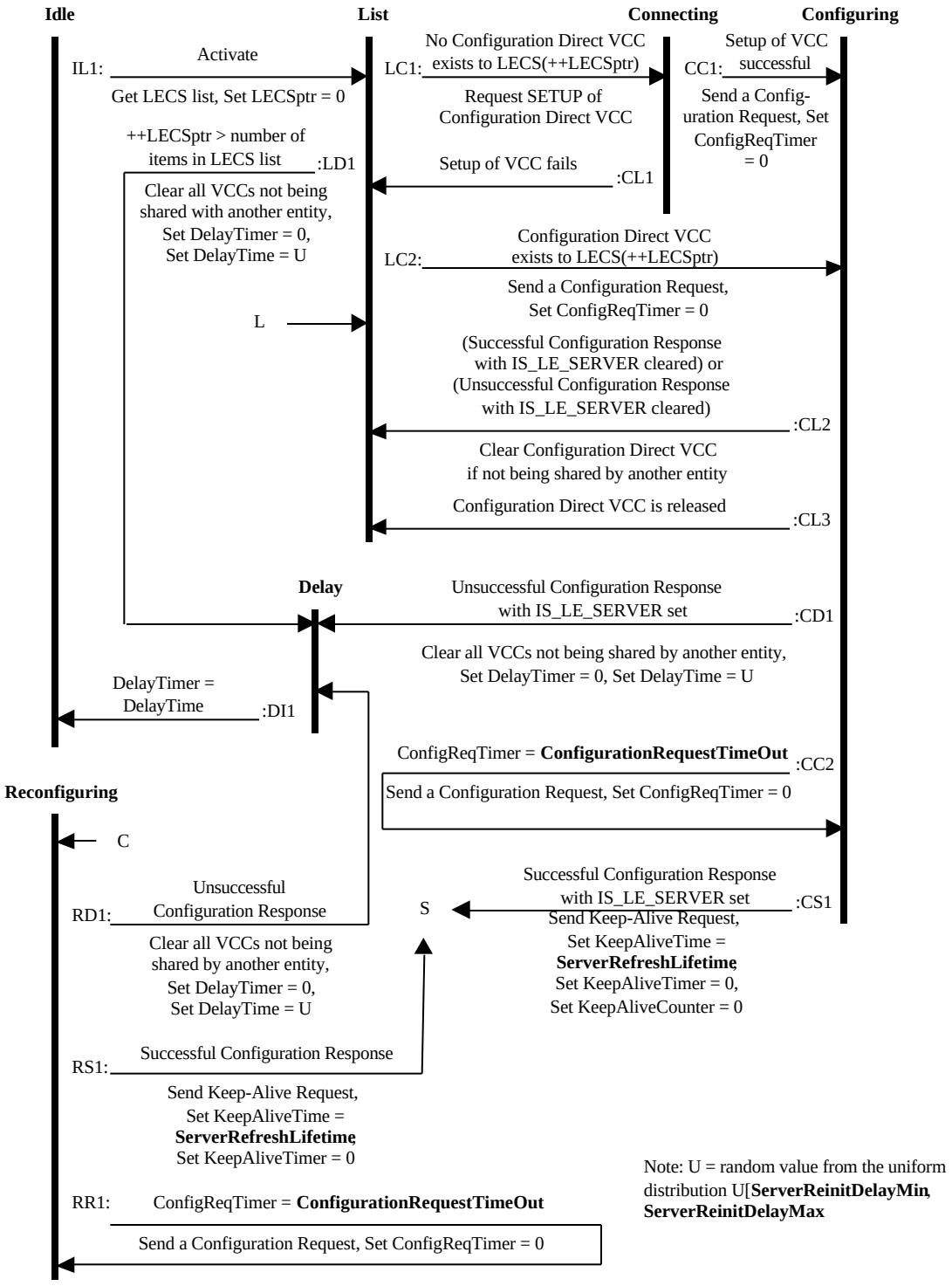

Figure 8-1 LES and SMS Configuration State Machine – Part 1

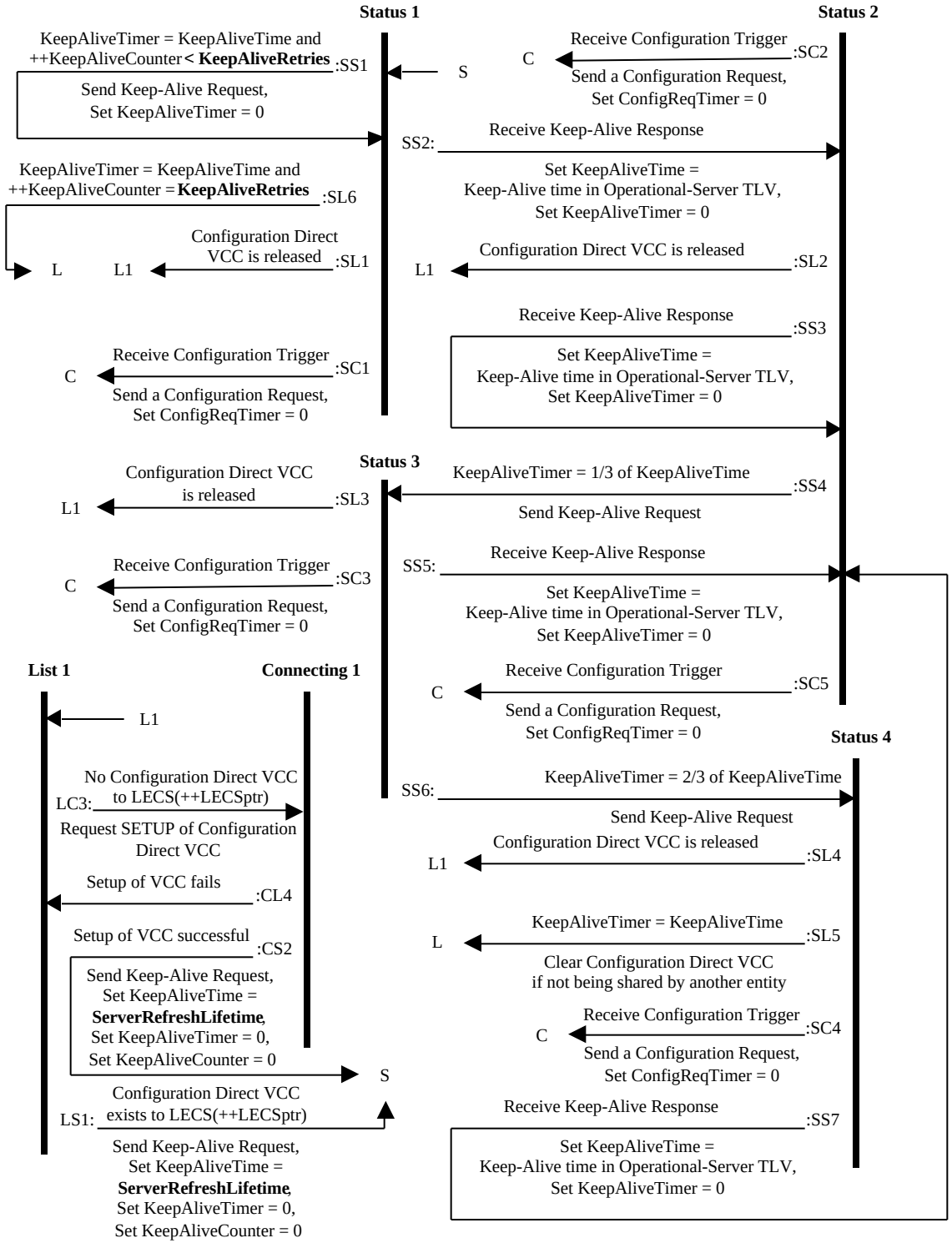

Figure 8-2 LES and SMS Configuration State Machine - Part 2

CC1: If the setup of the Configuration Direct VCC is successful, the server issues a Configuration Request to the LECS and moves to the Configuring state.

CC2: If a Configuration Response is not received before **ConfigurationRequestTimeout** seconds have elapsed, the server resends the Configuration Request and remains in the Configuring state.

CD1: If a server receives an unsuccessful Configuration Response with the IS_LE_SERVER flag set for a LES or with the IS_MCAST_SERVER set for a SMS, the server clears all VCCs that are not being shared by another entity and returns to the idle state after a random delay selected from the uniform distribution U[**ServerReinitDelayMin**, **ServerReinitDelayMax**].

CL1: If the setup of the Configuration Direct VCC fails, the server moves to the List state to try to configure with the next LECS in the list if one exists.

CL2: If a successful or unsuccessful Configuration Response is received by a LES with the IS_LE_SERVER flag cleared or by a SMS with the IS_MCAST_SERVER flag cleared, the server clears the VCC unless it is being shared by another entity and moves to the List state to try to configure with the next LECS in the list if one exists.

CL3: If the Configuration Direct VCC is released, the server moves to the List state to try to configure with the next LECS in the list if one exists.

CL4: If the setup of the Configuration Direct VCC fails, the server moves to the List 1 state to try to configure with the next LECS in the list if one exists.

CS1: If a successful Configuration Response is received by a LES with the IS_LE_SERVER flag set or by a SMS with the IS_MCAST_SERVER flag set, the server sends a Keep-Alive Request to the LECS, sets a timer to expire after **ServerRefreshLifetime** if no response is received, and initializes the KeepAliveCounter.

CS2: If the setup of the Configuration Direct VCC is successful and the KeepAlive Time has expired, then the server must enter the configuration phase.

CS3: If the setup of the Configuration Direct VCC is successful and the KeepAlive Time has not expired, the server sends a Keep-Alive Request to the LECS, sets a timer to expire after **ServerRefreshLifetime** if no response is received, and initializes the KeepAliveCounter.

DI1: After the random delay, the server returns to the idle state.

IL1: Whenever a server is operational it must be activated which causes it to obtain the list of LECSs.

LC1: If no VCC exists that can be used as the Configuration Direct VCC to the LECS, the server must attempt to set one up and await the result of the connection request.

LC2: If a VCC exists that can be used as a Configuration Direct VCC to LECS, the server issues a Configuration Request to the LECS and moves to the Connected state.

LC3: If no VCC exists that can be used as the Configuration Direct VCC to the LECS, the server must attempt to set one up and await the result of the connection request.

LD1: If the incremented LECSptr is beyond the end of the LECS list, then the server clears all VCCs that are not being shared by another entity and returns to the idle state after a random delay selected from the uniform distribution U[**ServerReinitDelayMin**, **ServerReinitDelayMax**].

LS1: If a VCC exists that can be used as a Configuration Direct VCC to LECS, the server sends a Keep-Alive Request to the LECS, sets a timer to expire after **ServerRefreshLifetime** if no response is received, and initializes the KeepAliveCounter

RD1: If an unsuccessful Configuration Response is received, the server clears all VCCs that are not being shared by another entity and returns to the idle state after a random delay selected from the uniform distribution U[**ServerReinitDelayMin**, **ServerReinitDelayMax**]

RR1: If a Configuration Response is not received before **ConfigurationRequestTimeout** seconds have elapsed, the server resends the Configuration Request and remains in the Configuring state.

RS1: If a successful Configuration Response is received, the server sends a Keep-Alive Request to the LECS and sets a timer to expire after **ServerRefreshLifetime** if no response is received.

SC1: If a Configuration Trigger is received, the server sends a new Configuration Request and waits for the response.

SC2: If a Configuration Trigger is received, the server sends a new Configuration Request and waits for the response.

SC3: If a Configuration Trigger is received, the server sends a new Configuration Request and waits for the response.

SC4: If a Configuration Trigger is received, the server sends a new Configuration Request and waits for the response.

SC5: If a Configuration Trigger is received, the server sends a new Configuration Request and waits for the response.

SL1: If the Configuration Direct VCC is released, the server moves to the List state to try to configure with the next LECS in the list if one exists.

SL2: If the Configuration Direct VCC is released, the server moves to the List state to try to configure with the next LECS in the list if one exists.

SL3: If the Configuration Direct VCC is released, the server moves to the List state to try to configure with the next LECS in the list if one exists.

SL4: If the Configuration Direct VCC is released, the server moves to the List state to try to configure with the next LECS in the list if one exists.

SL5: If Keep-Alive time seconds, as specified in the last received Operational-Server TLV, have elapsed, the server clears the Configuration Direct VCC if it is not being shared by another entity. The server moves to the List state to try to configure with the next LECS in the list if one exists.

SL6: If a Keep-Alive Response is not received before **ServerRefreshLifetime** seconds elapse and **KeepAliveRetries** Keep-Alive requests have been sent, the LES clears the Configuration Direct VCC if it is not being shared by another entity and the server moves to the List state to try to configure with the next LECS in the list if one exists

SS1: If a Keep-Alive Response is not received before **ServerRefreshLifetime** seconds elapse and **KeepAliveRetries** or less Keep-Alive requests have been sent, the server must retry the Keep-Alive Request.

SS2: If a Keep-Alive Response is received, the server records the Keep-Alive time contained in the Operational-Server TLV and uses this time to control the next Keep-Alive Request.

SS3: If a Keep-Alive Response is received, the server records the Keep-Alive time contained in the Operational-Server TLV and uses this time to control the next Keep-Alive Request.

SS4: If 1/3 of Keep-Alive time seconds, as specified in the last received Operational-Server TLV, have elapsed, resend the Keep-Alive Request.

SS5: If a Keep-Alive Response is received, the server records the Keep-Alive time contained in the Operational-Server TLV and uses this time to control the next Keep-Alive Request.

SS6: If 2/3 of Keep-Alive time seconds, as specified in the last received Operational-Server TLV, have elapsed, resend the Keep-Alive Request.

SS7: If a Keep-Alive Response is received, the server records the Keep-Alive time contained in the Operational-Server TLV and uses this time to control the next Keep-Alive Request

8.2 SMS Configuration State Machine

Each SMS in the ELAN runs an SMS Configuration State Machine. This state machine covers the functions of obtaining a list of LECSs, finding and connecting to an operational LECS, obtaining configuration information from the LECS, and obtaining new configuration information from the LECS.

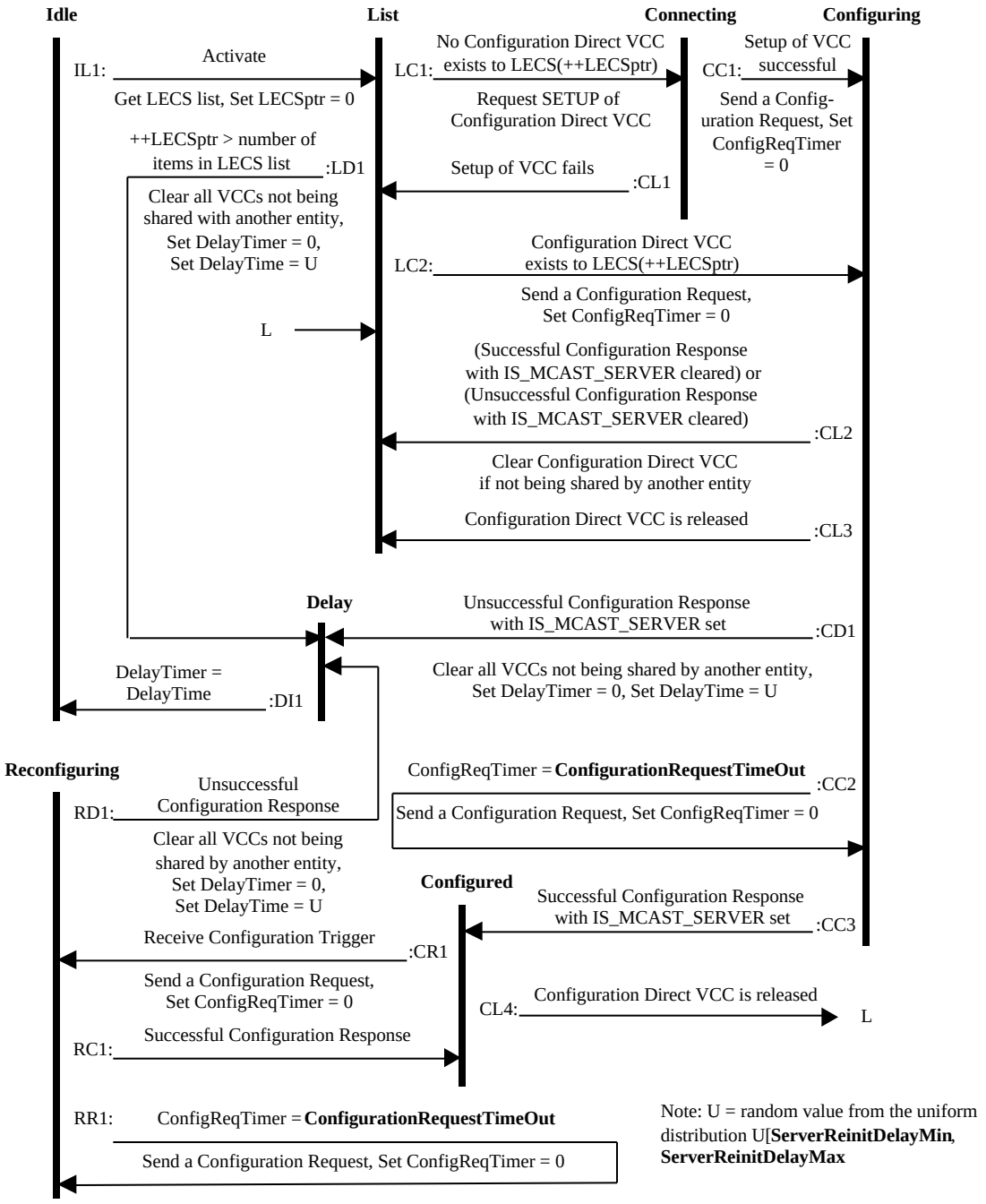

Figure 8-3 SMS Configuration State Machine

CC1: If the setup of the Configuration Direct VCC is successful, the SMS issues a Configuration Request to the LECS and moves to the Configuring state.

CC2: If a Configuration Response is not received before **ConfigurationRequestTimeout** seconds have elapsed, the SMS resends the Configuration Request and remains in the Configuring state.

CC3: If a successful Configuration Response is received with the IS_MCAST_SERVER flag set, the SMS moves to the configured state.

CD1: If an unsuccessful Configuration Response is received with the IS_MCAST_SERVER flag set, the SMS clears all VCCs that are not being shared by another entity and returns to the idle state after a random delay selected from the uniform distribution U[**ServerReinitDelayMin**, **ServerReinitDelayMax**].

CL1: If the setup of the Configuration Direct VCC fails, the SMS moves to the List state to try to configure with the next LECS in the list if one exists.

CL2: If a successful or unsuccessful Configuration Response is received with the IS_MCAST_SERVER flag cleared, the SMS clears the VCC unless it is being shared by another entity and moves to the List state to try to configure with the next LECS in the list if one exists.

CL3: If the Configuration Direct VCC is released, the SMS moves to the List state to try to configure with the next LECS in the list if one exists.

CL4: If the Configuration Direct VCC is released, the SMS moves to the List state to try to configure with the next LECS in the list if one exists.

CR1: If a Configuration Trigger is received, the SMS sends a new Configuration Request and waits for the response.

DI1: After the random delay, the SMS returns to the idle state.

IL1: Whenever an SMS is operational it must be activated which causes it to obtain the list of LECSs.

LC1: If no VCC exists that can be used as the Configuration Direct VCC to the LECS, the SMS must attempt to set one up and await the result of the connection request.

LC2: If a VCC exists that can be used as a Configuration Direct VCC to LECS, the SMS issues a Configuration Request to the LECS and moves to the Connected state.

LD1: If the incremented LECSptr is beyond the end of the LECS list, then the SMS clears all VCCs that are not being shared by another entity and returns to the idle state after a random delay selected from the uniform distribution U[**ServerReinitDelayMin**, **ServerReinitDelayMax**].

RC1: If a successful Configuration Response is received, the SMS moves to the configured state.

RD1: If an unsuccessful Configuration Response is received, the SMS clears all VCCs that are not being shared by another entity and returns to the idle state after a random delay selected from the uniform distribution U[**ServerReinitDelayMin**, **ServerReinitDelayMax**]

RR1: If **ConfigurationRequestTimeOut** seconds elapse without receiving a Configuration Response, retry the Configuration Request.

8.3 LES-Peer-LES Connection State Machine

An LES (called the local LES) runs one LES-Peer-LES Connection State Machine for each other LES in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the LESs will be in the same ELAN with the local LES.

For an LES that is in the **PeerServerList**, the state machine covers the functions of establishing both synchronization plane and control plane connectivity with the local LES. For an LES that is not in the **PeerServerList** but is discovered by the local LES through SCSP, the state machine covers the function of establishing control plane connectivity with the local LES.

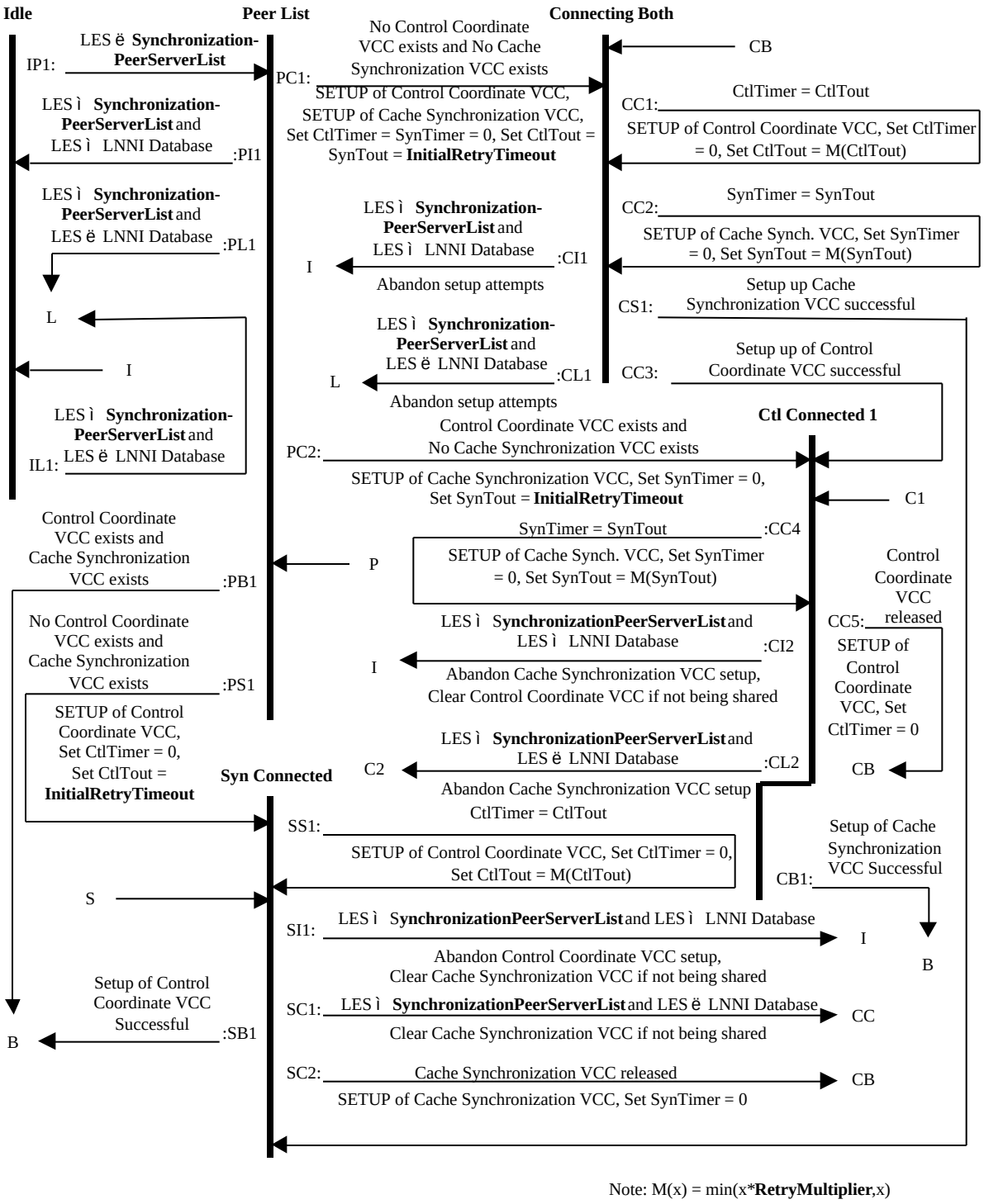

Figure 8-4 LES-Peer-LES Connection State Machine – Part 1

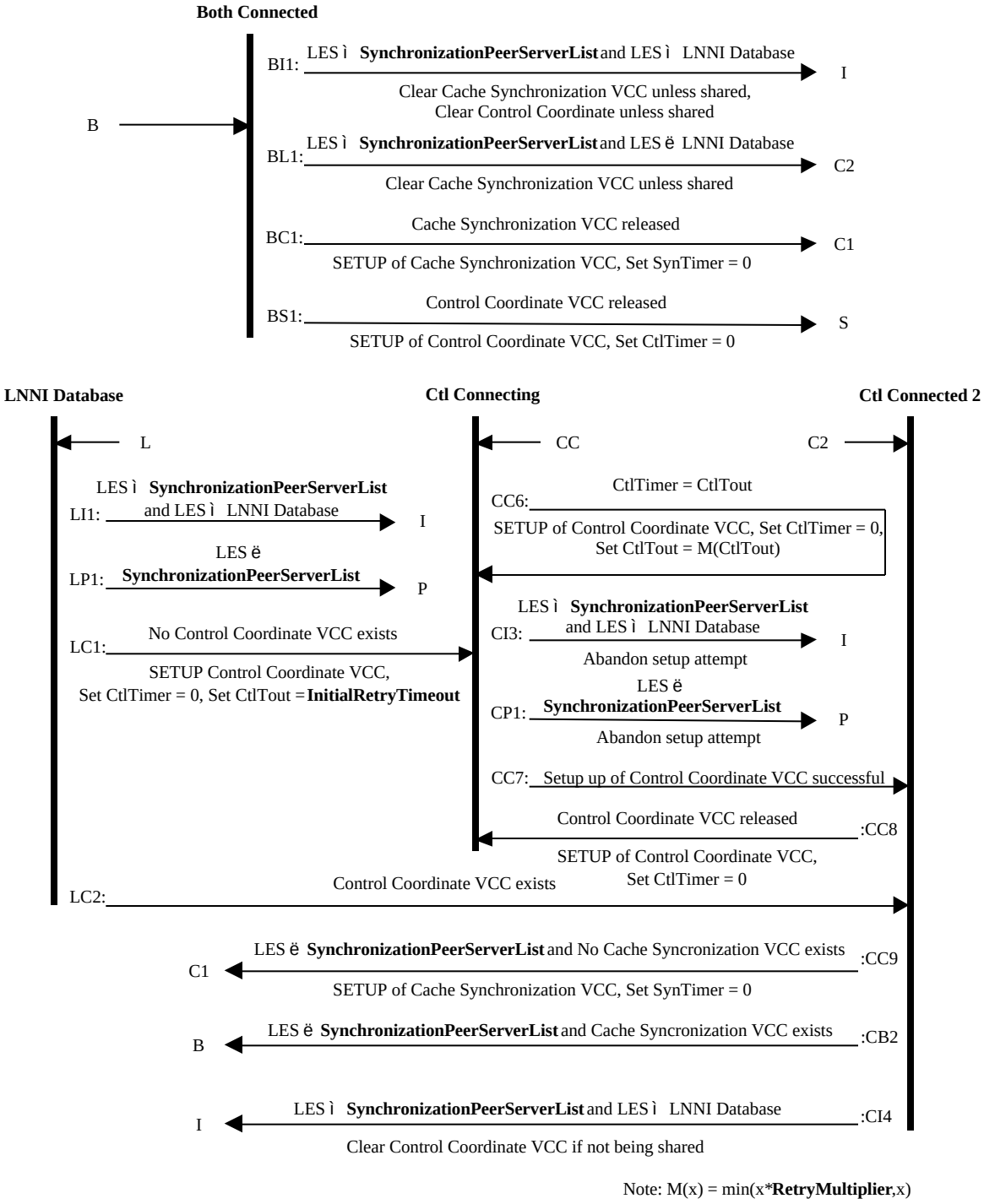

Figure 8-5 LES-PEER-LES Connection State Machine – Part 2

BC1: If the Cache Synchronization VCC is released, the local LES attempts to re-establish a Cache Synchronization VCC.

BI1: If the LES is removed from **SynchronizationPeerServerList** and is not in the LNNI Database (i.e., the local cache does not contain a CSA for the LES), then the local LES clears the Cache Synchronization VCC if it is not being shared by another entity, the local LES clears the Control Coordinate VCC if it is not being shared by another entity, and the LES returns to the Idle state.

BL1: If the LES is removed from **SynchronizationPeerServerList** and is in the LNNI Database (i.e., the local cache contains a CSA for the LES), then the local LES clears the Cache Synchronization VCC if it is not being shared by another entity and the LES goes to the Ctl Connected 2 state.

BS1: If the Control Coordinate VCC is released, the local LES attempts to re-establish a Control Coordinate VCC.

CB1: If the Cache Synchronization VCC setup successfully completes, the LES moves to the Both Connected state.

CB2: If the LES enters **SynchronizationPeerServerList** and a Cache Synchronization VCC exists to the LES, then the LES moves to the Both Connected State.

CC1: If the Control Coordinate VCC hasn't been setup before the retry timer expires, the local LES retries establishing the connection.

CC2: If the Cache Synchronization VCC hasn't been setup before the retry timer expires, the local LES retries establishing the connection.

CC3: If the Control Coordinate VCC setup successfully completes, the LES moves to the Ctl Connected 1 state.

CC4: If the Cache Synchronization VCC hasn't been setup before the retry timer expires, the local LES retries establishing the connection.

CC5: If the Control Coordinate VCC is released, the local LES attempts to re-establish a Control Coordinate VCC.

CC6: If the Control Coordinate VCC hasn't been setup before the retry timer expires, the local LES retries establishing the connection.

CC7: If the Control Coordinate VCC setup successfully completes, the LES moves to the Ctl Connected 2 state.

CC8: If the Control Coordinate VCC is released, the local LES attempts to re-establish a Control Coordinate VCC.

CC9: If the LES enters **SynchronizationPeerServerList** and no Cache Synchronization VCC exists to the LES, then the local LES attempts to setup a Cache Synchronization VCC and the LES moves to the Ctl Connected 1 State.

CI1: If the LES is removed from **SynchronizationPeerServerList** and is not in the LNNI Database (i.e., the local cache does not contain a CSA for the LES), then the local LES abandons the setup of both the Control Coordinate VCC and the Cache Synchronization VCC and the LES returns to the Idle state.

CI2: If the LES is removed from **SynchronizationPeerServerList** and is not in the LNNI Database (i.e., the local cache does not contain a CSA for the LES), then the local LES abandons the setup of the Cache Synchronization VCC, the local LES clears the Control Coordinate VCC if it is not being shared by another entity, and the LES returns to the Idle state.

CI3: If the LES is removed from the LNNI Database (i.e., the local cache does not contain a CSA for the LES), and is not in **SynchronizationPeerServerList** then the local LES abandons the Control Coordinate VCC setup attempt and the LES returns to the Idle state.

CI4: If the LES is removed from the LNNI Database (i.e., the local cache does not contain a CSA for the LES), and is not in **SynchronizationPeerServerList** then the local LES clears the Control Coordinate VCC if it is not being shared and the LES returns to the Idle state.

CL1: If the LES is removed from **SynchronizationPeerServerList** and is in the LNNI Database (i.e., the local cache contains a CSA for the LES), then the local LES abandons the setup of both the Control Coordinate VCC and the Cache Synchronization VCC and the LES goes to the LNNI Database state.

CL2: If the LES is removed from **SynchronizationPeerServerList** and is in the LNNI Database (i.e., the local cache contains a CSA for the LES), then the local LES abandons the setup of the Cache Synchronization VCC and the LES goes to the Ctl Connected 2 state.

CP1: If the LES enters **SynchronizationPeerServerList**, then the local LES abandons the setup of the Control Coordinate VCC and the LES moves to the Peer List state.

CS1: If the Cache Synchronization VCC setup successfully completes, the LES moves to the Syn Connected state.

IL1: If the LES enters the LNNI Database and is not in **SynchronizationPeerServerList**, then the LES moves to the LNNI Database state.

IP1: If the LES enters **SynchronizationPeerServerList**, then the LES moves to the Peer List state.

LC1: If there is no Control Coordinate VCC to the LES, the local LES attempts to setup a Control Coordination VCC to the LES.

LC2: If a Control Coordinate VCC exists to the LES, the LES moves to the Ctl Connected 2 state.

LI1: If the LES is removed from the LNNI Database (i.e., the local cache does not contain a CSA for the LES), and is not in **SynchronizationPeerServerList** then the LES returns to the Idle state.

LP1: If the LES enters **SynchronizationPeerServerList**, then the LES moves to the Peer List state.

PB1: If there is a Control Coordinate VCC to the LES and there is a Cache Synchronization VCC to the LES, the LES moves to the Both Connected state.

PC1: If there is no Control Coordinate VCC to the LES and there is no Cache Synchronization VCC to the LES, the local LES attempts to setup Control Coordinate and Cache Synchronization VCCs to the LES.

PC2: If there is a Control Coordinate VCC to the LES but there is no Cache Synchronization VCC to the LES, the local LES attempts to setup a Cache Synchronization VCC to the LES.

PI1: If the LES is removed from **SynchronizationPeerServerList** and is not in the LNNI Database (i.e., the local cache does not contain a CSA for the LES), then the LES returns to the Idle state.

PL1: If the LES is removed from **SynchronizationPeerServerList** and is in the LNNI Database, then the LES moves to the LNNI Database state.

PS1: If there is no Control Coordinate VCC to the LES and there is Cache Synchronization VCC to the LES, the local LES attempts to setup a Control Coordinate VCC to the LES.

SB1: If the Control Coordinate VCC setup successfully completes, the LES moves to the Both Connected state.

SC1: If the LES is removed from **SynchronizationPeerServerList** and is in the LNNI Database (i.e., the local cache contains a CSA for the LES), then the local LES clears the Cache Synchronization VCC if it is not being shared by another entity and the LES goes to the Ctl Connecting state.

SC2: If the Cache Synchronization VCC is released, the local LES attempts to re-establish a Cache Synchronization VCC.

SI1: If the LES is removed from **SynchronizationPeerServerList** and is not in the LNNI Database (i.e., the local cache does not contain a CSA for the LES), then the LES returns to the Idle state.

SS1: If the Control Coordinate VCC hasn't been setup before the retry timer expires, the local LES retries establishing the connection.

8.4 LES-Peer-SMS Connection State Machine

An LES (called the local LES) runs one LES-Peer-SMS Connection State Machine for each other SMS in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the SMSs will be in the same ELAN with the local LES.

For an SMS that is in the **SynchronizationPeerServerList**, the state machine covers the functions of establishing synchronization plane connectivity with the local LES.

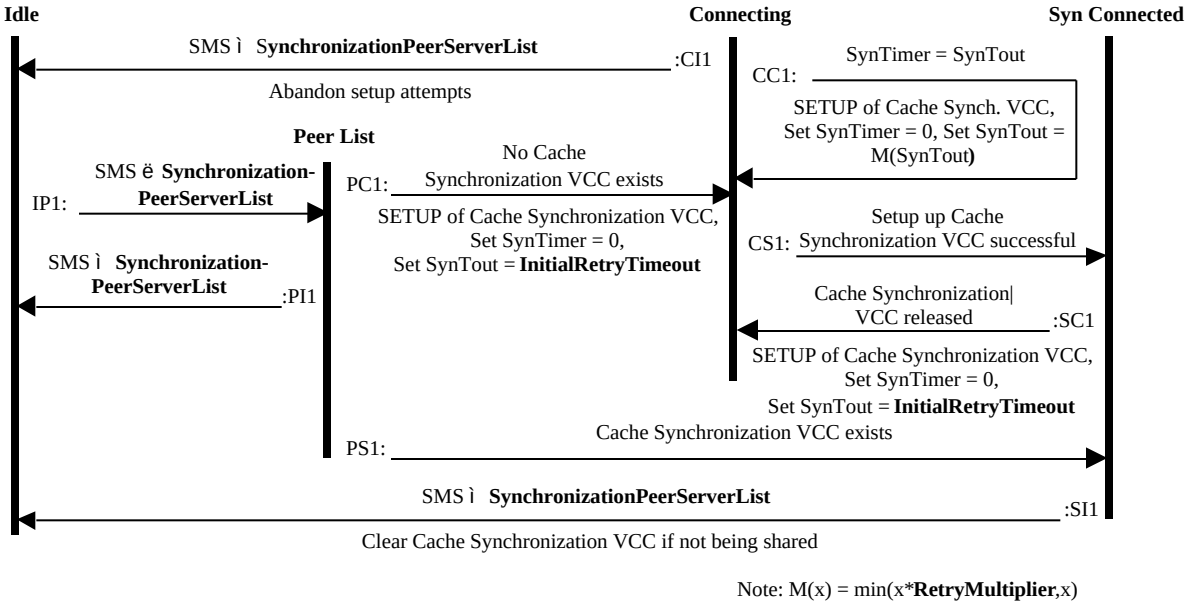

Figure 8-6 LES-PEER-SMS Connection State Machine

CC1: If the Cache Synchronization VCC hasn't been setup before the retry timer expires, the local LES retries establishing the connection.

CI1: If the SMS is removed from **SynchronizationPeerServerList,** then the local LES abandons the setup of the Cache Synchronization VCC and the SMS returns to the Idle state.

CS1: If the Cache Synchronization VCC setup successfully completes, the SMS moves to the Syn Connected state.

IP1: If the SMS enters **SynchronizationPeerServerList**, then the SMS moves to the Peer List state.

PC1: If there is no Cache Synchronization VCC to the SMS, the local LES attempts to setup a Cache Synchronization VCC to the SMS.

PI1: If the SMS is removed from **SynchronizationPeerServerList,** then the SMS returns to the Idle state.

PS1: If there is a Cache Synchronization VCC to the SMS, the SMS moves to the Syn Connected state.

SC1: If the Cache Synchronization VCC is released, the local LES attempts to re-establish a Cache Synchronization VCC.

SI1: If the SMS is removed from **SynchronizationPeerServerList**, then the SMS returns to the Idle state.

8.5 SMS-Peer-LES Synchronization Connection State Machine

An SMS (called the local SMS) runs one SMS-Peer-LES Connection State Machine for each other LES in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the LESs will be in the same ELAN with the local SMS.

For an LES that is in the **SynchronizationPeerServerList**, the state machine covers the functions of establishing synchronization plane connectivity with the local SMS.

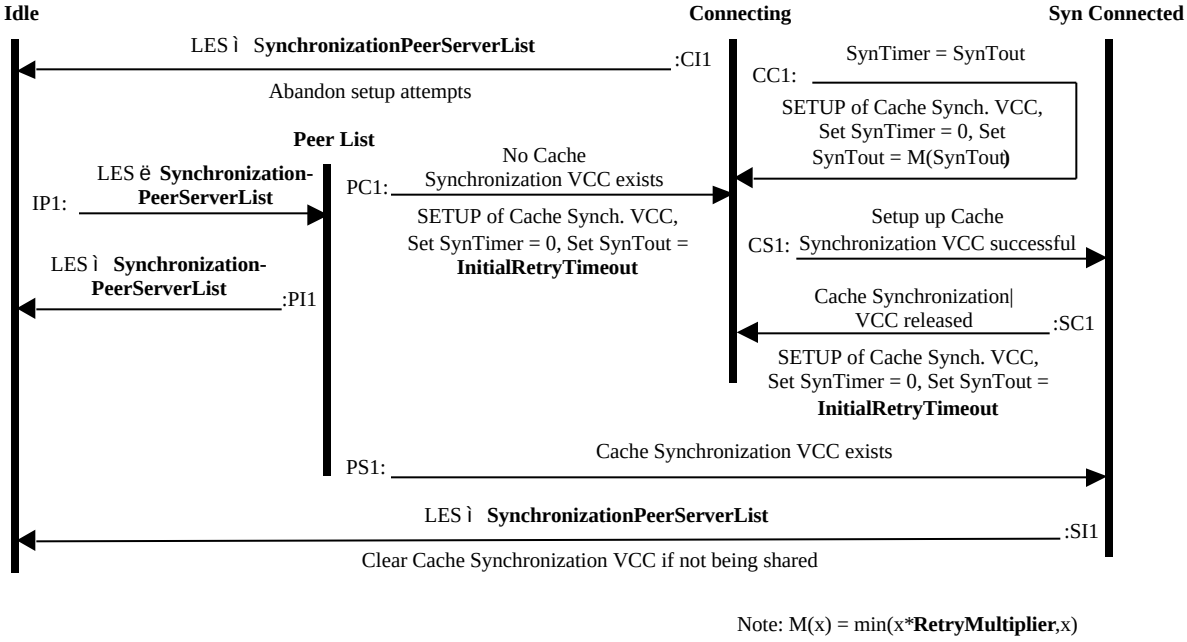

Figure 8-7 SMS-Peer-LES Synchronization Connection State Machine

CC1: If the Cache Synchronization VCC hasn't been setup before the retry timer expires, the local SMS retries establishing the connection.

CI1: If the LES is removed from **SynchronizationPeerServerList,** then the local SMS abandons the setup of the Cache Synchronization VCC and the LES returns to the Idle state.

CS1: If the Cache Synchronization VCC setup successfully completes, the LES moves to the Syn Connected state.

IP1: If the LES enters **SynchronizationPeerServerList**, then the LES moves to the Peer List state.

PC1: If there is no Cache Synchronization VCC to the LES, the local SMS attempts to setup a Cache Synchronization VCC to the LES.

PI1: If the LES is removed from **SynchronizationPeerServerList,** then the LES returns to the Idle state.

PS1: If there is a Cache Synchronization VCC to the LES, the LES moves to the Syn Connected state.

SC1: If the Cache Synchronization VCC is released, the local SMS attempts to re-establish a Cache Synchronization VCC.

SI1: If the LES is removed from **SynchronizationPeerServerList**, then the LES returns to the Idle state.

8.6 SMS-Peer-SMS Synchronization Connection State Machine

An SMS (called the local SMS) runs one SMS-Peer-SMS Connection State Machine for each other SMS in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the SMSs will be in the same ELAN with the local SMS.

For an SMS that is in the **SynchronizationPeerServerList**, the state machine covers the functions of establishing synchronization plane connectivity with the local SMS.

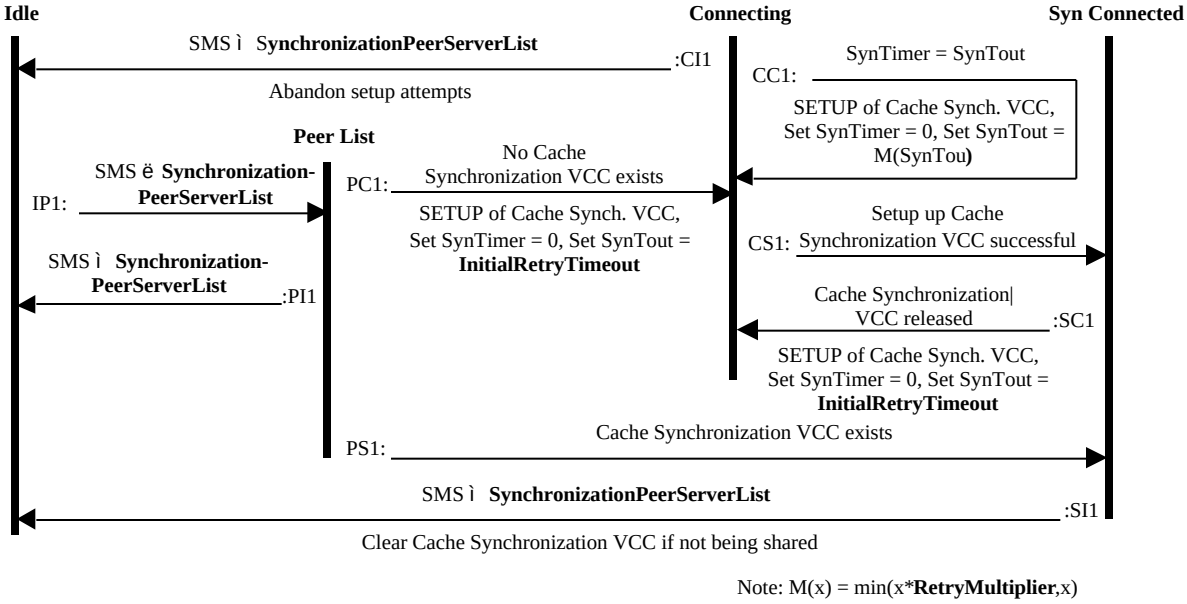

Figure 8-8 SMS-Peer-SMS Synchronization Connection State Machine

CC1: If the Cache Synchronization VCC hasn't been setup before the retry timer expires, the local SMS retries establishing the connection.

CI1: If the SMS is removed from **SynchronizationPeerServerList,** then the local SMS abandons the setup of the Cache Synchronization VCC and the SMS returns to the Idle state.

CS1: If the Cache Synchronization VCC setup successfully completes, the SMS moves to the Syn Connected state.

IP1: If the SMS enters **SynchronizationPeerServerList**, then the SMS moves to the Peer List state.

PC1: If there is no Cache Synchronization VCC to the SMS, the local SMS attempts to setup a Cache Synchronization VCC to the SMS.

PI1: If the SMS is removed from **SynchronizationPeerServerList,** then the SMS returns to the Idle state.

PS1: If there is a Cache Synchronization VCC to the SMS, the SMS moves to the Syn Connected state.

SC1: If the Cache Synchronization VCC is released, the local SMS attempts to re-establish a Cache Synchronization VCC.

SI1: If the SMS is removed from **SynchronizationPeerServerList**, then the SMS returns to the Idle state.

8.7 SMS-BUS Data Connection State Machine

An SMS runs one SMS-BUS Data Connection State Machine for each BUS in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the BUSs will be in the same ELAN with the SMS.

The state machine covers the function of the SMS connecting to the BUS to allow distribution of multicast data frames.

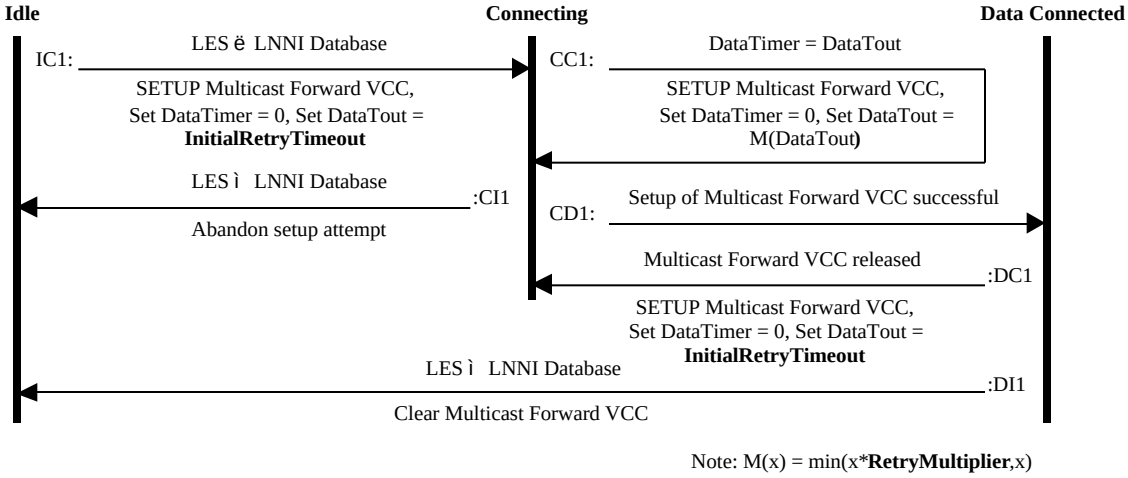

Figure 8-9 SMS-BUS Data Connection State Machine

CC1: If the Multicast Forward VCC hasn't been setup before the retry timer expires, the SMS retries establishing the connection.

CD1: If the Multicast Forward VCC setup successfully completes, the BUS moves to the Data Connected state.

CI1: If the LES that is associated with the BUS is removed from the LNNI Database, then the SMS abandons the setup of the Multicast Forward VCC and the BUS returns to the Idle state.

DC1: If the Multicast Forward VCC is released, the SMS attempts to re-establish a Multicast Forward VCC.

DI1: If the LES that is associated with the BUS leaves the LNNI Database, then the SMS clears the Multicast Forward VCC and the BUS returns to the Idle state.

IC1: If the LES that is associated with the BUS enters the LNNI Database, then the SMS attempts to setup a Multicast Forward VCC to the BUS.

8.8 SMS-SMS Data Connection State Machine

An SMS (called the local SMS) runs one SMS-SMS Data Connection State Machine for each <SMS, Multicast LAN Destination> pair in the universe. The vast majority of these state machines will remain in the Idle state since

only a tiny percentage of the <SMS, Multicast LAN Destination> pairs will be in the same ELAN with the local SMS.

The state machine covers the function of the local SMS connecting to the other SMSs to allow distribution of multicast data frames.

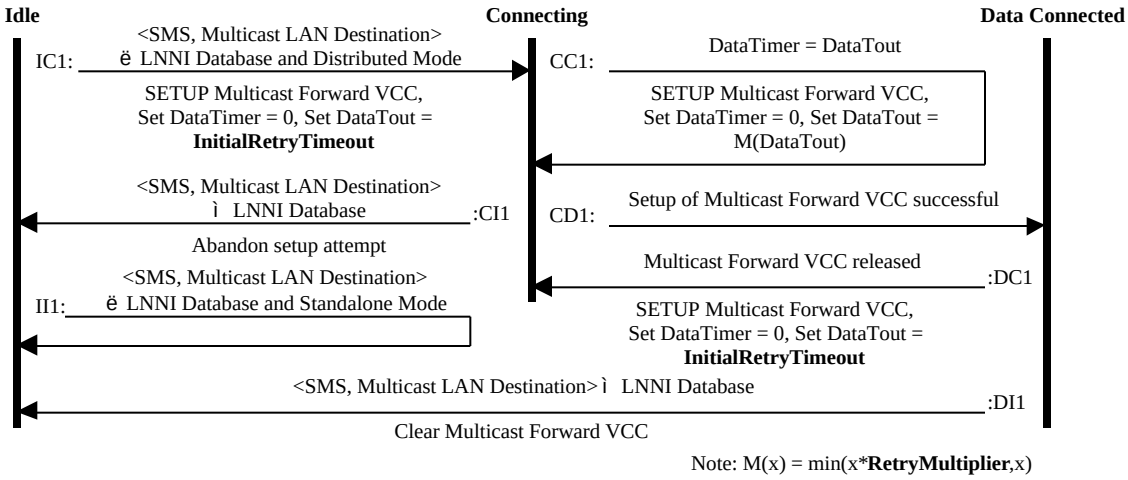

Figure 8-10 SMS-SMS Data Connection State Machine

CC1: If the Multicast Forward VCC hasn't been setup before the retry timer expires, the local SMS retries establishing the connection.

CD1: If the Multicast Forward VCC setup successfully completes, the <SMS, Multicast LAN Destination> pair moves to the Data Connected state.

CI1: If the <SMS, Multicast LAN Destination> pair is removed from the LNNI Database, then the local SMS abandons the setup of the Multicast Forward VCC and the <SMS, Multicast LAN Destination> pair returns to the Idle state.

DC1: If the Multicast Forward VCC is released, the local SMS attempts to re-establish a Multicast Forward VCC.

DI1: If the <SMS, Multicast LAN Destination> pair leaves the LNNI Database, then the SMS clears the Multicast Forward VCC and the <SMS, Multicast LAN Destination> pair returns to the Idle state.

IC1: If the <SMS, Multicast LAN Destination> enters the LNNI Database and the local SMS is in Distributed Mode, then the local SMS attempts to setup a Multicast Forward VCC to the SMS.

II1: If the <SMS, Multicast LAN Destination> pair enters the LNNI Database and the local SMS is in Standalone Mode, then the SMS remains in the Idle state.

8.9 SMS-LE Client Data Connection State Machine

An SMS (called the local SMS) runs one SMS-LE Cient Data Connection State Machine for each <LE Client, Multicast LAN Destination> pair in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the <LE Client, Multicast LAN Destination> pairs will be in the same ELAN with the local SMS.

The state machine covers the function of the SMS connecting to LE Clients to allow distribution of multicast data frames.

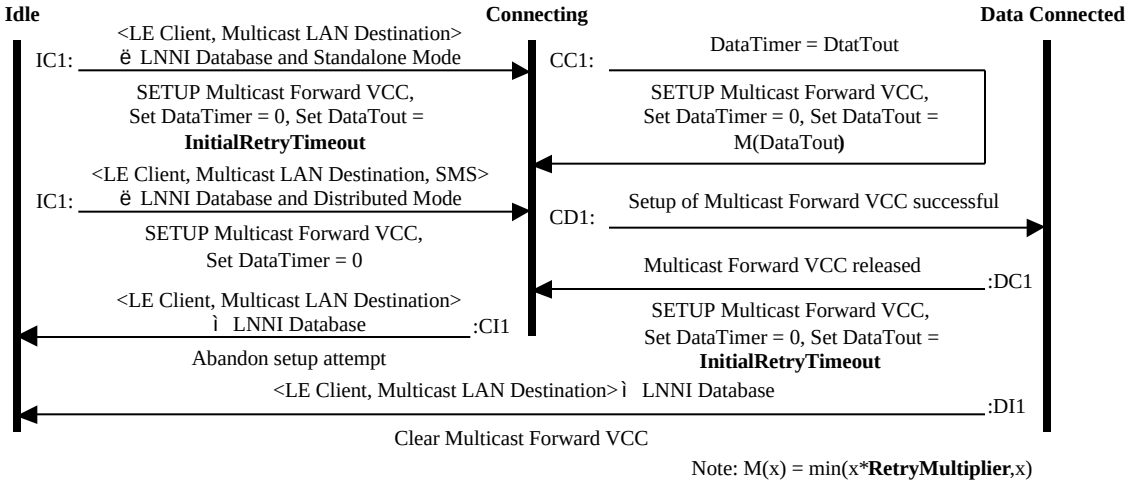

Figure 8-11 SMS-LE Client Data Connection State Machine

CC1: If the Multicast Forward VCC hasn't been setup before the retry timer expires, the local SMS retries establishing the connection.

CD1: If the Multicast Forward VCC setup successfully completes, the <SMS, Multicast LAN Destination> pair moves to the Data Connected state.

CI1: If the <SMS, Multicast LAN Destination> pair is removed from the LNNI Database, then the local SMS abandons the setup of the Multicast Forward VCC and the <SMS, Multicast LAN Destination> pair returns to the Idle state.

DC1: If the Multicast Forward VCC is released, the local SMS attempts to re-establish a Multicast Forward VCC.

DI1: If the <SMS, Multicast LAN Destination> pair leaves the LNNI Database, then the SMS clears the Multicast Forward VCC and the <SMS, Multicast LAN Destination> pair returns to the Idle state.

IC1: If the <LE Client, Multicast LAN Destination> enters the LNNI Database and the local SMS is in Standalone Mode, then the local SMS attempts to setup a Multicast Forward VCC to the LE Client.

IC2: If the <LE Client, Multicast LAN Destination> enters the LNNI Database with the local SMS assigned to service the LE Client and the local SMS is in Distributed Mode, then the local SMS attempts to setup a Multicast Forward VCC to the LE Client.

8.10 Server CSA Cache State Machine

A server (LES or SMS) called the local server runs one Server CSA Cache State Machine for each other server (LES or SMS) in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the other servers will be in the same ELAN with the local server.

The state machine covers the function of detecting which other LESs and SMSs are operational in the ELAN.

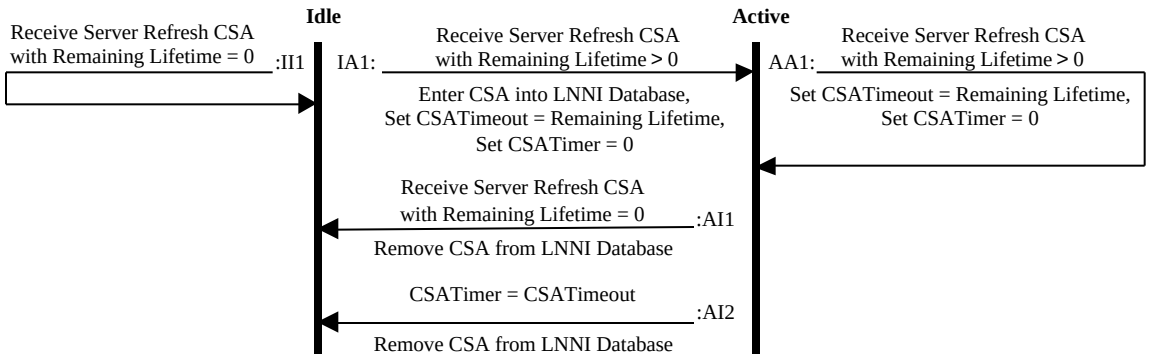

Figure 8-12 Server CSA Cache State Machine

AA1: If a more recent Server (LES or SMS) Refresh CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

AI1: If a more recent Server (LES or SMS) Refresh CSA is received[21] and the Remaining Lifetime in the CSA is = 0, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the Server moves to the Idle state.

AI2: If a CSA cache entry becomes older than the Remaining Life associated with it, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the Server moves to the Idle state.

IA1: If a more recent Server (LES or SMS) Refresh CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) enters the CSA into the LNNI Database, sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

II1: If a Server (LES or SMS) Refresh CSA is received[21] and the Remaining Lifetime in the CSA is = 0, the Server (LES or SMS) stays in the Idle state.

8.11 Server LE Client Join CSA Cache State Machine

A server (LES or SMS) called the local server runs one Server LE Client Join CSA Cache State Machine for each LE Client in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the LE Clients will be in the same ELAN with the local server.

The state machine covers the functions of detecting which LE Clients are operational in the ELAN and detecting and removing an LE Client from the ELAN when its join information is in conflict with another LE Client in the ELAN.

[21] This can occur during either Cache Alignment or during cache updates.

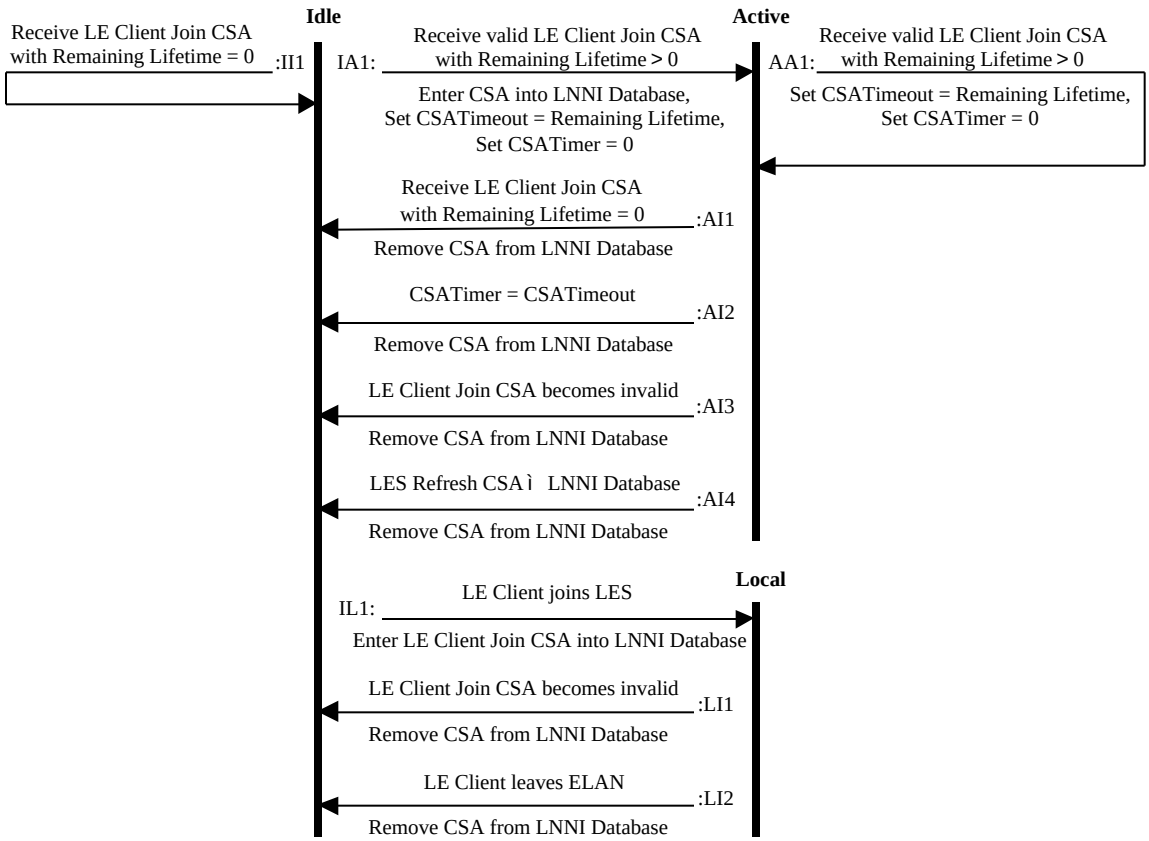

Figure 8-13 Server LE Client Join CSA Cache State Machine

AA1: If a more recent LE Client Join CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

AI1: If a more recent LE Client Join CSA is received[21] and the Remaining Lifetime in the CSA is = 0, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the LE Client moves to the Idle state.

AI2: If a CSA cache entry becomes older than the Remaining Life associated with it, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the LE Client moves to the Idle state.

AI3: If the LE Client Join CSA becomes invalid as per Section 5.4.2.9.5, then the LE Client Join CSA is removed from the LNNI Database and the LE Client is moved to the Idle state.

AI4: If the LES Refresh CSA to which the LE Client is local is removed from the LNNI Database, then the LE Client Join CSA is removed from the LNNI Database and the LE Client moves to the Idle state.

IA1: If a more recent LE Client CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) enters the CSA into the LNNI Database, sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

II1: If a LE Client Join CSA is received[21] and the Remaining Lifetime in the CSA is = 0, the LE Client stays in the Idle state.

IL1: If the LE Client joins the local LES, then the local LES creates a LE Client CSA and enters it into the LNNI Database.

LI1: If the LE Client Join CSA becomes invalid as per Section 5.4.2.9.5, then the LE Client Join CSA is removed from the LNNI Database and the LE Client is moved to the Idle state.

LI2: If the LE Client leaves the ELAN (is no longer joined with the local LES), then the LE Client Join CSA is removed from the LNNI Database and the LE Client is moved to the Idle state.

8.12 Server LE Client Unicast Registration CSA Cache State Machine

A server (LES or SMS) called the local server runs one Server LE Client Unicast Registration CSA Cache State Machine for each <LE Client, Unicast LAN Destination> pair in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the <LE Client, Unicast LAN Destination> pairs will be in the same ELAN with the local server.

The state machine covers the function of tracking which <LE Client, Unicast LAN Destination> pairs are associated with the ELAN.

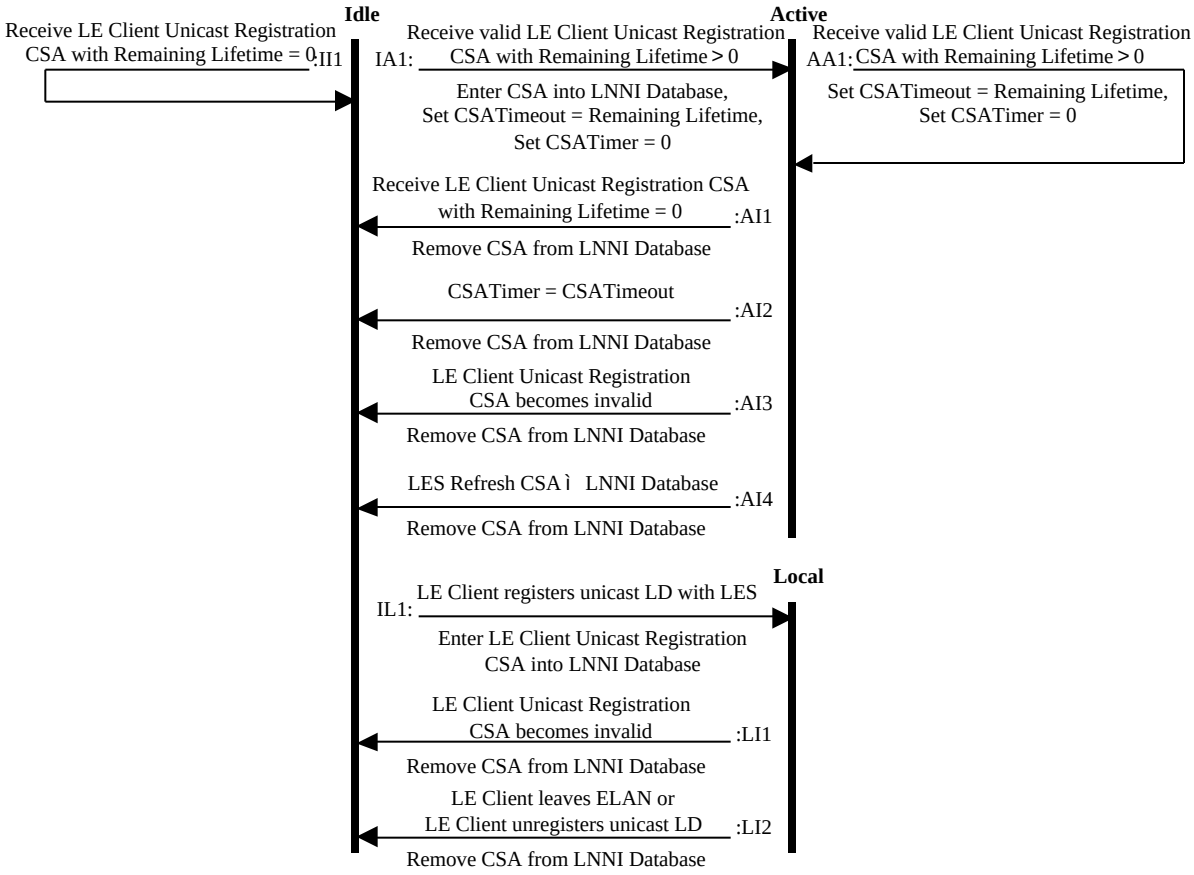

Figure 8-14 Server LE Client Unicast Registration CSA Cache State Machine

AA1: If a more recent LE Client Unicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

AI1: If a more recent LE Client Unicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is = 0, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the <LE Client, Unicast LAN Destination> pair moves to the Idle state.

AI2: If a CSA cache entry becomes older than the Remaining Life associated with it, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the <LE Client, Unicast LAN Destination> pair moves to the Idle state.

AI3: If the LE Client Unicast Registration CSA becomes invalid as per Section 5.4.2.9.8, then the LE Client Unicast Registration CSA is removed from the LNNI Database and the <LE Client, Unicast LAN Destination> pair is moved to the Idle state.

AI4: If the LES Refresh CSA for the LES that registered the unicast LAN Destination is removed from the LNNI Database, then the LE Client Unicast Registration CSA is removed from the LNNI Database and the <LE Client, Unicast LAN Destination> pair moves to the Idle state.

IA1: If a more recent LE Client Unicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) enters the CSA into the LNNI Database, sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

II1: If an LE Client Unicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is = 0, the <LE Client, Unicast LAN Destination> pair stays in the Idle state.

IL1: If a local LE Client registers the unicast LAN Destination, then the local LES creates a LE Client Unicast Registration CSA and enters it into the LNNI Database.

LI1: If the LE Client Unicast Registration CSA becomes invalid as per Section 5.4.2.9.8, then the LE Client Unicast Registration CSA is removed from the LNNI Database and the <LE Client, Unicast LAN Destination> pair is moved to the Idle state.

LI2: If the local LE Client that registered the unicast LAN Destination leaves the ELAN (is no longer joined with the local LES) or the local LE Client unregisters the unicast LAN Destination, then the LE Client Unicast Registration CSA is removed from the LNNI Database and the <LE Client, Unicast LAN Destination> pair is moved to the Idle state.

8.13 Server LE Client Multicast Registration CSA Cache State Machine

A server (LES or SMS) called the local server runs one Server LE Client Multicast Registration CSA Cache State Machine for each <LE Client, Multicast LAN Destination> pair in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the <LE Client, Multicast LAN Destination> pairs will be in the same ELAN with the local server.

The state machine covers the function of tracking which <LE Client, Multicast LAN Destination> pairs are associated with the ELAN.

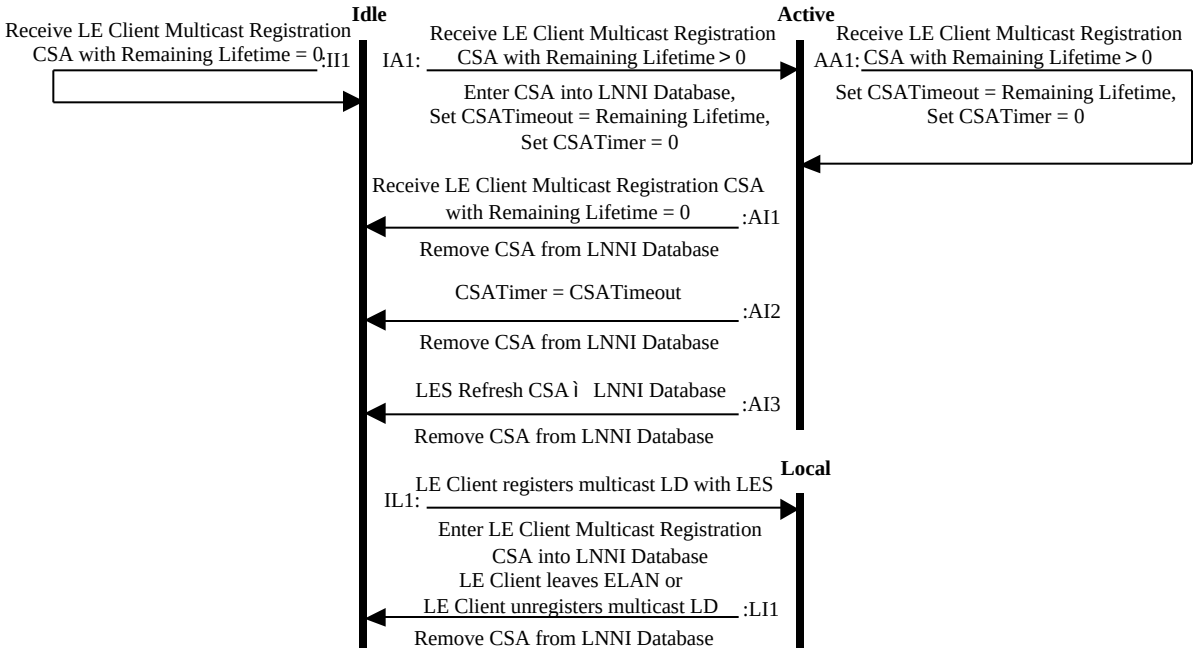

Figure 8-15 Server LE Client Multicast Registration CSA Cache State Machine

AA1: If a more recent LE Client Multicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

AI1: If a more recent LE Client Multicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is = 0, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the <LE Client, Multicast LAN Destination> pair moves to the Idle state.

AI2: If a CSA cache entry becomes older than the Remaining Life associated with it, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the <LE Client, Multicast LAN Destination> pair moves to the Idle state.

AI3: If the LES Refresh CSA for the LES that registered the multicast LAN Destination is removed from the LNNI Database, then the LE Client Multicast Registration CSA is removed from the LNNI Database and the <LE Client, Multicast LAN Destination> pair moves to the Idle state.

IA1: If a more recent LE Client Multicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) enters the CSA into the LNNI Database, sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

II1: If an LE Client Multicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is = 0, the <LE Client, Multicast LAN Destination> pair stays in the Idle state.

IL1: If a local LE Client registers the multicast LAN Destination, then the local LES creates a LE Client Multicast Registration CSA and enters it into the LNNI Database.

LI1: If the local LE Client that registered the multicast LAN Destination leaves the ELAN (is no longer joined with the local LES) or the local LE Client unregisters the multicast LAN Destination, then the LE Client Multicast Registration CSA is removed from the LNNI Database and the <LE Client, Multicast LAN Destination> pair is moved to the Idle state.

8.14 Server SMS Multicast Registration CSA Cache State Machine

A server (LES or SMS) called the local server runs one Server SMS Multicast Registration CSA Cache State Machine for each <SMS, Multicast LAN Destination> pair in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the <SMS, Multicast LAN Destination> pairs will be in the same ELAN with the local server.

The state machine covers the function of tracking which <SMS, Multicast LAN Destination> pairs are associated with the ELAN.

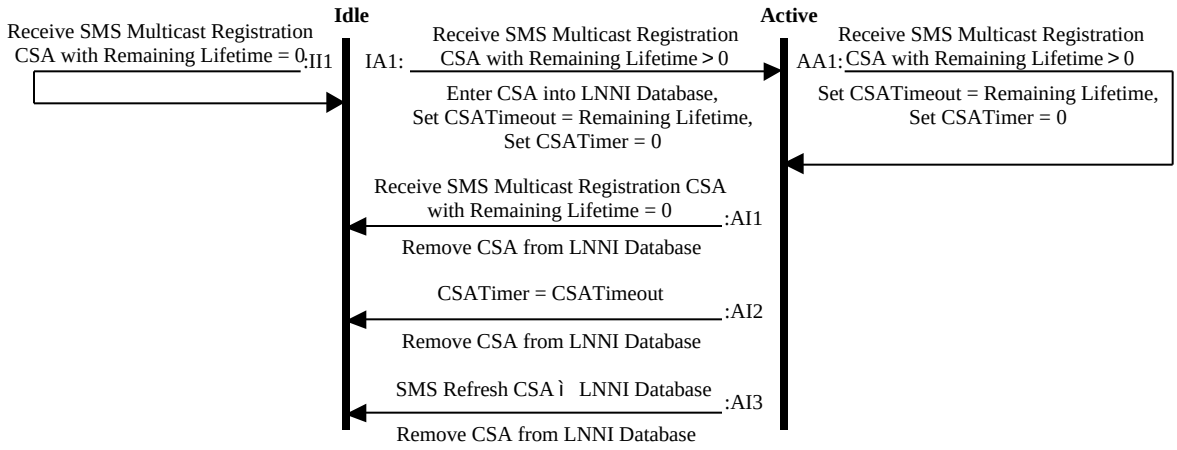

Figure 8-16 Server SMS Multicast Registration CSA Cache State Machine

AA1: If a more recent SMS Multicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

AI1: If a more recent SMS Multicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is = 0, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the <SMS, Multicast LAN Destination> pair moves to the Idle state.

AI2: If a CSA cache entry becomes older than the Remaining Life associated with it, then the local Server (LES or SMS) removes the CSA from the LNNI Database and the <SMS, Multicast LAN Destination> pair moves to the Idle state.

AI3: If the SMS Refresh CSA for the SMS that registered the multicast LAN Destination is removed from the LNNI Database, then the SMS Multicast Registration CSA is removed from the LNNI Database and the <SMS, Multicast LAN Destination> pair moves to the Idle state.

IA1: If a more recent SMS Multicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is > 0, then the local Server (LES or SMS) enters the CSA into the LNNI Database, sets the CSATimeout to the value of the Remaining Lifetime, and sets the CSATimer = 0.

II1: If an SMS Multicast Registration CSA is received[21] and the Remaining Lifetime in the CSA is = 0, the <SMS, Multicast LAN Destination> pair stays in the Idle state.

8.15 LES Local LE Client State Machine

An LES runs one LES Local LE Client State Machine for each LE Client in the universe. The vast majority of these state machines will remain in the Idle state since only a tiny percentage of the LE Clients will be in the same ELAN and join with the LES.

The state machine covers the functions of the LE Client joining the ELAN as local a LE Client of the LES, removing the LE Client from the ELAN when there is a join conflict with another LE Client in the ELAN, and forcing the LE Client to reconfigure when its preferred LES becomes operational.

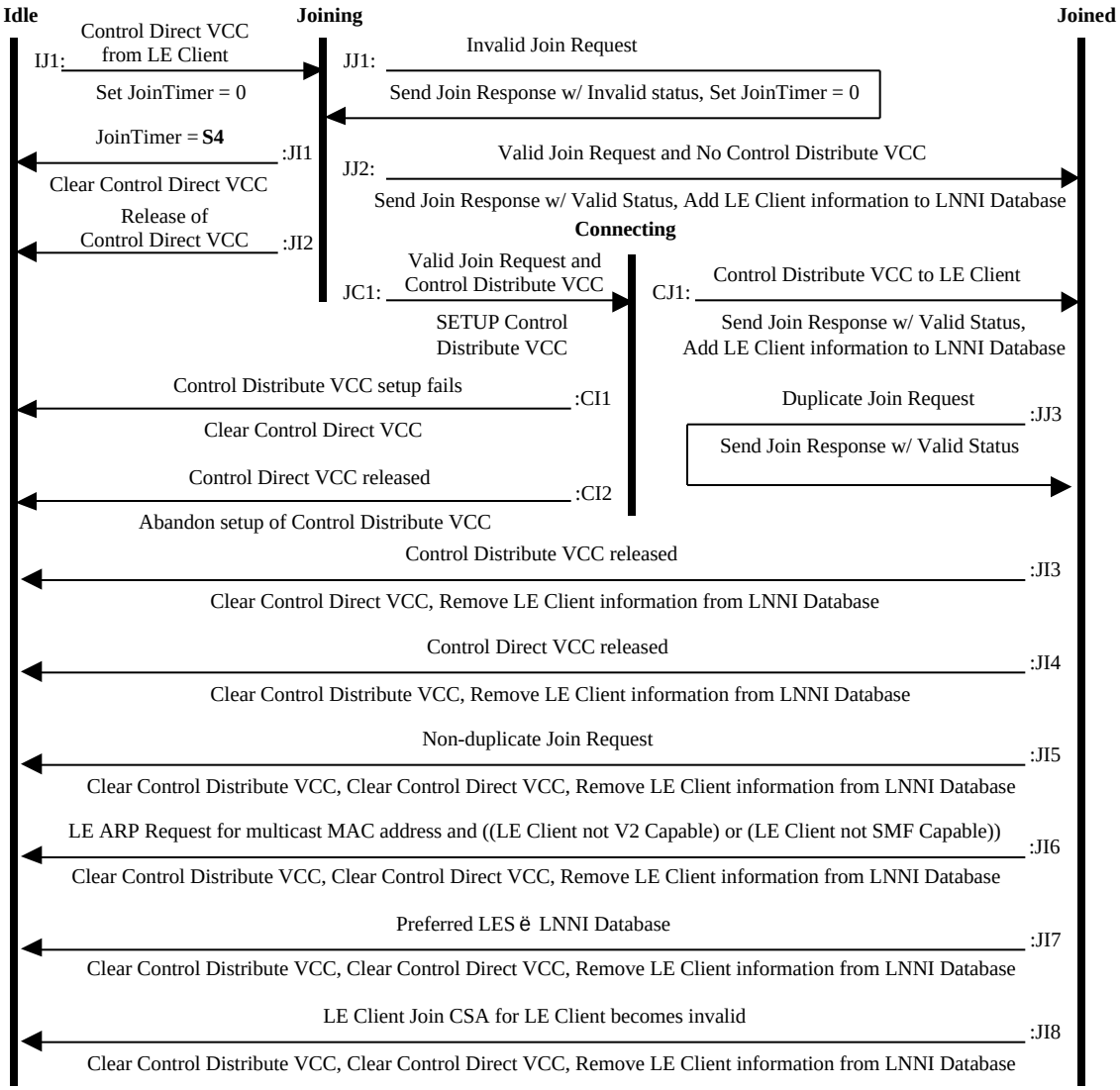

Figure 8-17 LES Local LE Client State Machine – Part 1

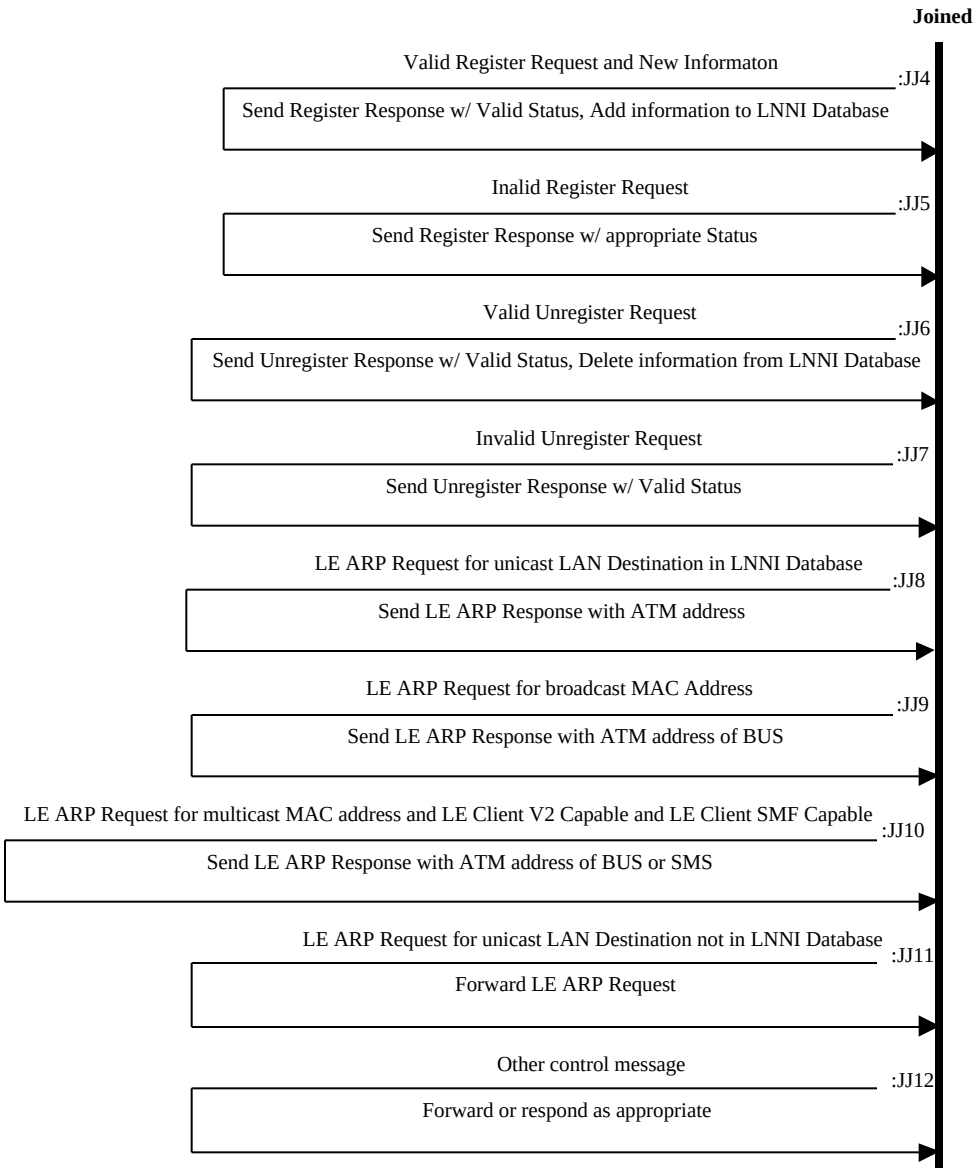

Figure 8-18 LES Local LE Client State Machine – Part 2

CI1: If the Control Distribute VCC setup fails, then the LES clears the Control Direct VCC with Reason Code "Normal, Unspecified."

CI2: If the Control Direct VCC is released, then the LES abandons the setup of the Control Distribute VCC and the LE Client returns to the Idle state.

CJ1: If the Control Distribute VCC setup is successful, then the LES sends a Join Response indicating a successful join and the LES adds the information in the Join Request to the LNNI Database.

IJ1: When the Control Direct VCC is successfully completed to the LES, the LE Client moves to the Joining state.

JC1: If a valid Join Request is received (see Sections 5.4.2.4, 5.4.2.5, 5.4.2.6, and 5.4.2.7 in [5]) and the LES is configured to use a Control Distribute VCC, then the LES initiates the establishment of a Control Distribute VCC to the LE Client.

JI1: If the LES does not receive a Join Request within **S4** seconds (see Section 5.1.2 in [5]), then the LES clears the Control Direct VCC and the LE Client moves to the Idle state.

JI2: If the Control Direct VCC is cleared, then the LE Client moves to Idle State.

JI3: If the Control Distribute is released, then the LES clears the Control Direct VCC with Reason Code "Normal, Unspecified" and all of the LE Client information is removed from the LNNI Database.

JI4: If the Control Direct is released, then the LES clears the Control Distribute VCC with Reason Code "Normal, Unspecified" and all of the LE Client information is removed from the LNNI Database.

JI5: If a Join Request is received that is not a duplicate of the Join Request that caused the transition to the Joined state, then the LES clears both the Control Distribute VCC and the Control Direct VCC with Reason Code "Normal, Unspecified" and all of the LE Client information is removed from the LNNI Database.

JI6: If an LE ARP Request is received from the LE Client which contains a multicast MAC address and the LE Client is not both V2 Capable and SMF Capable (see 7.2.6 or [5]), then the LES clears both the Control Distribute VCC and the Control Direct VCC with Reason Code "Normal, Unspecified" and all of the LE Client information is removed from the LNNI Database.

JI7: If the LE Client provided a Preferred LES in the Join Request and that LES enters the LNNI database, then the LES clears both the Control Distribute VCC and the Control Direct VCC with Reason Code "Normal, Unspecified" and all of the LE Client information is removed from the LNNI Database.

JI8: If the LE Client Join CSA corresponding to the LE Client becomes invalid as per Section 5.4.2.9.5, then the LES clears both the Control Distribute VCC and the Control Direct VCC with Reason Code "Normal, Unspecified" and all of the LE Client information is removed from the LNNI Database.

JJ1: If an invalid Join Request is received (see Sections 5.4.2.4, 5.4.2.5, 5.4.2.6, and 5.4.2.7 in [5]), then the LES sends a Join Response indicating a failed join attempt and resets the JoinTimer.

JJ2: If a valid Join Request is received (see Sections 5.4.2.4, 5.4.2.5, 5.4.2.6, and 5.4.2.7 in [5]) and the LES is configured to not use a Control Distribute VCC, then the LES sends a Join Response indicating a successful join and the LES adds the information in the Join Request to the LNNI Database.

JJ3: If a Join Request is received that is a duplicate of the Join Request that caused the transition to the Joined state, then the LES sends a Join Response indicating a successful join.

JJ4: If a valid Register Request is received (see Section 6.1.2 of [5]) and the information is not already in the LNNI Database, then the LES sends a Register Response indicating a successful Register Request and the LES adds the information to the LNNI Database. If the LAN Destination being registered is a multicast MAC address, the LES can assign the address to an SMS if there is one or more in the LNNI Database that serves this multicast MAC address.

JJ5: If an invalid Register Request is received (see Section 6.1.2 of [5]), then the LES sends a Register Response indicating an unsuccessful Register Request with appropriate SATUS code (see Section 6.1.2 of [5]).

JJ6: If a valid Unregister Request is received (see Section 6.1.2 of [5]), then the LES sends an Unregister Response indicating a successful Unregister Request and the LES removes the information to the LNNI Database.

JJ7: If an invalid Unregister Request is received (see Section 6.1.2 of [5]), then the LES sends an Unregister Response indicating a successful Unregister Request.

JJ8: If the LES receives an LE ARP Request for a unicast LAN Destination that is in the LNNI Database, then the LES sends an LE ARP Response with the ATM address in the LNNI Database associated with the unicast LAN Destination.

JJ9: If the LES receives an LE ARP Request for the all 1's broadcast MAC address, then the LES sends an LE ARP Response with the ATM address of the BUS that is associated with the LES.

JJ10: If an LE ARP Request is received from the LE Client which contains a multicast MAC address and the LE Client is both V2 Capable and SMF Capable (see 7.2.6 or [5]), then the LES sends an LE ARP Response with the ATM address an SMS in the LNNI Database that serves the multicast MAC address. If there is no SMS in the LNNI Database that serves the multicast MAC address, then the LES sends an LE ARP Response with the ATM address of the BUS that is associated with the LES.

JJ11: If the LES receives an LE ARP Request for a unicast LAN Destination that is not in the LNNI Database, then the LES forwards the LE ARP Request according to the forwarding rules in Section 5.6.4.1.

JJ12: If any other control message is received from either the LE Client or from another LES that requires forwarding to the LE Client, the LES responds to the control message or forwards it as appropriate.

8.16 LECS Main State Machine

A main state machine runs for the LECS. This is responsible for initializing the LECS and driving all the other LECS state machines namely ServerConfigProxy, LEClientConfigProxy and ServicePeerLecsProxy.

The ServerConfigProxy state machine runs one per active LES/SMS and is responsible for providing the initial configuration and updating it. The ServicePeerLecsProxy state machine runs one per peer LECS and is responsible for synchronization with the peer LECS and the LEClientConfigProxy state machine runs one per connected LE Client and is responsible for conveying configuration information to the LEClient.

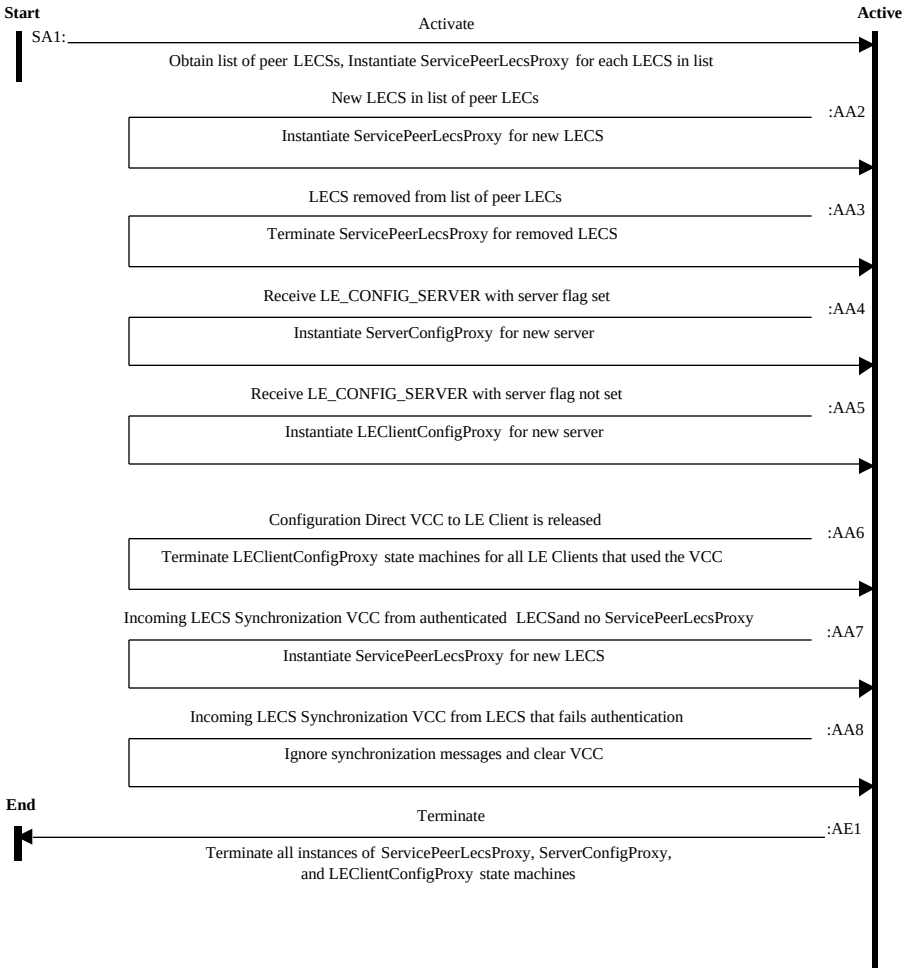

Figure 8-19 LECS Main State Machine

AA2: If a new LECS peer gets added,[22] an instance of ServicePeerLecsProxy is created for it which is responsible for LECS-LECS synchronization with this LECS.

AA3: If a peer LECS gets removed,[22] then the ServicePeerLecsProxy State Machine for that LECS is terminated.

AA4: If an LE_CONFIG_SERVER is received and the server flag is set to IS_LE_SERVER or IS_LE_MCAST then a new instance of ServerConfigProxy state machine is created which is responsible for all configuration handling for the server sending the configuration request.

AA5: If an LE_CONFIG_SERVER is received and the server flag is neither IS_LE_SERVER nor IS_LE_MCAST then an instance of LEClientConfigProxy state machine is created which is responsible for configuration of LE Clients.

AA6: If a Configuration Direct VCC to an LE Client is released, the LEClientConfigProxy state machine for each LE Client that used the VCC is terminated.

AA7: If another LECS attempts to setup a LECS Synchronization VCC to this LECS and the calling LECS is authenticated, then a ServicePeerLecsProxy state machine is instantiated if there is no existing ServicePeerLecsProxy for the calling LECS.

[22] This can occur in several ways including configuration and a repetition of the ILMI procedure.

AA8: If another LECS attempts to setup a LECS Synchronization VCC to this LECS and the calling LECS fails authentication, then all synchronization Messages on this VCC are discarded and the VCC is released.

AE1: If the LECS is terminated, then all instances of ServicePeerLecsProxy , ServerConfigProxy and LEClientConfigProxy state machines are terminated and the LECS moves to the End state.

SA1: Whenever a LECS is operational it should be activated which causes it to obtain[23] the list of peer LECS, instantiate a ServicePeerLecsProxy state machine for each element in the list, and move to the Active state.

8.17 ServicePeerLecsProxy State Machine

The ServicePeerLecsProxy state machine runs one per peer LECS and is responsible for establishing and maintaining the LECS Synchronization VCC with the peer LECS which is used for synchronizing the configuration information between the LECS's serving the same ELAN.

The ServicePeerLecsProxy state machine is created by the LECS Main state machine.

[23] The list can be obtained in several ways including configuration and ILMI. The state machine does not capture this detail.

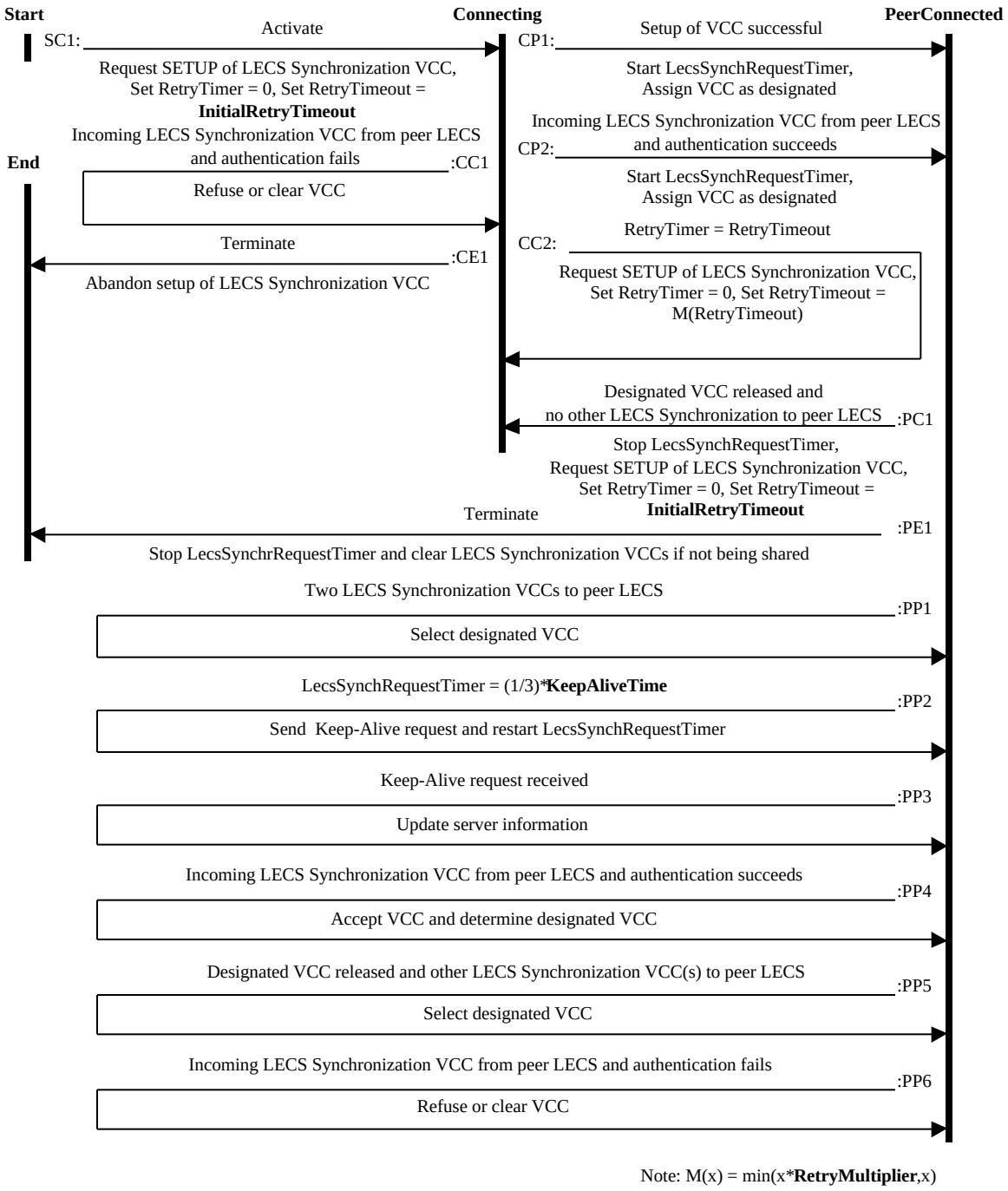

Figure 8-20 ServicePeerLecsProxy State Machine

CC1: If an incoming request for a Synchronization VCC is received from the peer LECS but the authentication fails, then the LECS either refuses the VCC or clears the VCC if has been accepted before completing the authentication.

CC2 If the setup of the VCC fails and the retry timer expires, the state machine retries the setup of the LECS Synchronization VCC.

CE1: If a Terminate event is received the ServicePeerLecsProxy abandons the setup of the LECS Synchronization VCC and terminates.

CP1: If the LECS Synchronization VCC gets established, then the ServicePeerLecsProxy starts a LecsSyncRequestTimer, assigns the VCC as the designated Synchronization VCC and moves to PeerConnected state. The designated LECS Synchronization VCC is used for sending all the messages to the peer.

CP2: If an incoming request for a Synchronization VCC is received from the peer LECS and the authentication succeeds, the VCC is accepted and assigned as the designated VCC and the LecsSynchRequestTimer is started.

PC1: If the designate VCC is released and there is no other LECS Synchronization VCC to the peer, the state machine attempts to setup a new LECS Synchronization VCC.

PE1: If a Terminate event is received the LecsSynchRequestTimer is stopped and each LECS Synchronization VCC to the peer LECS that is not being shared by another entity is cleared.

PP1: If two LECS Synchronization VCCs become established with the peer LECS, the designated LECS Synchronization VCC is chosen according to the LUNI rules.

PP2: If the LecsSyncRequestTimer = one third of **KeepAliveTime** seconds, the ServicePeerLecsProxy sends a Keep-Alive request to the peer LECS and restarts the LecsSyncRequestTimer.

PP3: If a Keep-Alive request is received, the information about active LESs and SMSs is updated from the contents of the Keep-Alive request.

PP4: If an incoming request for a Synchronization VCC is received from the peer LECS and the authentication succeeds, the VCC is accepted and the designated VCC is chosen according to the LNNI rules.

PP5: If the designated LECS Synchronization VCC is released and there are more LECS Synchronization VCCs to the peer LECS, then a new designated Synchronization VCC is chosen according to the LNNI rules.

PP6: If an incoming request for a Synchronization VCC is received from the peer LECS but the authentication fails, then the LECS either refuses the VCC or clears the VCC if has been accepted before completing the authentication.

SC1: Whenever a ServicePeerLecsProxy is activated it requests the setup of a LECS Synchronization VCC to the peer LECS and moves to Connecting state.

8.18 ServerConfigProxy State Machine

A ServerConfigProxy state machine runs for each active server (LES/SMS). This is controlled by the LECS Main state machine which is responsible for its activation and termination.

The ServerConfigProxy state machine covers the function of sending the configuration information, transmitting and receiving Keep-Alive messages and transmitting the configuration change trigger to the Servers (LES and SMS)

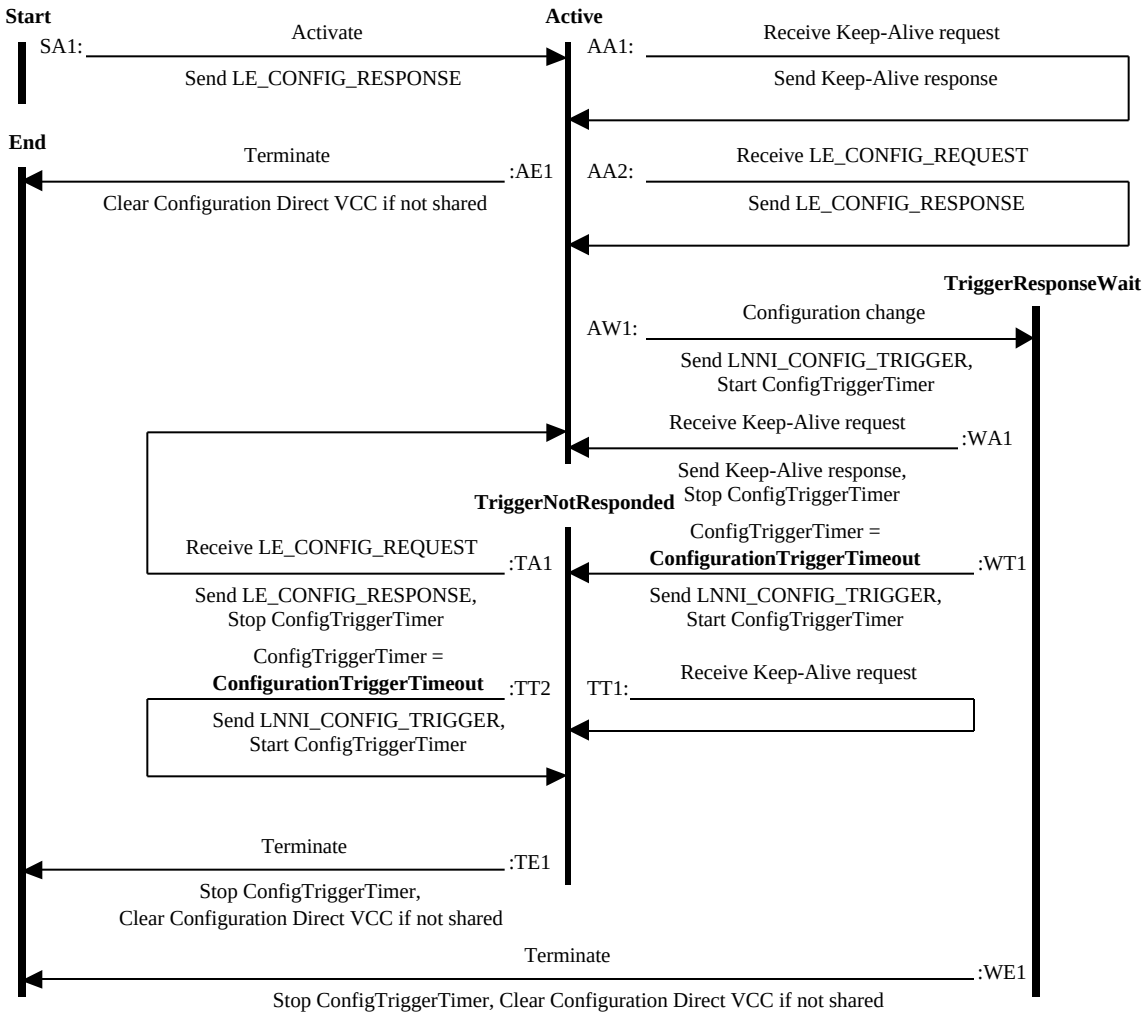

Figure 8-21 ServerConfigProxy State Machine

AA1: If a Keep-Alive Request is received, a Keep-Alive Response is sent out.

AA2: If an LE_CONFIG_REQUEST is received LE_CONFIG_RESPONSE is sent out.

AE1: If a terminate event is received, the ServerConfigProxy clears the Configuration Direct VCC if it is not shared by other entities and terminates.

AW1: If a Configuration change occurs, then the ServerConfigProxy sends a LNNI_Config_Trigger to the server, starts the ConfigTriggerTimer, sets the TriggerRetryCounter to Zero and moves to TriggerResponseWait state.

SA1: Whenever a ServerConfigProxy is activated, the LE_CONFIG_SERVER message should be passed on to it. The ServerConfigProxy sends a LE_CONFIG_RESPONSE and moves to Active state.

TA1: If an LE_CONFIG_REQUEST is received an LE_CONFIG_RESPONSE is sent out, the ConfigTriggerTimer is stopped and the ServerConfigProxy moves to the Active state.

TE1: If a Terminate event is received the ServerConfigProxy stops the ConfigTriggerTimer, clears the Configuration Direct VCC if it is not shared by other entities and terminates.

TT1: If a Keep-Alive request is received it is ignored.

TT2: If the ConfigTriggerTimeout event is received the ServerConfigProxy re-sends the LNNI_CONFIG_TRIGGER and starts the ConfigTriggerTimer.

WA1: If a LE_CONFIG_REQUEST is received the ServerConfigProxy sends a LE_CONFIG_RESPONSE, stops the ConfigTriggerTimer, and moves to the Active state

WE1: If a Terminate event is received the ServerConfigProxy stops the ConfigTriggerTimer, clears the Configuration Direct VCC if it is not shared by other entities and terminates.

WT1: If a ConfigTriggerTimeout event is received the ServerConfigProxy re-sends the LNNI_CONFIGURE_TRIGGER, starts the ConfigTriggerTimer and moves to the TriggerNotResponded state.

8.19 LEClientConfigProxy State Machine

The LEClientConfigProxy state machine runs one per connected LE Client and is responsible for conveying configuration information to the LE Client. This is controlled by the LECS Main state Machine which is responsible for its activation and termination.

The LEClientConfigProxy state machine covers the function of downloading the configuration information to the LE Clients.

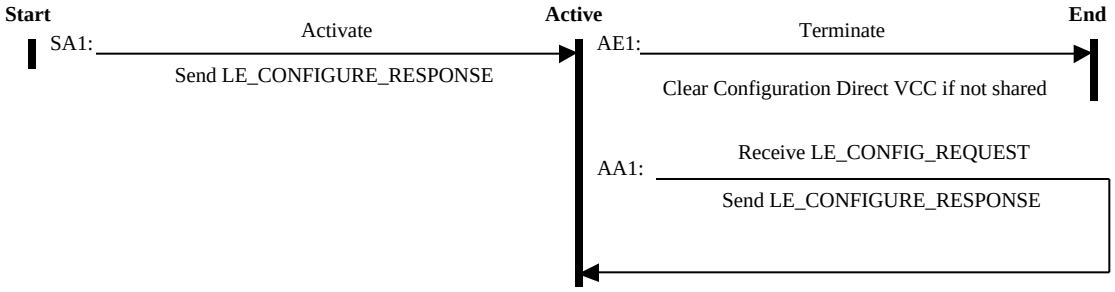

Figure 8-22 LEClientConfigProxy State Machine

AA1: If an LE_CONFIG_REQUEST is received, an LE_CONFIG_RESPONSE is sent out.

SA1: When the LEClientConfigProxy is activated, the LE_CONFIG_SERVER message is passed on to it. The LEClientConfigProxy sends a LE_CONFIG_RESPONSE and moves to Active state.

AE1: If a Terminate event is received the LEClientConfigProxy clears the Configuration Direct VCC if it is not being shared by other entities and terminates.

8.20 BUS State Machine

The BUS runs a state machine that is responsible for obtaining a list of peer BUSs and SMSs, and for creating a MulticastTarget state machine to represent each BUS, SMS, or LEClient to which it must forward multicast packets.

The BUS forwards incoming-multicast-send-connect indications to the MulticastTarget state machine, but the MulticastTarget state machine handles all other signalling events.

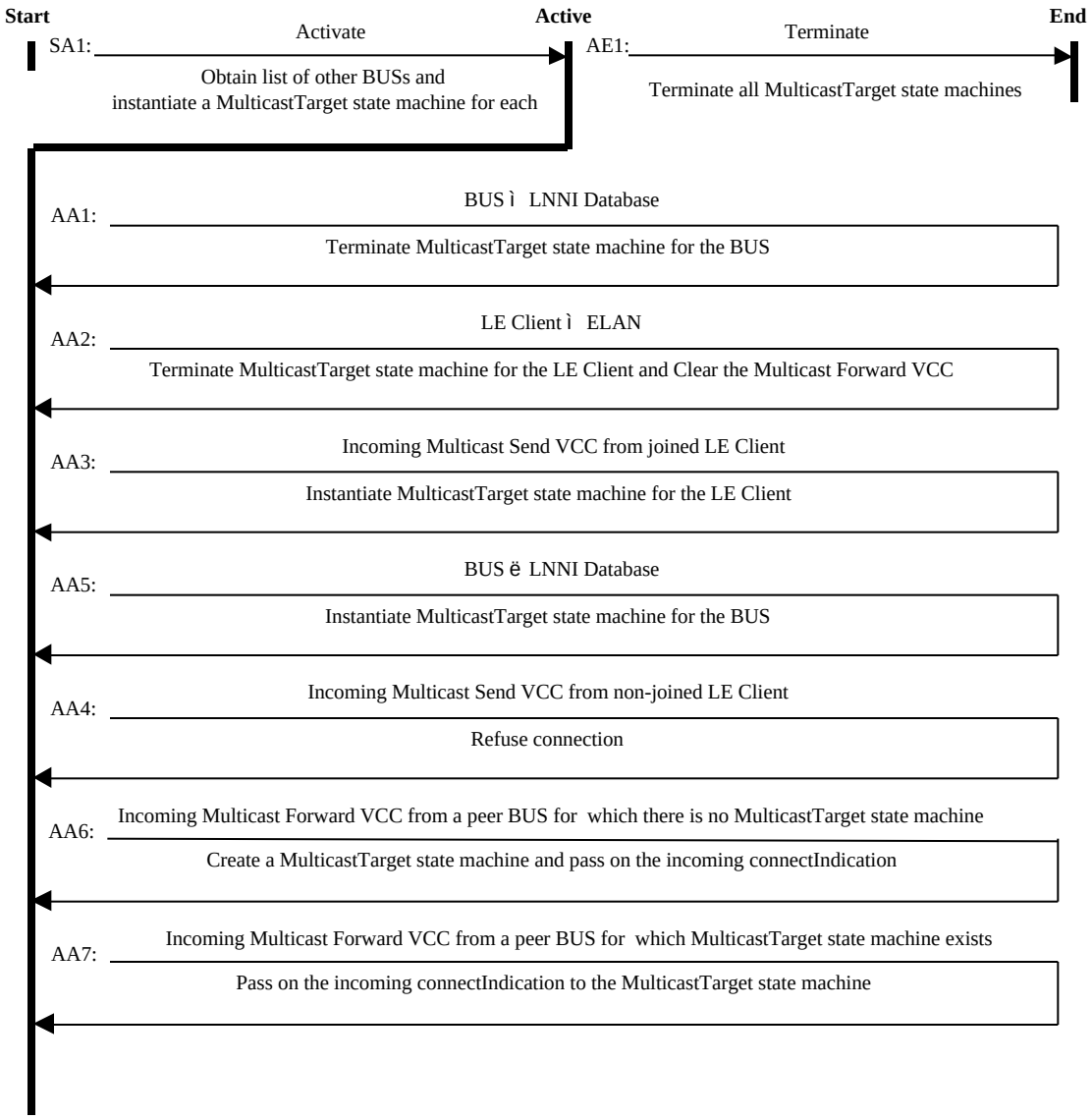

Figure 8-23 BUS State Machine

AA1: If a peer BUS entry gets removed from the LNNI database, the BUS terminates the MulticastTarget state machine for that BUS.

AA2: If an LE Client leaves the ELAN, the BUS terminates the MulticastTarget state machine for that LE Client and releases the MulticastSendVcc to the LE Client.

AA3: When there is an incoming Multicast Send VCC from an LE Client and that LE Client is joined with the LES, a MulticastTarget state machine is instantiated for the LE Client.

AA4: When there is an incoming Multicast Send VCC from an LE Client and that LE Client is not joined with the LES, the connection setup is refused.

AA5: When a new BUS enters the LNNI Database, the BUS responds by creating a MulticastTarget state machine to for the new BUS.

AA6: When there is an incoming Multicast Forward VCC from a peer BUS and there is no MulticastTarget state machine instantiated for this BUS, When the BUS receives a MulticastFwd-connect indication, and a Multicast Target does not exist for the peer, a MulticastTarget state machine is created for the peer BUS and the incoming connectIndication passed on to it.

AA7: When there is an incoming Multicast Forward VCC from a peer BUS and there exists a MulticastTarget state machine for this BUS, the incoming VCC connectIndication is passed on to the MulticastTarget state machine.

AE1: Whenever a BUS is terminated, it terminates all the MulticastTarget state machines and moves to the End state.

SA1: Upon activation, the BUS obtains the peerList (list of all other local BUSs in the ELAN) from its associated LES and it instantiates a MulticastTarget state machine for each.

8.21 PeerMulticastTarget State Machine

The PeerMulticastTarget state machine runs one per active peer BUS and is responsible for setting up and maintaining Multicast Forward VCCs.

The PeerMulticastTarget state machine is created by the BUS state machine.

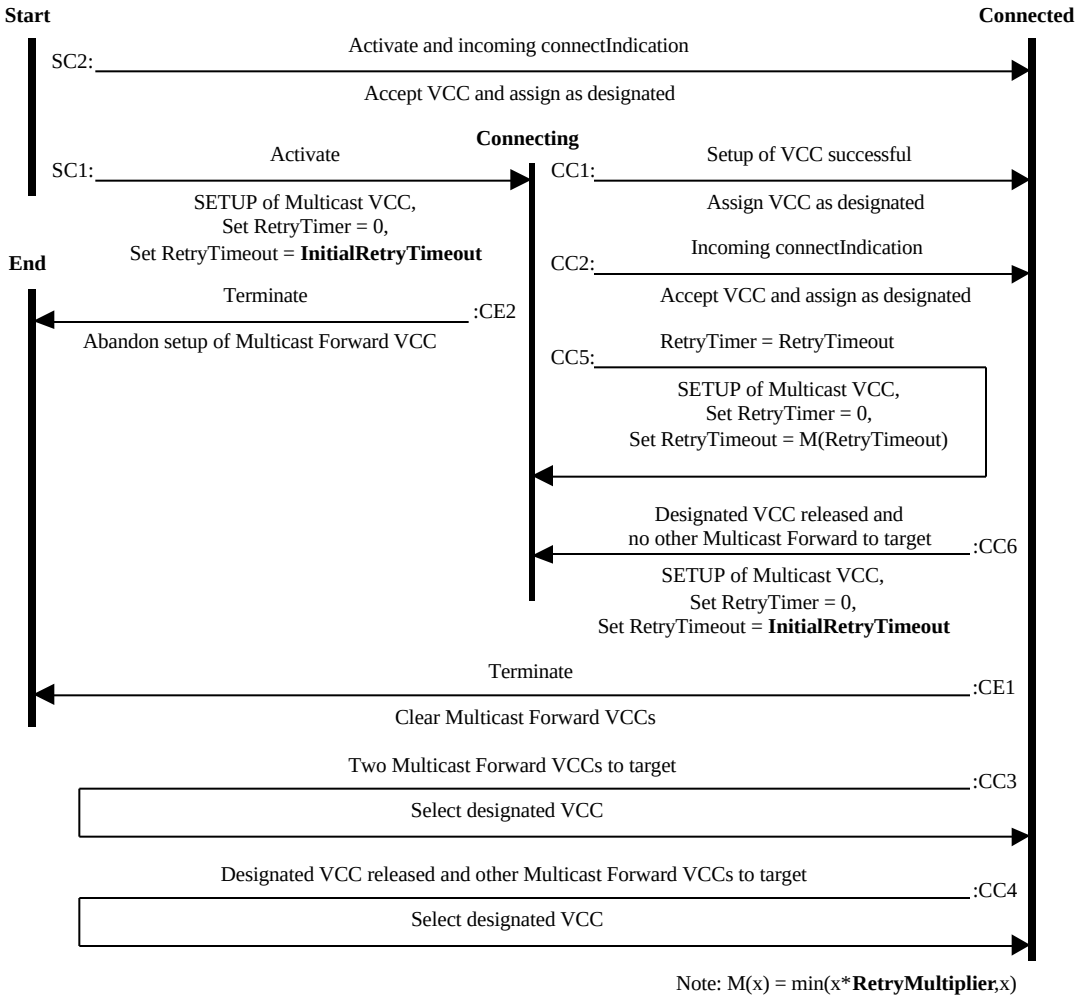

Figure 8-24 PeerMulticastTarget State Machine

CC1: If the setup is successful the state machine assigns the VCC as the designated VCC and moves to the Connected state. The designated VCC will be used for sending out all messages to the peer.

CC2: If an incoming VCC connectIndication is received from the BUS state machine, the state machine accepts the VCC, assigns the VCC as the designated VCC and moves to the Connected state.

CC3: If two Multicast Forward VCCs get established to the peer, the designated VCC is chosen according to the LNNI rules.

CC4: If the designated Multicast Forward VCC is released and there are more Multicast Forward VCCs to the target, then a new designated VCC is chosen according to the LNNI rules.

CC5: If the setup of the VCC fails and the retry timer expires, the state machine retries the setup of the Multicast VCC.

CC6: If the designated VCC is released and there is no other Multicast Forward VCC to the peer, the state machine tries to setup a Multicast VCC.

CE1: If a terminate event is received the state machine clears the VCCs and moves to the End state.

CE2: If a terminate event is received the state machine abandons the connection setup and terminates.

SC1: Whenever the state machine is activated with no incoming VCC connectIndication, it sends a connection setup request for a Multicast VCC to the peer and moves to the Connecting state.

SC2: Whenever the state machine is activated with an incoming VCC connectIndication it accepts VCC, assigns the VCC as the designated VCC for sending all requests to this target and moves to the Connected state.

8.22 LEClientMulticastTarget State Machine

The LEClientMulticastTarget state machine runs in a BUS with one per active local LE Client and is responsible for setting up and maintaining Multicast Forward VCCs.

The LEClientMulticastTarget state machine is created by the BUS state machine.

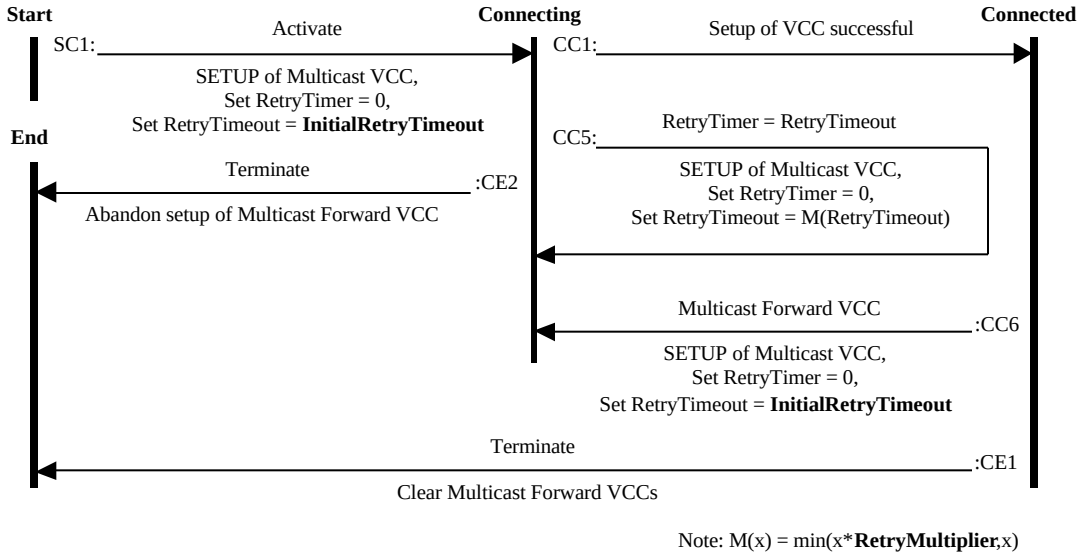

Figure 8-25 PeerMulticastTarget State Machine

CC1: If the setup is successful the state machine moves to the Connected state.

CC5: If the setup of the VCC fails and the retry timer expires, the state machine retries the setup of the Multicast VCC.

CC6: If the Multicast Forward VCC is released, the state machine tries to setup a new Multicast Forward VCC.

CE1: If a terminate event is received the state machine clears the VCCs and moves to the End state.

CE2: If a terminate event is received the state machine abandons the connection setup and terminates.

SC1: Whenever the state machine is activated, it sends a connection setup request for a Multicast VCC to the peer and moves to the Connecting state.

[End of Document]