LTE MME Reference Application

Service Definition

1092417 1.11a

LTE MME Reference Application

Service Definition

1092417 1.11a

Radisys Corporation

9450 Carroll Park Drive San Diego, CA 92121-2256

Phone: *+1 (858) 882-8800*

Fax: *+1 (858) 777-3389*

Web: *http://www.radisys.com*

LTE MME Reference Application
Service Definition
1092417 1.11a

Contributors: Continuous Computing Development Team, Naveen D’cruz.

Printed in U.S.A.

Contents

Figures

Tables

Preface

Objective

This document provides a detailed description of the services at the layer manager interface by the LTE MME Reference Application software (p/n 1000417), designed by Continuous Computing Corporation. This product is referred to as MME Application in the rest of the document.

Audience

Continuous Computing assumes that the readers of this document are familiar with telecommunication protocols, specifically LTE.

Document Organization

This document is organized into the following sections.

Section	Contents
1 Introduction	Describes the product and the interworking process.
2 Layer Manager Interface	Describes the layer manager interface primitives and procedures specific to MME Application.
3 MME Application Initialization and Shutdown	Defines the mechanisms to initialize the MME Application and preparing it for execution. It also describes the procedures to shutdown.

Document Set

The suggested reading order of this document set is:

1. *LTE MME Reference Application Functional Specification*

   Describes the features and highlights the protocol and system characteristics of the software, including the memory characteristics and conformance details.

2. *LTE MME Reference Application Service Definition*

   Describes the procedures and the layer manager interface that are used to pass information between the software and the other software elements. The Interface Primitives section describes the software services. The Procedures section describes and shows the flow of primitives and messages across the interfaces.

3. *SZT Interface Service Definition*

   Provides details about the internal lower layer primitives for the SZT interface with Trillium LTE S1AP Product. The Interface Primitives section describes the software services. The Interface Service Definition describes the interface procedures defined for the service provider software.

4. *S1AP Service Definition*

   Describes the procedures for S1AP at the layer manager interface. The Layer manager interface is used to configure, control and collect the status and statistics information from the S1AP software.

5. *EGT Interface Service Definition*

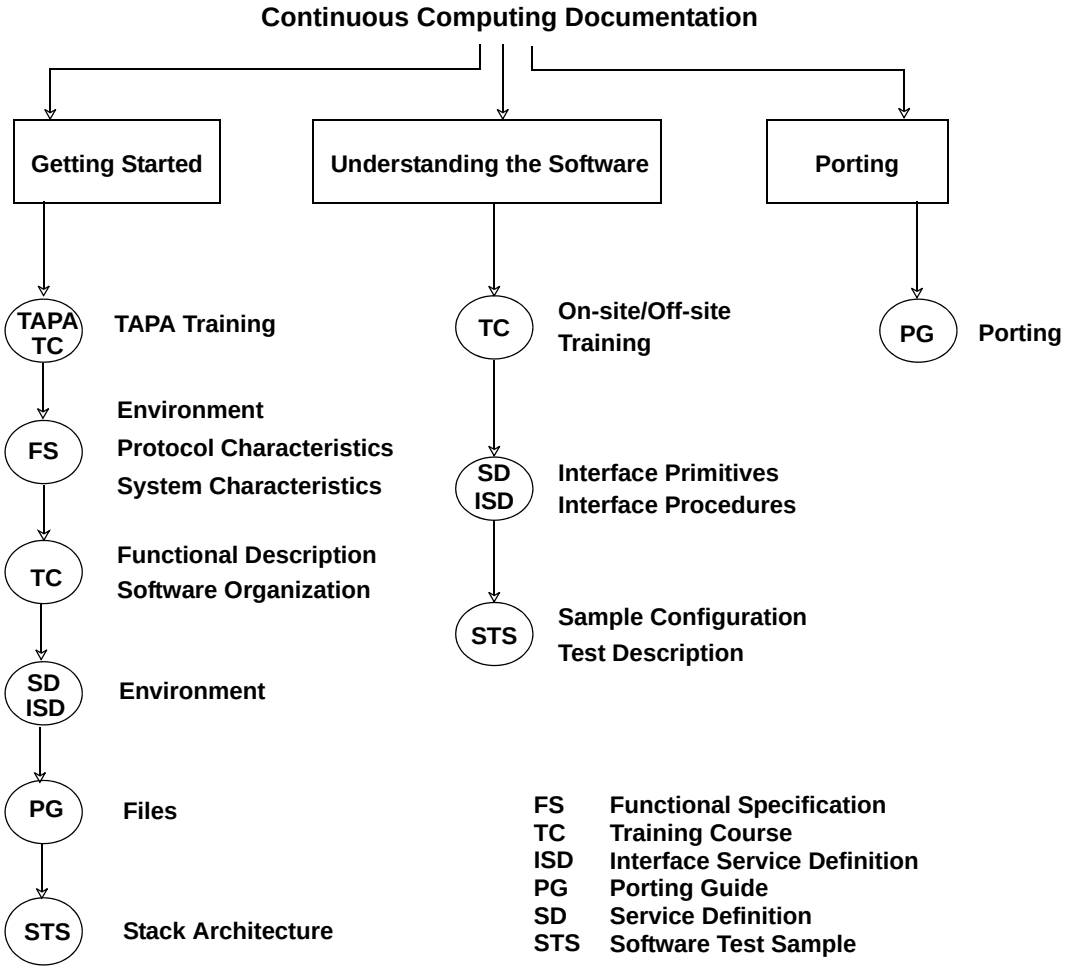
   Provides details about the internal lower layer primitives for the EGT interface with Trillium LTE eGTP Product. The Interface Primitives section describes the software services. The Interface Service Definition describes the interface procedures defined for the service provider software.

6. eGTP *Service Definition*

   Describes the procedures for eGTP at the layer manager interface. The Layer manager interface is used to configure, control, and collect the status and statistics information from the eGTP software.

Using Continuous Computing® Documentation

The following figure shows the various user approaches to using the software documentation. First time users must read the documents under the **Getting Started** column, where important sections and subsections are listed to the right of each document. For users familiar with the documentation, but who need to look up certain points concerning software use, **Understanding the Software** column is suggested. The **Porting** column is for users familiar with Trillium software and related telecommunications protocols and wish to install the software immediately onto their development environments.



Notations

This table displays the notations used in this document:

Notation	Explanation	Examples
Arial	**Titles**	**1.1 Title**
Book Antiqua	Body text	This is body text.
Bold	**Highlights information**	**Loose coupling, tight coupling, upper layer interface**
ALL CAPS	CONDITIONS, MESSAGES	AND, OR CONNECT ACK
Italics	*Document names, emphasis*	*MME Application Service Definition.* This adds *emphasis*.
Courier New Bold	**Code** **Filenames, pathnames**	**PUBLIC S16 VbMiLvbCfgReq (pst, cfg)** **Pst *pst;** **VbMngmt *cfg;**

Release History

This table lists the history of changes in successive revisions to this document:

Version	Date	Author (s)	Description
1.11a	October 20, 2011	Naveen Dcruz H	Addendum release for Radisys logo and template upgrade.
1.1	June 10, 2010	Raja Kumar DT	Initial release.

1

Introduction

1.1 Product Interfaces

Figure 1-1 shows the MME Application interfaces.

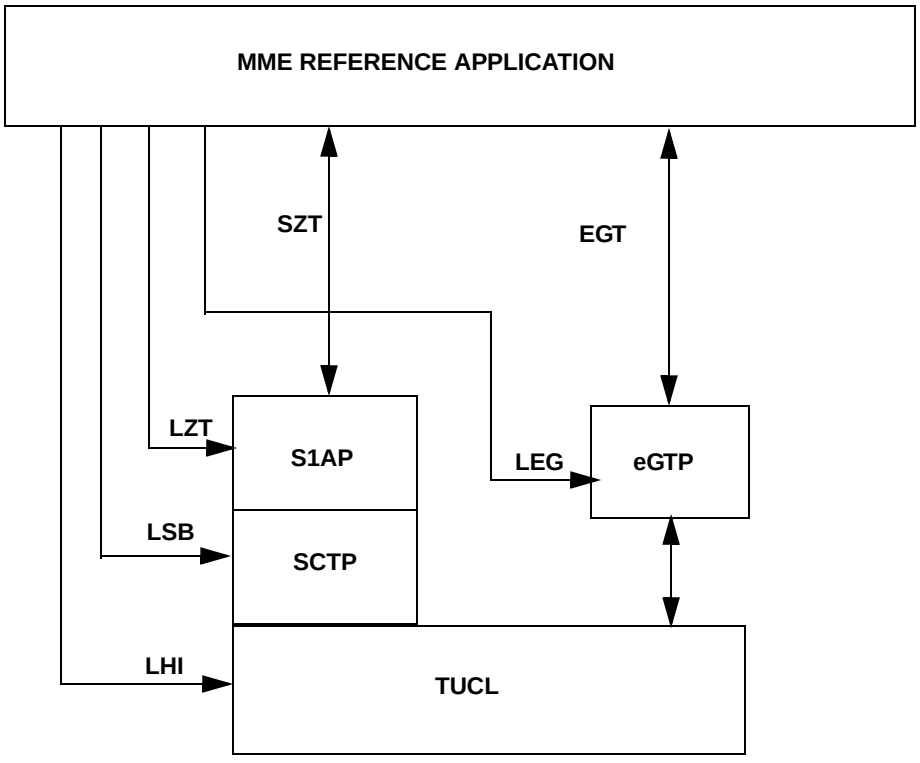


Figure 1-1: Trillium Advanced Portability Architecture

The following table lists the interfaces and describes their functions.

Table 1-1: LTE MME Reference Application interfaces

Interface	Description
System Services Interface (SSI)	Provides functions such as buffer management, timer management, date/time management, resource checking, initialization. Refer to the *System Services Interface Service Definition* for details.
Layer Manager Interface (LMI)	Provides the necessary functions to configure, control, and monitor the condition of each protocol layer. The interface is described later in this document.
Upper Interface (UI)	MME Application is at top most layer in the protocol stack, it does not have any upper user.
Lower Interface (SZT and EGT)	MME Application supports two lower layers- SZT interface with S1AP and EGT interface with eGTP. The lower interface (SZT) provides the necessary primitives for MME Application to configure S1AP layer parameters and transfer NAS Messages. S1AP layer is used by the MME Application layer to access its service providers (lower layer) services. Refer to the *SZT Interface Service Definition* for a detailed description of this interface. The lower interface (EGT) provides the necessary primitives for MME Application to configure eGTP layer parameters and data transfer. eGTP layer is used by the MME Application layer to access its service provider's (lower layer) services. Refer to the EGT *Interface Service Definition* for a detailed description of this interface.

MME Application interacts with the other layers and the layer manager by using the primitives and Service Access Points (SAPs) that are described later**.** MME Application also interacts with system services by using a simple function interface**.**

1.2 Product Entry Points

Initialization Entry Point **`vbActvInit`** - Initializes a task for MME Application.

Task Entry Point **`vbActvTsk`** - Schedules a layer activation task for MME Application.

Timer Entry Point **`vbActvTmr`** - Activates a task for MME Application timers.

1.3 Acronyms

The following acronyms are used in this document:

Table 1-2: Abbreviations

Abbreviation	Expansion
ANSI	American National Standards Institute
LI	Lower Interface
LMI	Layer Manager Interface
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
SAP	Service Access Point
SDU	Service Data Unit
SSI-SDK	System Services Interface - Software Development Kit
TAPA	Trillium Advanced Portability Architecture
UI	Upper Interface

For a list of commonly used terms, refer to the Engineering Glossary (part numberPREN026) at http://www.ccpu.com/search/glossary/

Product-specific Glossary

Abbreviation	Expansion
CNE	Core Network Emulator
eGTP/e-GTP	Evolved GTP
ENB/eNB/ eNodeB	Evolved Node B
FDD	Frequency Division Duplex
MAC	Medium Access Control
MME	Mobile Management Entity
NAS	Non Access Stratum
P-GW/PGW	PDN Gateway
RB	Radio Bearer
RLC	Radio Link Control

Abbreviation	Expansion
RRC	Radio Resource Control
S1AP	S1 Application Protocol
SCTP	Stream Control Transmission Protocol
SRB	Signalling Radio Bearer
SG	Serving Gateway
SM	Stack Manager
LTE	Long Term Evolution
TDD	Time Division Duplex
TUCL	TCP/UDP Convergence Layer
UE	User Equipment

2

Layer Manager Interface

This section describes the layer manager interface primitives, data structures, and procedures specific to MME Application. The layer manager interface for MME Application is described in the following subsections.

2.1 Primitive Listing

The layer manager interface provides the following functions.

Configuration

This function configures the protocol layer resources.

Table 2-1: Configuration primitives

Name	Description	Reference
`VbMiLvbCfgReq`	Configuration request	Section 2.3.1
`VbMiLvbCfgCfm`	Configuration confirm	Section 2.3.2

Unsolicited Status

This function indicates a change in the status of the protocol layer.

Table 2-2: Unsolicited status primitives

Name	Description	Implemented by
`VbMiLvbStaInd`	Status indication	Section 2.3.3

Control

This function activates and deactivates the protocol resources.

Table 2-3: Control primitives

Name	Description	Implemented by
VbMiLvbCntrlReq	Control request	Section 2.3.4
VbMiLvbCntrlCfm	Control confirm	Section 2.3.5

2.2 Product-specific Structures

Each management primitive has two common parameters like: **VbMngmt**, and **Pst**. These parameters are described here.

2.2.1 VbMngmt

The management structure carries all the information required to configure and control the MME Application layer.

```
typedef struct VbMngmt
    {
        Header          hdr;              /* Header */
        CmStatus        cfm;              /* Result of  operation */
        union
        {
            VbCfg       cfg;              /* Configuration */
            VbCntrl     cntrl;            /* Control */
            VbUstaDgn   usta;             /* Unsolicitated status */
        }u;
} VbMngmt;
```

Table 2-4: VbMngmt Data Structure

Data Field	Description	Reference
hdr	Header	--
cfm	Specify whether operation is success or not.	--
cfg	Carry all the configuration parameters for the MME Application layer.	Section 2.3.1.1
cntrl	Control structure	Section 2.3.4
usta	Unsolicited status	Section 2.3.3

This structure is used in every management primitive invoked.

2.2.2 Pst

In TAPA, a system consists of multiple TAPA entities or tasks the post structure:

- Is used to exchange and communicate primitives between various tasks.
- Is the first parameter of all UI, MI, and LI primitives
- Contains all the information required to identify the source and destination TAPA tasks. When the interface is loosely coupled between the source and the destination layers, the source layer provides all the information required by the system services to route the message buffer to the correct destination layer. In the destination layer, the post structure is only used to verify the identity of the source and to identify the specific primitive.
- Contains information that allows a message scheduler to schedule message efficiently. Each message can be assigned a priority. Messages can also be broadcast to all the tasks.

```
typede struct   pst             /* Parameters for SPstTsk */
{
   ProcId      dstProcId;      /* Destination processor ID (U16) */
   ProcId      srcProcId;      /* Source processor ID (U16) */
   Ent         dstEnt;         /* Destination entity (U8)  */
   Inst        dstInst;        /* Destination Instance (U8) */
   Ent         srcEnt;         /* Source entity (U8) */
   Inst        srcInst;        /* Source Instance (U8) */
   Prior       prior;          /* Priority (U8) */
   Route       route;          /* Route (U8) */
   Event       event;          /* Event (U8) */
   Region      region;         /* Region (U8) */
   Pool        pool;           /* Pool (U8) */
   Selector    selector;       /* Selector (U8) */
   CmIntfVer   intfVer;        /* Interface version (U16) */
} Pst;
```

The values used for the individual fields of the Post Structure at the management interface are described in Table 2-5. For primitives initiated by the stack manager to MME Application, the Post Structure is initialized as shown below:

Table 2-5: Pst Structure fields

Parameter	Description and Allowable values
dstProcId	Processor ID of MME Application. Allowable values: 0 to 255.
dstEnt	Entity ID of MME Application. Allowable value: ENTVB
dstInst	Instance ID of MME Application. Allowable values: 0 to 255
srcProcId	Processor ID of Stack manager. Allowable values: 0 to 255

Table 2-5: Pst Structure fields

Parameter	Description and Allowable values
srcEnt	Entity ID of Stack manager. Allowable value: ENTSM
srcInst	Instance ID of Stack manager. Allowable values: 0 to 255
prior	Priority value for the specific event.
route	Route information, if any.
event	The specific management request event.
region	Memory region information.
pool	Memory pool information.
selector	Specifies whether the stack manager is loosely coupled or tightly coupled with MME Application. This field is used to decide how the primitive is routed from the layer manager to the MME layer.

2.3 Primitives and Procedures

The following rules apply to each flow diagram in this section.

1. Time flows toward the bottom of the page.
2. The mnemonic above a line represents a function call or MME Application primitive.
3. The mnemonic below a line represents a MME Application message type.
4. A **+** indicates an OR condition (one path or another can be taken).
5. A o indicates an AND condition (both paths are taken in parallel).

   The following table define the abbreviations above the flow diagrams.

Table 2-6: Abbreviations used in flow diagrams

Abbreviation	Definition
ss	System services
sz	S1AP Layer
eg	eGTP Layer
vb	MME Application
lm	Layer manager

The MME Application-specific procedures are described in this document.

Note: In the primitives listed below, the type refers to whether these parameters are:

- **Mandatory** (customer needs to fill in the value) denoted by **M**.
- **Optional** (customer may or may not fill in the value) denoted by **O.**
- **Not Required** (the value is filled in by the layer below) denoted by **NR.**

2.3.1 Configuration Request

Name	Configuration request. **SmMiLvbCfgReq** is invoked in the layer manager. **VbMiLvbCfgReq** processes this request in MME Application.
Direction	Layer manager to MME Application.
Response	Configuration Confirm. **VbMiLvbCfgCfm** is invoked in the MME. **SmMiLvbCfgCfm** processes this request in layer manager.
Location	**vbsm_vbcfg.c** and **vb_mi.c**

Primitives

VbMiLvbCfgReq

SmMiLvbCfgReq

Synopsis

```
PUBLIC S16 VbMiLvbCfgReq (pst, cfg)
Pst        *pst;
VbMngmt  *cfg;
```

Parameters

pst

Pointer to the post structure. For the configuration request, the event field in the **Pst** structure is set to **EVTLVBCFGREQ**. Refer to Section 2.2.2.

cfg

This parameter represents the management structure described in Section 2.2.1, "VbMngmt." The parameters specific configuration request are described. This parameter is the pointer to the configuration structure. The configuration structure has the following format:

2.3.1.1 Configuration Data Structure

```
typedef struct VbMngmt
{
    Header         hdr;      /* Header */
    CmStatus       cfm;      /* Result of  operation */
    union
    {
        VbCfg      cfg;      /* Configuration */
        ....
        ....
        }u;
} VbMngmt;

typedef struct vbCfg
{
    union
    {
        VbMmeGenCfg   mmeGenCfg;   /* MME General config   */
        VbMmeLSapCfg  mmeLSapCfg;  /* MME Lower SAP config */
        }s;
} VbCfg;
```

Paramter	Description	Reference
mmeGenCfg	MME general configuration	Section 2.3.1.1.1
mmeLSapCfg	MME Lower SAP configuration	Section 2.3.1.1.2

2.3.1.1.1 MME General Configuration

This structure provides the general configuration information for the layer. In the general configuration request, the **ElmntId** field of the Header structure must be set to STGEN:

```
hdr.elmId.elmnt = STGEN.

typedef struct _vbMmeGenCfg
{
    Pst         lmPst;         /* Post structure for layer manager */
    Mem         mem;           /* Memory pool and region */
    U16        maxNmbSztSaps;/* Maximum Number of Lower (SZT) SAPs */
    U16        maxNmbEgtSaps;/* Maximum Number of Lower (EGT) SAPs */
    U16         maxNmbEnbs;    /* Maximum number of Enodebs */
    U16         maxNmbUes;     /* Maximum number of Ues */
    U16         tmrRes;        /* Timer resolution */
    CmTptAddr   selfAddr;      /* Transport Address of the self */
    CmTptAddr   sgwAddr;       /* Transport Address of the SGW Addr */
    CmTptAddr   pgwAddr;       /* Transport Address of the PGW Addr */
    VbSrvdGLst srvGLst;        /* Served GUMMEI list */
    VbMmeName mmeName;         /* Name of MME */
    VbTaiLst  tLst;            /* Tracking Area Info to send to UE  */
    VbEnbCbCfg eNbCfg[LVB_MAX_ENBS];/* Cfg for set of eNodeb's  */
} VbMmeGenCfg;
```

The parameters are:

Field	Description
lmPst	This is the post structure that the MME uses to communicate with the stack manager. In this structure, the values of the following parameters must be set as indicated: **dstEnt**: Entity ID of the stack manager -ENTSM. **dstInst**: Instance ID of the stack manager. **dstProcId**: Processor ID of the stack manager. **prior**: Priority with which MME must send any messages to the stack manager. **route**: Any routing information provided to system services, if the stack manager and MME are loosely coupled. **selector**: The coupling used when messages are sent from MME to the stack manager. **region** and **pool**: Memory region and memory pool from which message buffers are allocated for any messages sent from MME to the stack manager. **srcEnt**, **srcInst**, and **srcProcId** must not be filled. MME receives this information at initialization time from system services and fills in the appropriate values.

Field	Description
mem	Memory pool and region.
maxNmbSztSaps	Maximum number of Lower (SZT) SAPs.
maxNmbEgtSaps	Maximum Number of Lower (EGT) SAPs.
maxNmbEnbs	Maximum number of EnodeBs
maxNmbUes	Maximum number of UEs
tmrRes	Timer resolution
selfAddr	Transport Address of the self
sgwAddr	Transport Address of the SGW Address
pgwAddr	Transport Address of the PGW Address
srvGLst	Served GUMMEI List
mmeName	Name of MME
tLst	Tracking Area Info to send to UE
eNbCfg	Configuration for set of eNodeB's

2.3.1.1.2 MME Lower SAP Configuration

The MME lower SAP configuration structure is given below.

```
typedef struct vbMmeLSapCfg
{
    SuId              suId;           /* S1AP SAP ID */
    SpId              spId;           /* Service Provider SAP ID */
    Selector          selector;       /* Coupling */
    MemoryId          mem;            /* Memory pool */
    Priority          prior;          /* Priority */
    Route             route;          /* Route */
    ProcId            dstProcId;      /* Destination processor ID */
    Ent               dstEntId;       /* Destination entity ID */
    Inst              dstInstId;      /* Destination instance ID */
    U8                maxBndRetry;    /* Maximum number of bind
                                         retries allowed */
    TmrCfg           tBndTmr;        /* Bind timer for application */
    U8               type;           /* SAP Type */
VbMmeLSapCfg;
```

The parameters are:

Field	Description
suId	S1AP SAP ID or EGTP SAP ID
spId	Service provider SAP ID
selector	Lower interface selector
mem	Lower interface Memory pool and region
prior	Lower interface priority.
route	Lower interface route
dstPorcId	Destination processor ID
dstEntId	Destination entity ID
dstInstId	Destination instance ID
maxBndRetry	Maximum number of bind retries allowed
tBndTmr	Bind timer
type	SAP type

2.3.1.1.3 Configuration Procedure Callflow

The layer manager configures the various elements of MME Application using the management – configuration procedure, which the layer manager initiates. The MME Application configuration request primitive (**VbMiLvbCfgReq**) can be called more than once. **VbMiLvbCfgReq** primitives must be called before the bind primitives are called.

The following table lists the MME Application configuration request primitive types.

Name	Description
General	Passes parameters that apply to the entire MME Application software. It reserves the memory pool for its static memory requirements and register for timer service with SSI. It can be called once.
Lower SAP S1AP	MME Application lower SAP configuration. The SAP is used to communicate with the provider S1AP at the SZT interface.
Lower SAP eGTP	MME Application lower SAP configuration. The SAP is used to communicate with the provider eGTP at the EGT interface.

To operate properly, the configuration request primitive types must be called in the following order:

1. General
2. Lower SAP S1AP
3. Lower SAP eGTP

The system services primitives are called during the management – configuration procedure

.

Note: The register timer (**SRegTmr**) system services primitive is called during the general configuration request procedure to register the MME Application timer activation (**vbActvTmr**) function.

The **vbMngmt.t.cfg** structure specifies the parameters that the configuration request primitive (**VbMiLvbCfgReq**) uses.

Figure 2-1 shows the MME Application configuration data flow.

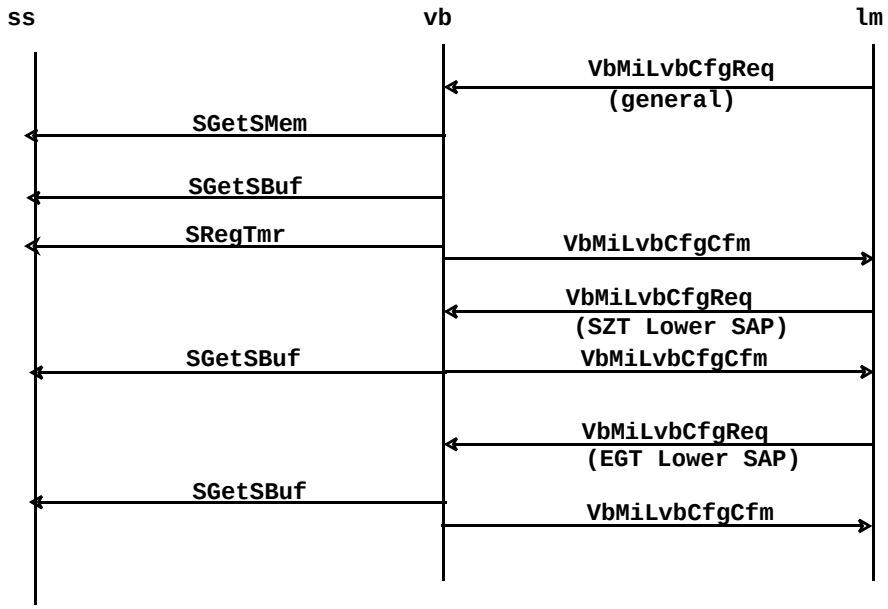


Figure 2-1: MME Application configuration data flow

2.3.2 Configuration Confirm

Name	Configuration confirm. **VbMiLvbCfgCfm** is invoked in MME Application. **SmMiLvbCfgCfm** processes this confirm primitive in the layer manager.
Direction	MME Application to Layer Manager
Response	Not applicable
Location	**vb_mi.c and vbsm_vbcfg.c**

Primitives

VbMiLvbCfgCfm

SmMiLvbCfgCfm

Synopsis

```
PUBLIC S16 VbMiLvbCfgCfm(pst, cfm)
Pst        *pst;
VbMngmt  *cfm;
```

Parameters

pst

Pointer to the post structure. For the configuration request, the event field in the Post structure is set to **EVTLVBCFGCFM**. Refer to Section 2.2.2

cfm

This parameter represents a pointer to the management structure described in Section 2.2.1. The configuration confirm specific parameters are described in the following sections.

```
typedef struct VbMngmt
{
    Header hdr;/* header */
    CmStatus cfm;/* result of  operation */
    union
    {
        ....
        ....
        ....
    }u;
} VbMngmt;
```

In the **Header** of the configuration confirm, the values of the relevant fields are set, as shown in the following table.

Field	Allowable Values
hdr.elmId.elmnt	Copied from the received configuration request primitive. Allowable values: • **STVBGEN**: For general configuration. • **STVBSZTSAP**: For S1AP SAP configuration. • **STVBEGTSAP**: For eGTP SAP configuration.
hdr.transId	Copied from the received configuration request to enable correlation of requests in the stack manager.

The status information is returned in the **CmStatus** data structure. For the configuration confirm primitive, the values of the **CmStatus** are:

status

Name	Description
LCM_PRIM_OK	Configuration request is processed to successful completion.
LCM_PRIM_NOK	The processing of the configuration request is aborted owing to an error. The error is qualified in the reason field.
LCM_PRIM_OK_NDONE	Configuration request is processed is not completed.

reason

The content of this field is set to **LCM_REASON_NOT_APPL** when **status** is set to **LCM_PRIM_OK**. When the **status** field is set to **LCM_PRIM_NOK**, the content of the reason field can be set to any of the following values, which indicates the reason for the failure of the primitive. The following table lists the values valid for general and Upper SAP configuration.

Value	Description
LCM_REASON_NOT_APPL	Reason not applicable. It is used with **LCM_PRIM_OK**.
LCM_REASON_INVALID_ENTITY	Invalid entity specified in MME entity configuration request.
LCM_REASON_INVALID_INSTANCE	Invalid instance specified in MME instance configuration request.
LCM_REASON_INVALID_MSGTYPE	Invalid message type.
LCM_REASON_MEM_NOAVAIL	Memory allocation failed. Either memory cannot be reserved using **SGetSMem**, or memory cannot be allocated to the required static structures using **SGetSBuf**.
LCM_REASON_INVALID_ELMNT	Value of the **hdr.elmId.elmnt** is invalid.
LCM_REASON_RECONFIG_FAIL	Reconfiguration on the specific element failed.
LCM_REASON_REGTMR_FAIL	Timer registration (**SRegTmr**) failed.
LCM_REASON_GENCFG_NOT_DONE	General configuration is not done.
LCM_REASON_INVALID_ACTION	Invalid action.
LCM_REASON_INVALID_SUBACTION	Invalid sub action.
LCM_REASON_INVALID_STATE	Invalid state.
LCM_REASON_INVALID_SAP	Invalid SAP ID in the SAP configuration request.
LCM_REASON_INVALID_PAR_VAL	Invalid parameter value.
LCM_REASON_QINIT_FAIL	Queue initialization failure.
LCM_REASON_NEG_CFM	Negative confirmation.
LCM_REASON_UPDTMR_EXPIRED	Update timer expired.
LCM_REASON_MISC_FAILURE	Miscellaneous failures.
LCM_REASON_EXCEED_CONF_VAL	Exceeds configured value.
LCM_REASON_HASHING_FAILED	Hashing failed.
LCM_REASON_PEERCFG_NOT_DONE	Peer configuration is not done.

2.3.3 Unsolicited Status Indication

Name	Unsolicited Status Indication. **VbMiLvbStaInd** is invoked in MME Application, and **SmMiLvbStaInd** processes confirm primitive in the layer manager.
Direction	MME Application to Layer Manager
Response	Not applicable
Location	**vb_mi.c** and **vbsm_vbcfg.c**

Primitives

VbMiLvbStaInd

SmMiLvbStaInd

Synopsis

```
PUBLIC S16 VbMiLvbStaInd(pst, sta)
Pst       *pst;
VbMngmt *usta;
```

Parameters

pst

Pointer to the post structure. For the configuration request, the event field in the **Pst** structure is set to EVTLVBUSTAIND. Refer to Section 2.2.2

usta

Unsolicited status indication. It is a pointer to the management structure. It has the following format.

```
typedef struct VbMngmt
{
    Header hdr;/* header*/
    CmStatus cfm;/* result of  operation */
    union
    {
        ....
        ....
        VbUstaDgn usta;/* Alarm */
    }u;
} VbMngmt;

typedef struct _vbUstaDgn
```

```
{
    CmAlarm       alarm;
} VbUstaDgn;
```

hdr

Header structure.

cfm

Common status structure. It is not used for this primitive.

alarm

Alarm category and event. This structure provides information about the date and time, category, event, and cause for an alarm generation.

```
typedef struct cmAlarm
{
    DateTime     dt;           /* Date and time */
    U16          category;     /* Category of the alarm */
    U16          event;        /* Event */
    U16          cause;        /* Cause for event */
} CmAlarm;
```

Primitive	Description
dt	Date and time.
category	Specifies the category of the alarm.

The allowable values of **category** are:

Allowable Values	Description
LCM_CATEGORY_INTERFACE	Interface-related alarm.
LCM_CATEGORY_RESOURCE	System resource related alarm.
LCM_CATEGORY_PROTOCOL	Protocol-related alarm.
LCM_CATEGORY_INTERNAL	Internal state-related alarm.

evnt

This parameter specifies the event that caused the generation of a status indication to the layer manager, from the MME software. Event codes are not unique and must be interpreted in conjunction with the category of the generated alarm. For each alarm category, the allowable values are:

Allowable Values	Description
LVB_EVENT_MME_S1AP_BNDCFM	S1AP bind successful
LVB_EVENT_MME_EGTP_BNDCFM	EGTP bind successful

cause

This parameter specifies the cause for the alarm. The allowable values are:

Allowable Values	Description
LVB_CAUSE_BND_SUCC	Bind successful
LVB_CAUSE_ARP_SUCC	ARP successful

Description

The management – unsolicited status procedure provides unsolicited status information about MME Application elements to the layer manager. MME Application initiates this procedure. The MME Application status indication primitive (**VbMiLvbStaInd**) can be called more than once and, if the unsolicited status is enabled, at any time after the configuration procedure. The MME Application status indication primitive is not called if the unsolicited status is disabled. The unsolicited status can be enabled or disabled with the management – control procedure.

The **vbMngmt.t.usta** structure specifies parameters that are used by the status indication (**VbMiLvbStaInd**) primitive.

Figure 2-2 shows the management – unsolicited status procedure.

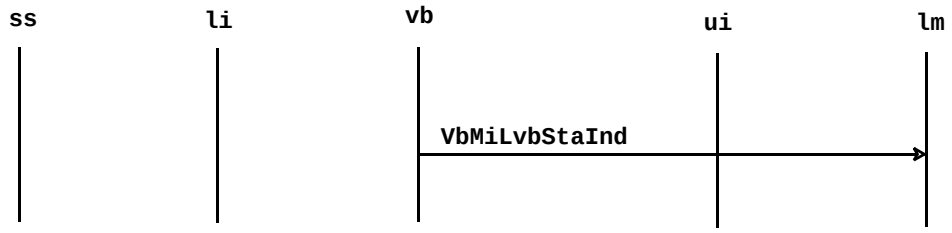


Figure 2-2: Management – Unsolicited status procedure

Note: The MME Application uses this function to alert the layer manager to a significant change in the software state, or when an error is encountered while processing a primitive.

2.3.4 Control Request

Name	**VbMiLvbCntrlReq**
Direction	Layer Manager to MME Application.
Response	Control Confirm. **VbMiLvbCntrlCfm** is invoked in the MME. **SmMiLvbCntrlCfm** processes this request in layer manager.
Location	**vb_mi.c** and **vbsm_vbcfg.c**

Primitives

VbMiLvbCntrlReq

SmMiLvbCntrlReq

Synopsis

```
PUBLIC S16 VbMiLvbCntrlReq(pst, cntrl)
Pst       *pst;
VbMngmt   *cntrl;
```

Parameters

pst

Pointer to the post structure. For the configuration request, the event field in the **Pst** structure is set to **EVTLVBCNTRLREQ**. Refer to Section 2.2.2

cntrl

Pointer to the control structure. Control structure has the following format:

```
typedef struct VbMngmt
{
    Header          hdr;   /* Header */
    CmStatus        cfm;   /* Result of  operation */

    union
    {
        ....
        VbCntrl    cntrl;  /* Control */
        ....
    }u;
} VbMngmt;
    typedef struct vbCntrl
    {
        VbMmeCntrl     mmeCntrl;
} VbCntrl;
    typedef struct vbMmeCntrl
    {
        DateTime      dt;    /**< Date */
        U8 action;            /**< Action to be performed  Bind,
        Unbind        or shutdown*/
        U8  subAction;   /**< subaction to be performed, SADBGB
                              and SAUSTA */
        union
        {
#ifdef DEBUGP
        VbDbgCntrl   dbg;
#endif /* DEBUGP */
        VbSapCntrl        sap;     /**< The layer manager can
                                        selectively    bind/unbind
                                        the different SAPs. */

        }u;
}VbMmeCntrl;
typedef struct vbDbgCntrl
    {
        U32 dbgMask;
}
typdef struct vbSapCntrl
{
        SpId  id;
```

}

The parameters are:

Parameter	Description
dt	This field is not used in this primitive. It is used in the **VbYyLvbCntrlCfm** primitive.
action	This field specifies the action that the MME Application layer must take.
subAction	This field specifies the protocol element on which the MME Application layer takes the specified action.
sap	Contains the information required for SAP control.
dbg **dbgMask**	The layer manager can selectively enable/disable various levels of debug printing. The allowable values of the **dbgMask** are: **LVB_DBGMASK_MEM**- Memory debug mask **LVB_DBGMASK_TRC** - Trace debug mask **LVB_DBGMASK_ERROR**- Error debug mask **LVB_DBGMASK_INFO** - Information debug mask **dbgMask** can be set to a combination of the aforementioned allowable values. For example, if the debugging prints at the upper and lower interfaces of the layer are to be enabled/disabled, the **dbgMask** is set to **DBGMASK_UI** | **DBGMASK_LI**.

Table 2-7: Allowable Action Values of the MME Application Layer

action	subaction	elmnt
AENA	**SADBG**	**STVBGEN**
ADISIMM	**SAUSTA**	**STVBSZTSAP**
ASHUTDOWN	**--**	**--**
ADEL	**--**	**--**
ABND	**--**	**--**
AUBND	**--**	**--**

Description

The layer manager uses the management – control procedure to control MME Application elements. The layer manager initiates this procedure. The MME Application control request primitive (**VbMiLvbCntrlReq**) can be called more than once and at any time after the management – configuration procedure.

The following MME Application control request primitive types can be called:

The **vbMngmt.t.cntrl.action** field specifies the control request primitive type.

The **vbMngmt.t.cntrl.subaction** field specifies the element to be controlled.

The **vbMngmt.t.cntrl** structure specifies the parameters that are used by the control request (**VbMiLvbCntrlReq**) primitive.

Figure 2-3 shows the management – control request procedure.

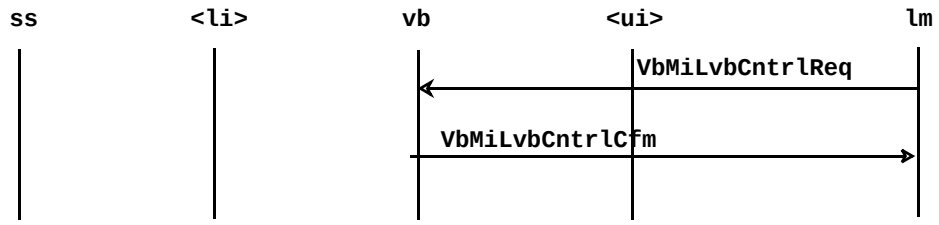


Figure 2-3: Management – Control request procedure

2.3.5 Control Confirm

Name	Control Confirm. **VbMiLvbCntrlCfm** is invoked in MME Application, and **SmMiLvbCntrlCfm** processes this confirm primitive in the layer manager.
Direction	MME Application to Layer Manager
Response	Not applicable
Location	**vb_mi.c** and **vbsm_vbcfg.c**

Primitives

VbMiLvbCntrlCfm

SmMiLvbCntrlCfm

Synopsis

```
PUBLIC S16 VbMiLvbCntrlCfm(pst, cfm)
Pst        *pst;
VbMngmt  *cfm;
```

Parameters

pst

Pointer to the post structure. For the configuration request, the event field in the **Pst** structure is set to **EVTLVBCNTRLCFM**. Refer to Section 2.2.2

cfm

This parameter represents a pointer to the management structure described in Section 2.2.1. The control confirm specific parameters are described in the following sections.

```
typedef struct VbMngmt
{
   Header        hdr;        /* Header*/
   CmStatus      cfm;        /* Result of  operation*/
   union
   {
       ....
       ....
       ....
   }u;
} VbMngmt;
```

In the **Header** parameter of the configuration confirm primitive, the values of the relevant fields are set as follows:

Field	Allowable Values
hdr.elmId.elmn	Copied from the received configuration request primitive. Allowable values: **STVBGEN**: For general configuration. **STVBSZTSAP**: For S1AP SAP configuration. **STVBEGTSAP**: For eGTP SAP configuration.
hdr.tranId	Copied from the received control request to enable correlation of requests in the stack manager.

The status information is returned in the **CmStatus** data structure. For the control confirm primitive, the values of the **CmStatus** are set as follows:

status

Name	Description
LCM_PRIM_OK	Control request is processed to successful completion.
LCM_PRIM_OK_NDONE	The processing of the control request is deferred. This value is sent only if MME cannot immediately service the control request.
LCM_PRIM_NOK	The processing of the control request is aborted owing to an error. The error is qualified in the reason field.

reason

The contents of this field are set to **LCM_REASON_NOT_APPL,** if the **status** field is set to **LCM_PRIM_OK**.

When the **status** field is set to the value **LCM_PRIM_NOK**, the contents of the reason field can be set to any of the following values, which indicates the reason for the failure of the primitive.

The following values are valid for all control operations:

Value	Description
LCM_REASON_NOT_APPL	Reason not applicable. It is used with **LCM_PRIM_OK**.
LCM_REASON_INVALID_ENTITY	Invalid entity specified in MME entity configuration request.
LCM_REASON_INVALID_INSTANCE	Invalid instance specified in MME instance configuration request.
LCM_REASON_INVALID_MSGTYPE	Invalid message type
LCM_REASON_MEM_NOAVAIL	Memory allocation failed. Either memory cannot be reserved using **SGetSMem**, or memory cannot be allocated to the required static structures using **SGetSBuf**.
LCM_REASON_INVALID_ELMNT	Value of the **hdr.elmId.elmnt** is invalid.
LCM_REASON_RECONFIG_FAIL	Reconfiguration on the specific element failed.
LCM_REASON_REGTMR_FAIL	Timer registration (**SRegTmr**) failed.
LCM_REASON_GENCFG_NOT_DONE	General configuration is not done
LCM_REASON_INVALID_ACTION	Invalid action
LCM_REASON_INVALID_SUBACTION	Invalid sub action
LCM_REASON_INVALID_STATE	Invalid state
LCM_REASON_INVALID_SAP	Invalid SAP ID in the SAP configuration request.
LCM_REASON_INVALID_PAR_VAL	Invalid parameter value
LCM_REASON_QINIT_FAIL	Queue initialization failure
LCM_REASON_NEG_CFM	Negative confirmation
LCM_REASON_UPDTMR_EXPIRED	Update timer expired
LCM_REASON_MISC_FAILURE	Miscellaneous failures
LCM_REASON_EXCEED_CONF_VAL	Exceeds configured value
LCM_REASON_HASHING_FAILED	Hashing failed
LCM_REASON_PEERCFG_NOT_DONE	Peer configuration is not done

3

MME Application Initialization and Shutdown

The interface procedures define the mechanisms by which MME Application software interacts, through primitives, with any adjacent software.

The procedures differ depending on whether a tightly or loosely coupled interface is used. A tightly coupled interface implies that the interface consists of direct function calls between the two layers. A loosely coupled interface implies that the interface consists of passing messages between the two layers through system services. Refer to the *System Services Interface Service Definition* for more details about interface coupling. The following description assumes a tightly coupled interface.

Note: Procedures specific to MME Application Layer Manager Interface are described in this document.

3.1 MME Application Initialization

For initialization, the interface procedures must be performed in the following order:

1. Initialization of MME Application

   Initializes the MME Application software. The layer manager begins the procedure when it registers the initialization function for MME Application using the **`SRegTTsk`** primitive. System services then calls the initialization function **`vbActvInit`**. The initialization function must be called once before any other primitives or functions in MME Application are called. It creates a system task by calling the **`SCreateSTsk`** primitive on SSI. Now, the TAPA task must be attached to this system task to make it scheduled by SSI.

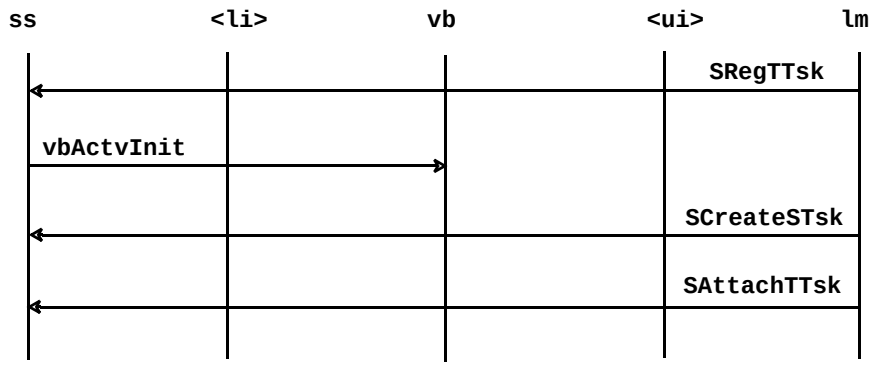


Figure 3-1: MME Application Initialization

2. Management – Configuration

   Configures the MME Application software. For the configuration sequence, refer to Figure 2-1.

3. Lower layer – Bind

   MME Application software binds itself to its lower layers S1AP and eGTP. The layer manager initiates the bind with the lower layer by sending the control request to the MME Application with the **`action`** set to "Bind and Enable". The MME Application then sends the bind request to the lower layer. After receiving the bind confirm from the lower layer, MME Application sends an alarm to the layer manager, if alarms are enabled.

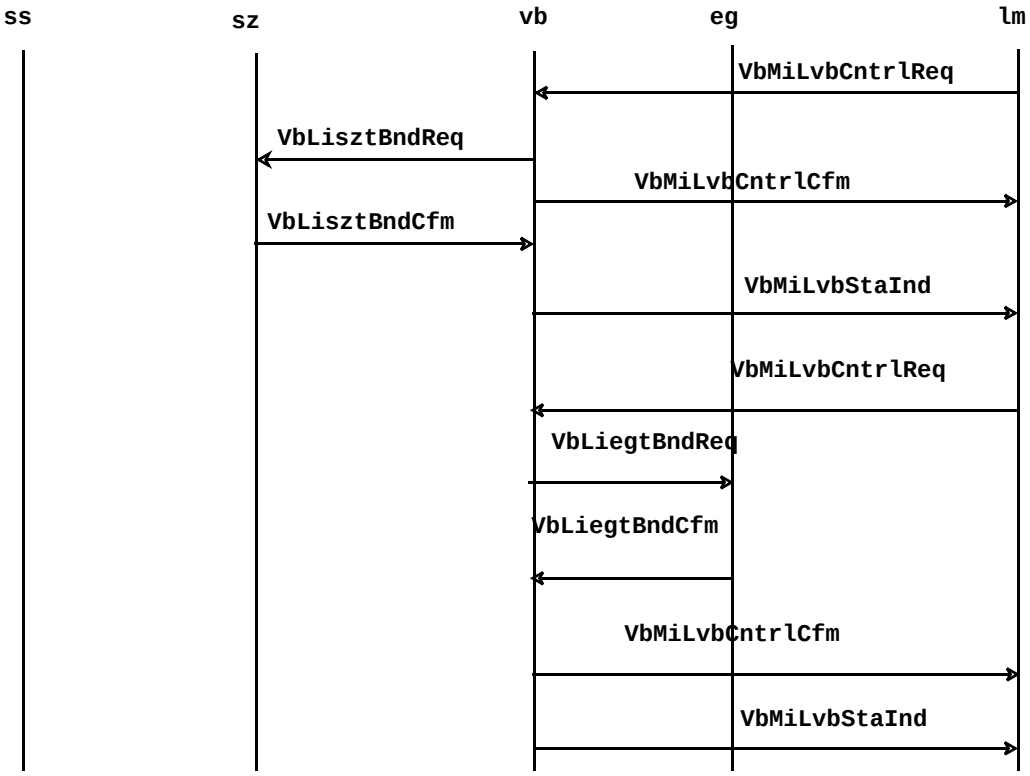


Figure 3-2: Lower Layer Bind

3.2 MME Application Shutdown

For shutdown, the interface procedures must be performed in the following order:

1. Service provider SAP unbind

   MME Application software unbinds itself from its layers S1AP and eGTP. The layer manager initiates the unbind with the lower layer by sending the control request to the MME Application with the **`action`** set to "Unbind and Disable". The MME Application then sends the unbind request to the lower layer after clearing the resources associated with this SAP.

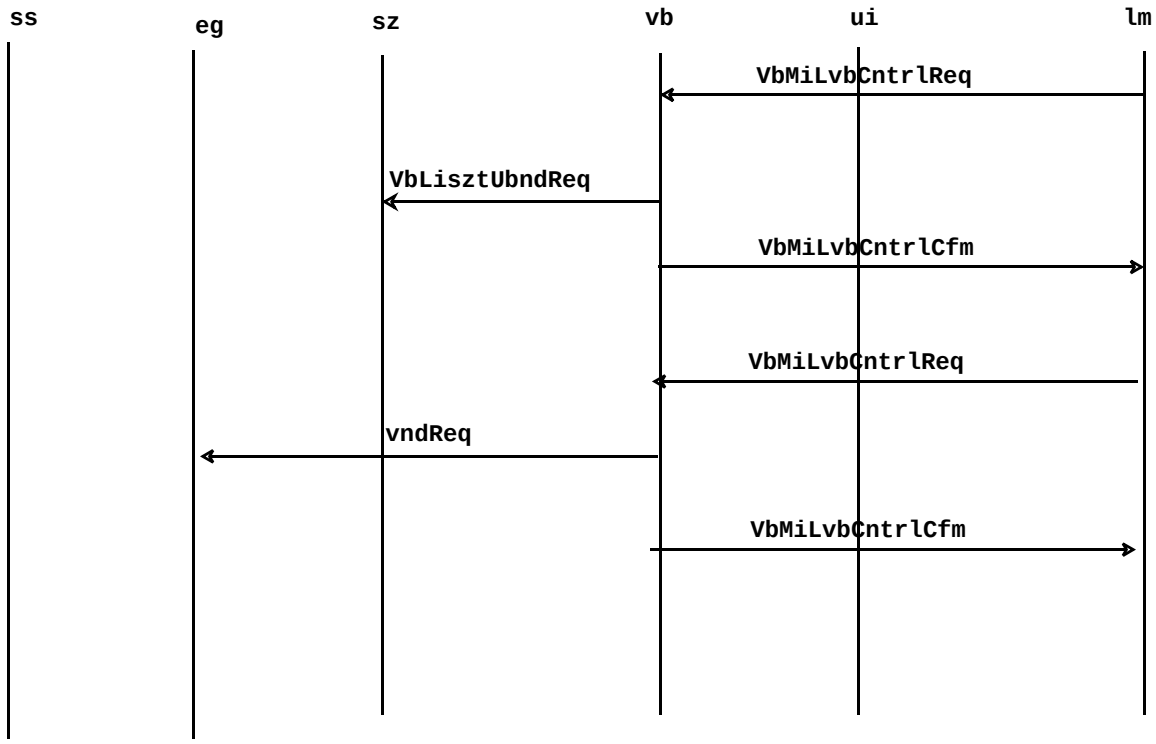


Figure 3-3: Service Provider SAP Unbind

2. Shutdown Request

   The layer manager initiates the shutdown request to the MME Application by sending the control request with the **`action`** set to "Shutdown". This operation brings back the MME Application to its pre-configured state.

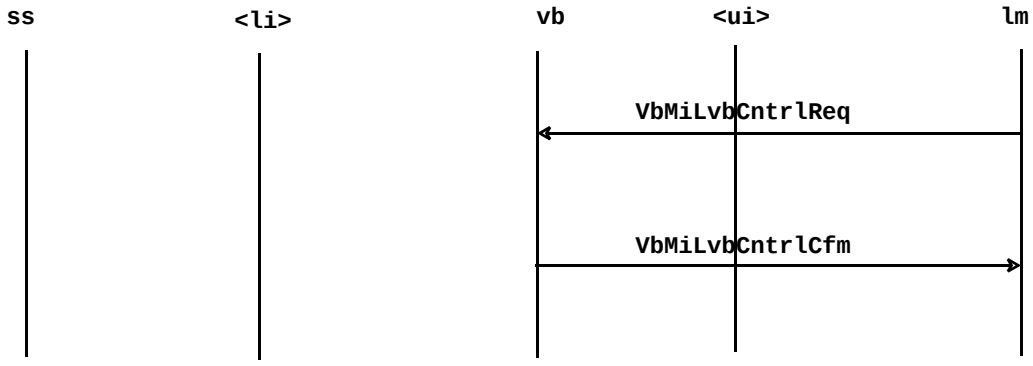


Figure 3-4: Shutdown Request

Now, the layer is deactivated and non operational.

References

Refer to the following documents for more information:

LTE MME Reference Application Functional Specification, Continuous Computing Corporation (p/n 1091417)

SZT Interface Service Definition, Continuous Computing Corporation (p/n 1100361).

EGT Interface Service Definition, Continuous Computing Corporation (p/n 1100368).

TAPA Concepts and Guidelines Common Document, Continuous Computing Corporation (p/n 1111011).

System Services Interface Service Definition, Continuous Computing Corporation (p/n 1111001).

3GPP TS 24.301 V8.2.1 (2009-06): Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS)

3G TS 36.413, version 8.1.0 (2008-03), E-UTRAN S1 Application Protocol.

3G TS 36.413, version 8.2.0 (2008-06), E-UTRAN S1 Application Protocol.

3G TS 29.274, version 1.3.0 (2008-10), Evolved GPRS Tunnelling Protocol for Control Plane.

3G TS 29.281, version 8.1.0 (2008-10), GPRS Tunnelling Protocol -User Plane.

Index

R

S

T

U