Mobile Application Part Interface (MAPI) Specification

Mobile Application Part Interface (MAPI) Specification

Version 1.1 Edition 7.20141001
Updated October 25, 2014
Distributed with Package openss7-1.1.7.20141001

Abstract:

This document is a Specification containing technical details concerning the implementation of the Mobile Application Part Interface (MAPI) for OpenSS7. It contains recommendations on software architecture as well as platform and system applicability of the Mobile Application Part Interface (MAPI). It provides abstraction of the Mobile Application Part (MAP) interface to these components as well as providing a basis for Mobile Application Part control for other Mobile Application Part protocols.

Brian Bidulock <bidulock@openss7.org> for
The OpenSS7 Project <http://www.openss7.org/>

Published by:

OpenSS7 Corporation
1469 Jefferys Crescent
Edmonton, Alberta T6L 6T1
Canada

Notice:

OpenSS7 Corporation disclaims all warranties with regard to this documentation including all implied warranties of merchantability, fitness for a particular purpose, non-infringement, or title; that the contents of the document are suitable for any purpose, or that the implementation of such contents will not infringe on any third party patents, copyrights, trademarks or other rights. In no event shall OpenSS7 Corporation be liable for any direct, indirect, special or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with any use of this document or the performance or implementation of the contents thereof.

Short Contents

Table of Contents

List of Figures

List of Tables

Preface

Notice

Abstract

This document is a Specification containing technical details concerning the implementation of the Mobile Application Part Interface (MAPI) for OpenSS7. It contains recommendations on software architecture as well as platform and system applicability of the Mobile Application Part Interface (MAPI).

This document specifies a Mobile Application Part Interface (MAPI) Specification in support of the OpenSS7 Mobile Application Part (MAP) protocol stacks. It provides abstraction of the Mobile Application Part interface to these components as well as providing a basis for Mobile Application Part control for other Mobile Application Part protocols.

Purpose

The purpose of this document is to provide technical documentation of the Mobile Application Part Interface (MAPI). This document is intended to be included with the OpenSS7 STREAMS software package released by *OpenSS7 Corporation*. It is intended to assist software developers, maintainers and users of the Mobile Application Part Interface (MAPI) with understanding the software architecture and technical interfaces that are made available in the software package.

Intent

It is the intent of this document that it act as the primary source of information concerning the Mobile Application Part Interface (MAPI). This document is intended to provide information for writers of OpenSS7 Mobile Application Part Interface (MAPI) applications as well as writers of OpenSS7 Mobile Application Part Interface (MAPI) Users.

Audience

The audience for this document is software developers, maintainers and users and integrators of the Mobile Application Part Interface (MAPI). The target audience is developers and users of the OpenSS7 SS7 stack.

Revision History

Take care that you are working with a current version of this documentation: you will not be notified of updates. To ensure that you are working with a current version, check the OpenSS7 Project website for a current version.

A current version of this specification is normally distributed with the *OpenSS7* package, `openss7-1.1.7.20141001`.[1]

[1] `http://www.openss7.org/repos/tarballs/openss7-1.1.7.20141001.tar.bz2`

Version Control

Although the author has attempted to ensure that the information in this document is complete and correct, neither the Author nor OpenSS7 Corporation will take any responsibility in it. *OpenSS7 Corporation* is making this documentation available as a reference point for the industry. While *OpenSS7 Corporation* believes that these interfaces are well defined in this release of the document, minor changes may be made prior to products conforming to the interfaces being made available. *OpenSS7 Corporation* reserves the right to revise this software and documentation for any reason, including but not limited to, conformity with standards promulgated by various agencies, utilization of advances in the state of the technical arts, or the reflection of changes in the design of any techniques, or procedures embodied, described, or referred to herein. *OpenSS7 Corporation* is under no obligation to provide any feature listed herein.

```
$Log: mapi.texi,v $
Revision 1.1.2.2  2011-02-07 02:21:40  brian
- updated manuals

Revision 1.1.2.1  2009-06-21 10:53:58  brian
- added files to new distro
```

ISO 9000 Compliance

Only the TEX, texinfo, or roff source for this maual is controlled. An opaque (printed, postscript or portable document format) version of this manual is a **UNCONTROLLED VERSION**.

Disclaimer

OpenSS7 Corporation **disclaims all warranties with regard to this documentation including all implied warranties of merchantability, fitness for a particular purpose, non-infrincement, or title; that the contents of the manual are suitable for any purpose, or that the implementation of such contents will not infringe on any third party patents, copyrights, trademarks or other rights. In no event shall** *OpenSS7 Corporation* **be liable for any direct, indirect, special or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action or contract, negligence or other tortious action, arising out of or in connection with any use of this documentation or the performance or implementation of the contents thereof.**

U.S. Government Restricted Rights

If you are licensing this Software on behalf of the U.S. Government ("Government"), the following provisions apply to you. If the Software is supplied by the Department of Defense ("DoD"), it is classified as "Commercial Computer Software" under paragraph 252.227-7014 of the DoD Supplement to the Federal Aquisition Regulations ("DFARS") (or any successor regulations) and the Government is acquiring only the license rights granded herein (the license rights customarily provided to non-Government users). If the Software is supplied to any unit or agency of the Government other than DoD, it is classified as "Restricted Computer Software" and the Government's rights in the Software are defined in paragraph 52.227-19 of the Federal Acquisition Regulations ("FAR") (or any successor regulations) or, in the cases of NASA, in paragraph 18.52.227-86 of the NASA Supplerment to the FAR (or any successor regulations).

Acknowledgements

The OpenSS7 Project was funded in part by:

- Monavacon Limited
- OpenSS7 Corporation

Thanks to the subscribers to and sponsors of The OpenSS7 Project. Without their support, open software like this would not be possible.

As with most open source projects, this project would not have been possible without the valiant efforts and productive software of the Free Software Foundation, the Linux Kernel Community, and the open source software movement at large.

1 Introduction

This document specifies a STREAMS-based kernel-level instantiation of the 3GPP GSM/UMTS TS.29002 Mobile Application Part (MAP) service definition. The Mobile Application Part Interface (MAPI) enables the user of a mobile application service to access and use a wide variety of conforming mobile application service providers without specific knowledge of the provider's protocol. The service interface is designed to support any operation class of transaction protocol. This interface only specifies access to mobile application layer service providers, and does not address issues concerning mobile application layer management, protocol performance, and performance analysis tools.

This specification assumes that the reader is familiar with the 3GPP GSM/UTMS reference points and protocol stacks, Mobile Application Part services, and STREAMS.

1.1 Related Documentation

- **3GPP TS 29.002 Mobile Application Part (MAP)**
- **3GPP TS 29.202**
- **1993 ITU-T Q.771 Recommendation**
- **System V Interface Definition, Issue 2 – Volume 3**
- **CCITT Recommendation X.200**
- **CCITT Recommendation X.201**
- System V Interface Definition, Issue 2 - Volume 3

1.1.1 Role

This document specifies an interface that supports the service provided by the Mobile Applications Part (MAP) specification for 3GPP GSM/UMTS as described in 3GPP TS 29.002. These specifications are targeted for use by developers and testers of protocol modules that require Mobile Applications Part services.

1.2 Definitions, Acronyms, Abbreviations

Application Context
Object Identifier
Calling Party
The Calling Party.

Called Party
The Called Party.

Operations Class
One of 5 ISO/OSI Transport Protocol Classes.

MAP Mobile Applications Part

TCAP Transaction Capabilities Application Part

SCCP Service Connection Control Part

MTP Message Transfer Part

TR Transaction Sub-Layer

TC	Component Sub-Layer
IMSI	International Mobile Station Identifier
MSISDN	Mobile Station ISDN Directory Number (E.164)
ITU	International Telecommunications Union
ITU-T	International Telecommunications Union – Telecom Sector
OSI	Open Systems Interconnect
ISO	International Organization for Standardization
MAP User	A user of the Mobile Application Part (MAP) Interface.
MAP Provider	A provider of the Mobile Application Part (MAP) Interface.
MAPI	The Mobile Application Part (MAP) Interface.
MS	Mobile Station.
Components	Transaction components as defined in ITU-T Recommendation Q.771.
QoS	Quality of Service
STREAMS	A communication services development facility first available with UNIX System V Release 3.

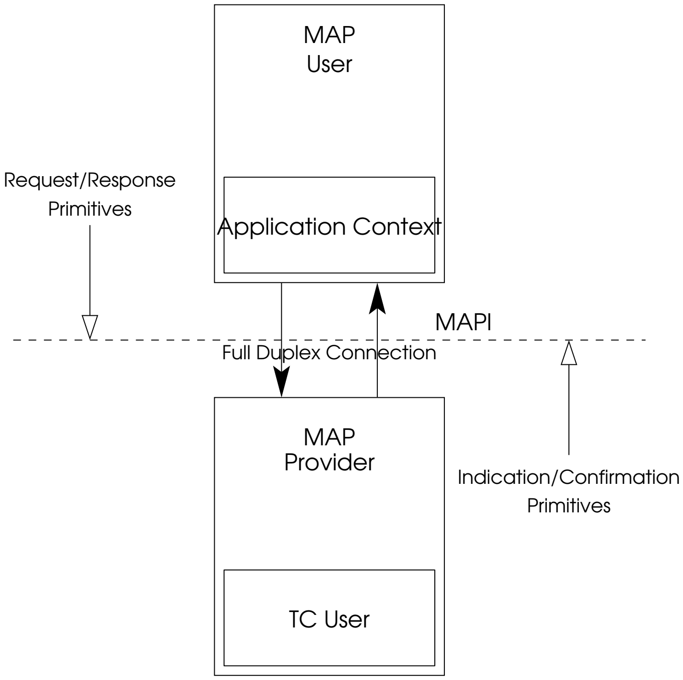
2 The Mobile Application Layer

The Mobile Application Layer provides the means to manage the operation of a mobile network. It is responsible for the routing and management of data exchange between the MAP-User entities.

2.1 Model of the MAPI

The MAPI defines the services provided by the MAP layer to the MAP-User at the boundary between the MAP layer and the MAP layer user entity. The interface consists of a set of primitives defined as STREAMS messages that provide access to the MAP layer services, and are transferred between the MAP user entity and the MAP provider. These primitives are of two types: ones that originate from the MAP user, and others that originate from the MAP provider. The primitives that originate from the MAP user make requests to the MAP provider, or respond to an event of the MAP provider. The primitives that originate from the MAP provider are either confirmation of a request or are indications to the MAP user that the event has occurred. *Figure 1* shows the model of the MAPI. (See Modelling Principles in 3GPP TS 29.002 V6.6.0.)

Figure 1 – *Module of the MAP Interface*



The MAPI allows the MAP provider to be configured with any MAP user (such as an HLR, MSC or SGSN application) that also conforms to the MAPI. A MAP user can also be a user program that conforms to the MAPI and accesses the MAP provider using '`putmsg()`' and '`getmsg()`' system calls.[1]

[1] See Section "putmsg(2s)" in *The UNIX Manual Pages*, also see Section "getmsg(2s)" in *The UNIX Manual Pages*.

2.2 MAP Services

The features of the MAP are defined in terms of the services provided by the MAP provider, and the individual primitives that may flow between the MAP users and the MAP provider.

The services provided by the MAP are based on a number of application contexts corresponding to the 3GPP TS 29.002 Mobile Application Part application contexts. In addition, the MAP supports services for local management.

- Local Management
- Common Services
- Mobility Services

2.2.1 Local Management

The MAP specifications also define a set of local management functions that apply to all application contexts. These services have local significance only.

Tables 1 and 2 summarizes the MAPI service primitives by their state and service.

2.2.2 Common Services

Table X. Common Services

Service	Invoker	Performer
MAP_OPEN		
MAP_DELIM		
MAP_CLOSE		
MAP_UABORT	User	User
MAP_PABORT	Provider	User
MAP_NOTICE		
MAP_STC1		
MAP_STC2		
MAP_STC3		
MAP_STC4		

2.2.3 Specific Services

Table 3. Specific Services

Service	Invoker	Performer
MAP_ACTSS	MSC	VLR
	VLR	HLR
MAP_ACTTM	HLR	VLR or SGSN
MAP_ALERTSC	HLR	SMS-IWMSC
MAP_ANYMOD	SCF	HLR
	PNA	VLR or SGSN
MAP_ANYSUB	SCF	HLR
MAP_AUTH	VLR	MSC
MAP_AUTHFAIL	VLR or SGSN	HLR
MAP_CANCLOC	HLR	VLR or SGSN
MAP_CKIEMI	VLR	MSC

	MSC or SGSN	EIR
MAP_DEACTSS	MSC	VLR
	VLR	HLR
MAP_DEACTTM	HLR	VLR or SGSN
MAP_DELSUBD	HLR	VLR or SGSN
MAP_ERASECC	MSC	VLR
	VLR	HLR
MAP_ERASESS	MSC	VLR
	VLR	HLR
MAP_FAILURE	GGSN	HLR
MAP_FAS	MSC-A	MSC-B
MAP_FCKSSIND	HLR	VLR
MAP_FGRCSIG	Anchor MSC	Relay MSC
MAP_MTFSMS	SMS-GMSC	MSC or SGSN
MAP_MOFSMS	MSC	SMS-IWMSC
MAP_GETPASS	HLR	VLR
	VLR	MSC
MAP_INFORMSC	HLR	SMS-GMSC
MAP_INSSUBSD	HLR	VLR or SGSN
MAP_INTERGSS	MSC	VLR
	VLR	HLR
MAP_ISTALERT	VMSC or GMSC	HLR
MAP_ISTCMD	HLR	VMSC or GMSC
MAP_MMEVENT	VLR or SGSN	SCF
MAP_MSGPRSPRES	HLR	GGSN
MAP_SUBSDATAMOD	HLR	SCF
MAP_PREPGC	Anchor MSC	Relay MSC
MAP_PREPHO	MSC-A	MSC-B
MAP_PREPSH	MSC-B	MSC-A
MAP_PROCAS	MSC	VLR
MAP_PROCGC	Relay MSC	Anchor MSC
MAP_PROCUSS	MSC	VLR
	VLR	HLR
MAP_PROVRN	HLR	VLR
MAP_PROVSLOC	GMLC	VMSC or SGSN
MAP_PROVSUBI	VLR	SGSN
MAP_PURGEMS	VLR or SGSN	HLR
MAP_RDYSM	MSC	VLR
	VLR or SGSN	HLR
MAP_REGCC	MSC	VLR
	VLR	HLR
MAP_REGPW	MSC	VLR
	VLR	HLR
MAP_REGSS	MSC	VLR
MAP_REMUSRFREE	HLR	VLR
MAP_SMDELIVSTAT	SMS-GMSC	HLR
MAP_RESET	HLR	VLR or SGSN
MAP_RESTORE	VLR	HLR
MAP_SENDGCEND	Relay MSC	Anchor MSC

MAP_SENDEND	MSC-B	MSC-A
MAP_SENDAUTHI	VLR or SGSN	HLR
MAP_SENDIMSI	VLR	MSC
MAP_SENDID	VLR	old VLR
MAP_SENDSMSRI	SMS-GMSC	HLR
MAP_SENDGPRSRI	GGSN	HLR
MAP_SENDLCSRI	GMLC	HLR
MAP_SENDRI	GMSC or SCF	HLR or NPLR
MAP_SETREP	HLR	VLR
MAP_STATUS	VLR	HLR
MAP_SUBLOCREP	VMSC or SGSN	GMLC
MAP_USSDNTFY	SCF	HLR
	HLR	VLR
	VLR	MSC
MAP_USSDREQ	SCF	HLR
	HLR	VLR
	VLR	MSC
MAP_UDGPRSLOC	SGSN	HLR
MAP_UDLOC	VLR	HLR

2.3 Service Categories

2.3.1 Mobility Management

2.3.1.1 Location Management Services

Service	**Invoker**	**Performer**
MAP_UDLOC	VLR	HLR
MAP_CANCLOC	HLR	VLR or SGSN
MAP_SENDID	VLR	old VLR
MAP_PURGEMS	VLR or SGSN	HLR
MAP_UDGPRSLOC	SGSN	HLR
MAP_MMEVENT	VLR or SGSN	SCF

2.3.1.2 Paging and Search

Service	**Invoker**	**Performer**
MAP_PAGE	VLR	MSC
MAP_SEARCH	VLR	MSC

2.3.1.3 Access Management Services

Service	**Invoker**	**Performer**
MAP_PROCAS	MSC	VLR

2.3.1.4 Handover Services

Service	**Invoker**	**Performer**
MAP_PREPHO	MSC-A	MSC-B
MAP_SENDEND	MSC-B	MSC-A
MAP_PROCAS	MSC-B	MSC-A

MAP_FAS	MSC-A	MSC-B
MAP_PREPSH	MSC-B	MSC-A
MAP_ALLOCHN	MSC	VLR
MAP_SENDHOREP	VLR	MSC-B

2.3.1.5 Authentication Management Services

Service	**Invoker**	**Performer**
MAP_AUTH	VLR	MSC
MAP_SENDAUTHI	VLR or SGSN	HLR
MAP_AUTHFAIL	VLR or SGSN	HLR

2.3.1.6 Security Management Services

Service	**Invoker**	**Performer**
MAP_SETCM	VLR	MSC

2.3.1.7 IEMI Managment Services

Service	**Invoker**	**Performer**
MAP_CKIEMI	VLR	MSC
	MSC or SGSN	EIR
MAP_GETIEMI	VLR	MSC

2.3.1.8 Subscriber Management Services

Service	**Invoker**	**Performer**
MAP_INSSUBSD	HLR	VLR or SGSN
MAP_DELSUBD	HLR	VLR or SGSN

2.3.1.9 Identity Management Services

Service	**Invoker**	**Performer**
MAP_PROVIMSI	VLR	MSC
MAP_FNEWIMSI	VLR	MSC

2.3.1.10 Fault Recovery Services

Service	**Invoker**	**Performer**
MAP_RESET	HLR	VLR or SGSN
MAP_FCKSSIND	HLR	VLR
MAP_RESTORE	VLR	HLR

2.3.1.11 Subscriber Information Services

Service	**Invoker**	**Performer**
MAP_ANYQRY	SCF	HLR or GMLC or NPLR
MAP_PROVSUBI	VLR	SGSN
MAP_ANYSUB	SCF	HLR
MAP_ANYMOD	SCF	HLR
	PNA	VLR or SGSN
MAP_SUBSDATAMOD	HLR	SCF

2.3.2 Operation and Maintenance

2.3.2.1 Subscriber Tracing Services

Service	**Invoker**	**Performer**
MAP_ACTTM	HLR	VLR or SGSN
MAP_DEACTTM	HLR	VLR or SGSN
MAP_TRACESA	VLR	MSC

2.3.2.2 Miscellaneous Operation and Maintenance Services

Service	**Invoker**	**Performer**
MAP_SENDIMSI	VLR	MSC

2.3.3 Call Handling

2.3.3.1 Call Handling Services

Service	**Invoker**	**Performer**
MAP_SENDRI	GMSC or SCF	HLR or NPLR
MAP_PROVRN	HLR	VLR
MAP_RESUME	VMSC	GMSC
MAP_PREPGC	Anchor MSC	Relay MSC
MAP_PROCGC	Relay MSC	Anchor MSC
MAP_FGRCSIG	Anchor MSC	Relay MSC
MAP_SENDGCEND	Relay MSC	Anchor MSC
MAP_SETREP	HLR	VLR
MAP_STATUS	VLR	HLR
MAP_REMUSRFREE	HLR	VLR
MAP_ISTALERT	VMSC or GMSC	HLR
MAP-ISTCMD	HLR	VMSC or GMSC

2.3.4 Supplementary Service

2.3.4.1 Supplementary Service Services

Service	**Invoker**	**Performer**
MAP_REGSS	MSC	VLR
MAP_ERASESS	MSC	VLR
	VLR	HLR
MAP_ACTSS	MSC	VLR
	VLR	HLR
MAP_DEACTSS	MSC	VLR
	VLR	HLR
MAP_INTERGSS	MSC	VLR
	VLR	HLR
MAP_REGPW	MSC	VLR
	VLR	HLR
MAP_GETPASS	HLR	VLR
	VLR	MSC
MAP_PROCUSS	MSC	VLR
	VLR	HLR
MAP_USSDREQ	SCF	HLR
	HLR	VLR

	VLR	MSC
MAP_USSDNTFY	SCF	HLR
	HLR	VLR
	VLR	MSC
MAP_SSINV	MSC or HLR	SCF
MAP_REGCC	MSC	VLR
	VLR	HLR
MAP_ERASECC	MSC	VLR
	VLR	HLR

2.3.5 SMS Management Services

2.3.5.1 Short Message Service Management Services

Service	Invoker	Performer
MAP_SENDSMSRI	SMS-GMSC	HLR
MAP_MOFSMS	MSC	SMS-IWMSC
MAP_SMDELIVSTAT	SMS-GMSC	HLR
MAP_RDYSM	MSC	VLR
MAP_RDYSM	VLR or SGSN	HLR
MAP_ALERTSC	HLR	SMS-IWMSC
MAP_INFORMSC	HLR	SMS-GMSC
MAP_SENDMTSMSI	MSC	VLR
MAP_SENDMOSMSI	MSC	VLR
MAP_MTFSMS	SMS-GMSC	MSC or SGSN

2.3.6 Network-Requested PDP Context Activation

2.3.6.1 Network-Requested PDP Context Activation Services

Service	Invoker	Performer
MAP_SENDGPRSRI	GGSN	HLR
MAP_FAILURE	GGSN	HLR
MAP_MSGPRSPRES	HLR	GGSN

2.3.7 Location Service Management

2.3.7.1 Location Service Management Services

Service	Invoker	Performer
MAP_SENDLCSRI	GMLC	HLR
MAP_PROVSLOC	GMLC	VMSC or SGSN
MAP_SUBLOCREP	VMSC or SGSN	GMLC

2.4 MAP User Entities

2.4.1 Mobile Switching Centre (MSC) Service Sets

Table X. MSC performed services.

Service	Invoker	Performer
MAP_PAGE	VLR	MSC
MAP_SEARCH	VLR	MSC
MAP_SENDHOREP	VLR	MSC
MAP_AUTH	VLR	MSC
MAP_SETCM	VLR	MSC
MAP_CKIEMI	VLR	MSC
MAP_GETIEMI	VLR	MSC
MAP_PROVIMSI	VLR	MSC
MAP_FNEWIMSI	VLR	MSC
MAP_TRACESA	VLR	MSC
MAP_SENDIMSI	VLR	MSC
MAP_ISTCMD	HLR	MSC
MAP_GETPASS	VLR	MSC
MAP_USSDREQ	VLR	MSC
MAP_USSDNTFY	VLR	MSC
MAP_MTFSMS	SMS-GMSC	MSC
MAP_PROVSLOC	GMLC	MSC

Table X. MSC invoked services.

Service	Invoker	Performer
MAP_PROCAS	MSC	VLR
MAP_CKIEMI	MSC	EIR
MAP_ALLOCHN	MSC	VLR
MAP_RESUME	MSC	GMSC
MAP_ISTALERT	MSC	HLR
MAP_REGSS	MSC	VLR
MAP_ERASESS	MSC	VLR
MAP_ACTSS	MSC	VLR
MAP_DEACTSS	MSC	VLR
MAP_INTERGSS	MSC	VLR
MAP_REGPW	MSC	VLR
MAP_PROCUSS	MSC	VLR
MAP_SSINV	MSC	SCF
MAP_REGCC	MSC	VLR
MAP_ERASECC	MSC	VLR
MAP_MOFSMS	MSC	SMS-IWMSC
MAP_RDYSM	MSC	VLR
MAP_SENDMTSMSI	MSC	VLR
MAP_SENDMOSMSI	MSC	VLR
MAP_SUBLOCREP	MSC	GMLC

Table X. MSC peer services.

Service	Invoker	Performer
MAP_PREPHO	MSC-A	MSC-B
MAP_SENDEND	MSC-B	MSC-A
MAP_PROCAS	MSC-B	MSC-A

MAP_FAS	MSC-A	MSC-B
MAP_PREPSH	MSC-B	MSC-A
MAP_PREPGC	Anchor MSC	Relay MSC
MAP_PROCGC	Relay MSC	Anchor MSC
MAP_FGRCSIG	Anchor MSC	Relay MSC
MAP_SENDGCEND	Relay MSC	Anchor MSC

2.4.2 Home Location Register (HLR) Service Sets

Table 1. HLR performed services.

Service	Invoker	Performer
MAP_UDLOC	VLR	HLR
MAP_PURGEMS	VLR or SGSN	HLR
MAP_UDGPRSLOC	SGSN	HLR
MAP_SENDAUTHI	VLR or SGSN	HLR
MAP_AUTHFAIL	VLR or SGSN	HLR
MAP_RESTORE	VLR	HLR
MAP_ANYQRY	SCF	HLR
MAP_ANYSUB	SCF	HLR
MAP_ANYMOD	SCF	HLR
MAP_SENDRI	GMSC or SCF	HLR
MAP_STATUS	VLR	HLR
MAP_ISTALERT	VMSC or GMSC	HLR
MAP_ERASESS	VLR	HLR
MAP_ACTSS	VLR	HLR
MAP_DEACTSS	VLR	HLR
MAP_INTERGSS	VLR	HLR
MAP_REGPW	VLR	HLR
MAP_PROCUSS	VLR	HLR
MAP_REGCC	VLR	HLR
MAP_ERASECC	VLR	HLR
MAP_SENDSMSRI	SMS-GMSC	HLR
MAP_SMDELIVSTAT	SMS-GMSC	HLR
MAP_RDYSM	VLR or SGSN	HLR
MAP_SENDGPRSRI	GGSN	HLR
MAP_FAILURE	GGSN	HLR
MAP_USSDREQ	SCF	HLR
MAP_SENDLCSRI	GMLC	HLR
MAP_USSDNTFY	SCF	HLR

Table 2. HLR invoked services.

Service	Invoker	Performer
MAP_CANCLOC	HLR	VLR or SGSN
MAP_INSSUBSD	HLR	VLR or SGSN
MAP_DELSUBD	HLR	VLR or SGSN
MAP_RESET	HLR	VLR or SGSN
MAP_FCKSSIND	HLR	VLR

MAP_SUBSDATAMOD	HLR	SCF
MAP_ACTTM	HLR	VLR or SGSN
MAP_DEACTTM	HLR	VLR or SGSN
MAP_PROVRN	HLR	VLR
MAP_SETREP	HLR	VLR
MAP_REMUSRFREE	HLR	VLR
MAP_ISTCMD	HLR	VMSC or GMSC
MAP_GETPASS	HLR	VLR
MAP_USSDREQ	HLR	VLR
MAP_USSDNTFY	HLR	VLR
MAP_SSINV	HLR	SCF
MAP_ALERTSC	HLR	SMS-IWMSC
MAP_INFORMSC	HLR	SMS-GMSC
MAP_MSGPRSPRES	HLR	GGSN

2.4.3 Visitor Location Register (VLR) Service Sets

Table 3. VLR performed services.

Service	**Invoker**	**Performer**
MAP_CANCLOC	HLR	VLR
MAP_PROCAS	MSC	VLR
MAP_ALLOCHN	MSC	VLR
MAP_INSSUBSD	HLR	VLR
MAP_DELSUBD	HLR	VLR
MAP_SENDID	VLR	old VLR
MAP_RESET	HLR	VLR
MAP_FCKSSIND	HLR	VLR
MAP_ANYMOD	PNA	VLR
MAP_ACTTM	HLR	VLR
MAP_DEACTTM	HLR	VLR
MAP_PROVRN	HLR	VLR
MAP_SETREP	HLR	VLR
MAP_REMUSRFREE	HLR	VLR
MAP_REGSS	MSC	VLR
MAP_ERASESS	MSC	VLR
MAP_ACTSS	MSC	VLR
MAP_DEACTSS	MSC	VLR
MAP_INTERGSS	MSC	VLR
MAP_REGPW	MSC	VLR
MAP_GETPASS	HLR	VLR
MAP_PROCUSS	MSC	VLR
MAP_USSDREQ	HLR	VLR
MAP_USSDNTFY	HLR	VLR
MAP_REGCC	MSC	VLR
MAP_ERASECC	MSC	VLR
MAP_RDYSM	MSC	VLR
MAP_SENDMTSMSI	MSC	VLR
MAP_SENDMOSMSI	MSC	VLR

Table 4. VLR invoked services.

Service	Invoker	Performer
MAP_UDLOC	VLR	HLR
MAP_SENDID	VLR	old VLR
MAP_PURGEMS	VLR	HLR
MAP_MMEVENT	VLR	SCF
MAP_PAGE	VLR	MSC
MAP_SEARCH	VLR	MSC
MAP_SENDHOREP	VLR	MSC-B
MAP_AUTH	VLR	MSC
MAP_SENDAUTHI	VLR	HLR
MAP_AUTHFAIL	VLR	HLR
MAP_SETCM	VLR	MSC
MAP_CKIEMI	VLR	MSC
MAP_GETIEMI	VLR	MSC
MAP_PROVIMSI	VLR	MSC
MAP_FNEWIMSI	VLR	MSC
MAP_RESTORE	VLR	HLR
MAP_PROVSUBI	VLR	SGSN
MAP_TRACESA	VLR	MSC
MAP_SENDIMSI	VLR	MSC
MAP_STATUS	VLR	HLR
MAP_ERASESS	VLR	HLR
MAP_ACTSS	VLR	HLR
MAP_DEACTSS	VLR	HLR
MAP_INTERGSS	VLR	HLR
MAP_REGPW	VLR	HLR
MAP_GETPASS	VLR	MSC
MAP_PROCUSS	VLR	HLR
MAP_USSDREQ	VLR	MSC
MAP_USSDNTFY	VLR	MSC
MAP_REGCC	VLR	HLR
MAP_ERASECC	VLR	HLR
MAP_RDYSM	VLR	HLR

2.4.4 Equipment Identity Register (EIR) Service Sets

Table X. EIR performed services.

Service	Invoker	Performer
MAP_CKIEMI	MSC or SGSN	EIR

2.4.5 Serving GPRS Support Node (SGSN) Service Sets

Table X. SGSN performed services.

Service	Invoker	Performer
MAP_CANCLOC	HLR	SGSN
MAP_INSSUBSD	HLR	SGSN

MAP_DELSUBD	HLR	SGSN
MAP_RESET	HLR	SGSN
MAP_PROVSUBI	VLR	SGSN
MAP_ANYMOD	PNA	SGSN
MAP_ACTTM	HLR	SGSN
MAP_DEACTTM	HLR	SGSN
MAP_MTFSMS	SMS-GMSC	SGSN
MAP_PROVSLOC	GMLC	SGSN

Table X. SGSN invoked services.

Service	Invoker	Performer
MAP_PURGEMS	SGSN	HLR
MAP_UDGPRSLOC	SGSN	HLR
MAP_MMEVENT	SGSN	SCF
MAP_SENDAUTHI	SGSN	HLR
MAP_AUTHFAIL	SGSN	HLR
MAP_CKIEMI	SGSN	EIR
MAP_RDYSM	SGSN	HLR
MAP_SUBLOCREP	SGSN	GMLC

2.4.6 Gateway GPRS Support Node (GGSN) Service Sets

Table X. GGSN performed services.

Service	Invoker	Performer
MAP_MSGPRSPRES	HLR	GGSN

Table X. GGSN invoked services.

Service	Invoker	Performer
MAP_SENDGPRSRI	GGSN	HLR
MAP_FAILURE	GGSN	HLR

2.4.7 Interworking MSC (IWMSC) for Short Message Service Service Sets

Table X. SMS-IWMSC performed services.

Service	Invoker	Performer
MAP_MOFSMS	MSC	SMS-IWMSC
MAP_ALERTSC	HLR	SMS-IWMSC

2.4.8 Gateway MSC (GMSC) for Short Message Service Service Sets

Table X. SMS-GMSC performed services.

Service	Invoker	Performer
MAP_INFORMSC	HLR	SMS-GMSC

Table X. SMS-GMSC invoked services.

Service	Invoker	Performer
MAP_SENDSMSRI	SMS-GMSC	HLR
MAP_SMDELIVSTAT	SMS-GMSC	HLR
MAP_MTFSMS	SMS-GMSC	MSC or SGSN

2.4.9 Gateway Mobile Location Center (GMLC) Service Sets

Table X. GMLC performed services.

Service	Invoker	Performer
MAP_ANYQRY	SCF	GMLC
MAP_SUBLOCREP	VMSC or SGSN	GMLC

Table X. GMLC invoked services.

Service	Invoker	Performer
MAP_SENDLCSRI	GMLC	HLR
MAP_PROVSLOC	GMLC	VMSC or SGSN

3 MAPI Services Definition

This chapter describes the services of the MAP primitives. Time-sequence diagrams that illustrate the sequence of primitives is included. (Conventions for the time-sequence diagrams are defined in ITU-T X.210). The format of the primitives will be defined later in the document.

3.1 Local Management Services Definition

The services defined in this section are outside the scope of the international standards. These services apply to all applications contexts and specific service primitives. They are invoked for the initialization or de-initialization of a stream connected to the MAP provider. They are also used to manage options supported by the MAP provider and to report information on the supported parameter values.

3.1.1 MAP Information Reporting Service

This service provides information on the options supported by the MAP provider.

- `MAP_INFO_REQ`: This primitive requests that the MAP provider return the values of all the supported protocol parameters. This request may be invoked during any phase.
- `MAP_INFO_ACK`: This primitive is in response to the `MAP_INFO_REQ` primitive and returns the values of the supported protocol parameters to the MAP user.

The sequence of primitives for mobile application part information management is shown in Figure 2.

Figure 2. Sequence of Primitives;
MAP Information Reporting Serivce

3.1.2 MAP User Bind Service

The service allows a network address and application contexts to be associated with a stream. It allows the MAP user to negotiate the number of dialogues that can remain outstanding for that MAP user (a dialogue is considered unacknowledged while it is awaiting a corresponding response or abort from the MAP user). This service also defines a mechanism that allows a stream (bound to a network address of the MAP user) to be reserved to handle remote initiated dialogues only. This stream is referred to as the listener stream.

- `MAP_BIND_REQ`: This primitive requests that the MAP user be bound to a particular network address and application context, and negotiate the number of allowable outstanding open indications for that address.
- `MAP_BIND_ACK`: This primitive is in response to the `MAP_BIND_REQ` primitive and indicates to the user that the specified MAP user has been bound to a network address and application context.

The sequence of primitives for the MAP user bind service is shown in Figure 3.

Figure 2. Sequence of Primitives;
MAP User Bind Service

3.1.3 MAP User Unbind Service

This service allows the MAP user to be unbound from a network address and application context.

- `MAP_UNBIND_REQ`: This primitive requests that the MAP user be unbound from the network address and application context(s) to which it was previously bound.

The sequence of primitives for MAP user unbind service is shown in Figure 4.

Figure 4. Sequence of Primitives;
MAP User Unbind & Receipt Acknowledgement Services

3.1.4 Receipt Acknowledgement Service

- `MAP_OK_ACK`: This primitive indicates to the MAP user that the previous MAP user originated primitive was received successfully by the MAP provider.

An example showing the sequence of primitives for successful receipt acknowledgement is depicted in Figure 4.

3.1.5 Options Management Service

This service allows the MAP user to manage the QOS parameter values and other options associated with the MAP provider.

- `MAP_OPTMGMT_REQ`: This primitive allows the MAP user to select default values for QoS parameters within the range supported by the MAP provider, and to indicate the default selection of additional options supported by the MAP provider.

Figure 5 shows the sequence of primitives for MAP options management service.

Figure 5. Sequence of Primitives;
Options Management Service.

3.1.6 Error Acknowledgement Service

- `MAP_ERROR_ACK`: This primitive indicates to the MAP user that a non-fatal error has occurred in the last MAP user originated request or response primitive (listed in Figure 6), on the stream.

Figure 6 shows the sequence of primitives for the error acknowledgement service.

Figure 6. Sequence of Primitives;
Error Acknowledgement Service.

3.2 Dialogue Handling Definition

This section describes the required MAP service primitives that define the Dialogue Handling interface.

The queue model for Dialogue Handling is discussed in more detail in CCITT X.219.

The queue module represents the operation of a MAP dialogue in the abstract by a pair of queues linking the two MAP user entities. There is one queue for each direction of information flow. Each queue represents a flow control function in one direction of transfer. The ability of a user to add objects to a queue will be determined by the behaviour of the user removing objects from that queue, and the state of the queue. The pair of queues is considered to be available for each potential MAP dialogue. Objects that are entered or removed from the queue are either as a result of interactions at the two MAP users, or as a result of MAP provider initiatives.

- A queue is empty until a dialogue object has been entered and can be returned to this state, with loss of its contents, by the MAP provider.
- Objects can be entered into a queue as a result of the actions of the source MAP user, subject to control by the MAP provider.
- Objects may also be entered into a queue by the MAP provider.
- Objects are removed from the queue under the control of the MAP user in n the same order as they were entered except:
  - if the object is of a type defined to be able to advance ahead of the preceding object (however, no object is defined to be able to advance ahead of another object of the same type), or
  - if the following object is defined to be destructive with respect to the preceding object on the queue. If necessary, the last object on the queue will be deleted to allow a destructive object to be entered – the will therefore always be added to the queue. For example, "abort" objects are defined to be destructive with respect to all other objects.

Table Y shows the ordering relationships among the queue model objects.

Table Y. Ordering Relationship Between Queue Model Objects.

Object X					
Object Y	**Open**	**Components**	**Delimiter**	**Close**	**Abort**
Open	N/A	–	–	DES	
Components	N/A	–	–	DES	
Deilmiter	N/A	–	–	DES	
Close	N/A	–	–	DES	
Abort	N/A	N/A	N/A	–	

AA Indicates that Object X is defined to be able to advance ahead of preceding Object Y.

DES Indicates that Object X is defined to be destructive with respect to the preceding Object Y.

– Indicates that Object X is neither destructive with respect to Object Y, nor able to advance ahead of Object Y.

N/A Indicates that Object X will not occur in a position succeeding Object Y in a valid state of a queue.

3.2.1 Dialogue Establishment Phase

A pair of queues is associated with a MAP dialogue between two MAP user entities when the MAP provider receives a `MAP_OPEN_REQ` primitive at one of the MAP user interfaces resulting in the "begin" object being entered into the queue. The queues will remain associated with the MAP dialogue until a `MAP_CLOSE_REQ` or *indication* primitive (resulting in an "end" object) is either entered or removed from a queue. Similarly, in the queue from the called MAP user, objects can be entered into the queue only after the "continue" object associated with the `MAP_OPEN_RES` primitive has been entered into the queue. Alternatively, the called MAP user can enter a an "end" or "abort" object into the queue instead of the "continue" object to terminate the dialogue.

The MAP dialogue establishment procedure will fail if the MAP provider is unable to establish a TCAP dialogue, or if the destination MAP user is unable to accept the `MAP_OPEN_IND`.

3.2.1.1 User Primitives for Successful MAP Dialogue Establishment

- MAP_OPEN_REQ: This primitive requests that the MAP provider establish a dialogue to the specified destination under an application context.
- MAP_OPEN_RES: This primitives requests that the MAP provider accept a previous dialogue indication.

3.2.1.2 Provider Primitives for Successful MAP Dialogue Establishment

- MAP_OPEN_IND: This primitive indicates to the MAP user that a dialogue request has been made within an application context by a user at the specified source address.
- MAP_OPEN_CON: This primitive indicates to the MAP user that a dialogue request has been confirmed on the specified responding address.

The sequence of primitives for a successful MAP dialogue establishment is defined by the time sequence diagram as shown in Figure 7. The sequence of primitives for the MAP dialogue response token value determination is shown in Figure 8 (procedures for MAP response token value determination are discussed later).

**Figure 7. Sequence of Primitives;
Successful MAP Dialogue Establishment**

**Figure 8. Sequence of Primitives;
MAP Dialogue Response Token Value Determination**

3.2.2 Dialogue Component Transfer Phase

Flow control within the MAP dialogue is performed by management of the queue capacity, and by allowing objects of certain types to be inserted to the queues as shown in Table 4.

Table 4. Flow Control Relationship Between Queued Model Objects

Object X				
Object Y	Class 1	Class 2	Class 3	Class 4
Class 1	Yes	Yes	Yes	Yes
Class 2	Yes	Yes	Yes	Yes
Class 3	Yes	Yes	Yes	Yes
Class 4	Yes	Yes	Yes	Yes

Yes The addition of Object X may prevent further addition of Object Y.

No The addition of Object X may not prevent the addition of Object Y.

3.2.2.1 User Primitives for Component Transfer

- **Service Specific Requests**: These primitives request that the MAP provider invoke a service and transfer the specified (and any subsequent) components.
- **Service Specific Responses**: These primitives request that the MAP provider return result for a service invocation and transfer the specified (and any subsequent) components.
- MAP_DELIM_IND: This primitive requests that the MAP provider deliver any accumulated components from previous service specific request or response primitives.

3.2.2.2 Provider Primitives for Component Transfer

- **Service Specific Indications**: These primitives indicate to the MAP user an invocation and the indicated (and any subsequent) components.
- **Service Specific Confirmations**: These primitives indicate to the MAP user a return result and the indicated (and any subsequent) components.
- `MAP_DELIM_IND`: This primitive indicates that the MAP provider has delivered any accumulated components belonging to the previous service specific indication or confirmation primitive.

Figure 9 shows the sequence of primitives for successful component transfer. The sequence of primitives may remain incomplete if a `MAP_ABORT` or `MAP_CLOSE` primitive occurs.

Figure 9. Sequence of Primitives;
Component Transfer.

The sequence of primitives in a successful Class 1 operation is defined in the time sequence diagram as shown in Figure 10.

Figure 10. Sequence of Primitives;
Successful Confirmation of Receipt.

The sequence of primitives as shown above may remain incomplete if a `MAP_UABORT`, `MAP_PABORT` or `MAP_CLOSE` primitive occurs (see Table 3). A MAP user must not issue a *Service Specific Response* primitive if no *Service Specific Indication* primitive for a confirmed service has been received, or if all such *Service Specific Indication* primitives have been previously confirmed. Following a `MAP_UABORT` or `MAP_PABORT`, a MAP user may not issue a *Service Specific Response* to confirm outstanding *Service Specific Indication* primitives received before the abort procedure was signalled.

Note — The witholding of confirmation of a service by a MAP user can have an effect on the attainable throughput.

The sequence of primitives for component transfer is shown in the time sequence diagram in Figure 11. This sequence of primitives may remain incomplete if a `MAP_ABORT` or `MAP_CLOSE` primitive is issued.

Figure 11. Sequence of Primitives;
Expedited Data Transfer.

3.2.3 Dialogue Termination Phase

The MAP dialogue release procedure is initialized by the insertion of an “abort” or “end” object into the queue. As shown in Table 3, the “abort” is destructive with respect to other objects in the queue, and eventually results in the emptying of queues and termination of the MAP dialogue.

The sequence of primitivies depends on the origin of the release action. The sequence of primitives may be:

1. invoked by one MAP user, with a request from that MAP user leading to an indication to the other;
2. invoked by both MAP users, with a request from each of the MAP users;
3. invoked by the MAP provider, with an indication to each of the MAP users;
4. invoked independently by one MAP user and the MAP provider, with a request from the originating MAP user and an indication to the other.

3.2.3.1 User Primitives for Dialogue Abort

- MAP_ABORT_REQ: This primitive requests that the MAP provider deny an outstanding request for an association (MAP_OPEN_IND) or abort an existing association.

3.2.3.2 Provider Primitives for Dialogue Abort

- MAP_UABORT_IND: This primitive indicates to the MAP user that either a request for association has been denied or an existing association has been terminated.
- MAP_PABORT_IND: This primitive indicates to the MAP user that either a request for association has been denied or an existing association has been terminated.

The sequence of primitives are shown in the time sequence diagrams in n Figures 16, 17, 18 and 19.

**Figure 16. Sequence of Primitives;
MAP User Invoked Abort.**

**Figure 17. Sequence of Primitives;
Simultaneous MAP User Invoked Abort.**

**Figure 18. Sequence of Primitives;
MAP Provider Invoked Abort.**

**Figure 19. Sequence of Primitives;
Simultaneous MAP User and MAP Provider Invoked Abort.**

A MAP user may reject a MAP association establishment by issuing a MAP_ABORT_REQ. The originator parameter in the MAP_UABORT_IND primitive will indicate MAP user invoked release. The sequence of events is shown in Figure 20.

**Figure 20. Sequence of Primitives;
MAP User Rejection of a MAP Association Establishment Attempt.**

If the MAP provider is unable to establish a MAP association, it indicates this to the requester by a MAP_PABORT_IND. The originator in this primitive indicates a MAP provider invoked release. This is shown in Figure 21.

**Figure 21. Sequence of Primitives;
MAP Provider Rejection of a MAP Association Establishement Attempt.**

4 MAPI Primitives

This section describes the format and parameters of the MAPI primitives (Appendix A shows the mapping of the MAPI primitives to the primitives defined in 3GPP TS 29.002 and ANSI41-D). Also, it discusses the states in which the primitive is valid, the resulting state and the acknowledgement that the primitive expects. (The state/event tables for these primitives are shown in Appendix B. The precedence tables for the MAPI primitives are shown in Appendix C.) Rules for 3GPP and ANSI conformance are described in Addendum to this document.

Tabel 4. Local Management Service Primitives.

SERVICE	PRIMITIVE	PARAMETERS
Information	`MAP_INFO_REQ`	
Reporting	`MAP_INFO_ACK`	
Bind	`MAP_BIND_REQ`	
	`MAP_BIND_ACK`	
Unbind	`MAP_UNBIND_REQ`	
Error	`MAP_ERROR_ACK`	
Success	`MAP_OK_ACK`	

Tables 5, 6 and 7 provide a summary of the MAP primitives and their parameters.

Table 5. Association Establishment MAP Service Primitives.

SERVICE	PRIMITIVE	PARAMETERS
Association	`MAP_OPEN_REQ`	(Application context name, Destination address, Destination reference, Originating address, Originating reference, Specific information)
Establishment	`MAP_OPEN_IND`	(Application context name, Destination address, Destination reference, Originating address, Originating reference, Specific information)
	`MAP_OPEN_RES`	(Application context name, Responding address, Result, Refuse-reason, Specific information)
	`MAP_OPEN_CON`	(Application context name, Responding address, Result, Refuse-reason, Specific information, Provider-error)

Table 6. Component Transfer Service Primitives.

SERVICE	PRIMITIVE	PARAMETERS
Service	MAP_*XXX*_REQ	(Specific information)
Specific	MAP_*XXX*_IND	(Specific information)
Primitives	MAP_*XXX*_RES	(Specific information)
	MAP_*XXX*_CON	(Specific information)
Delimitation	`MAP_DELIM_REQ`	(none)
	`MAP_DELIM_IND`	(none)

Table 7. Association Termination MAP Service Primitives.

SERVICE	PRIMITIVE	PARAMETERS
Association	`MAP_CLOSE_REQ`	(Release method, Specific information)

Termination	`MAP_CLOSE_IND`	(Specific information)
Assocation	`MAP_ABORT_REQ`	(User reason, Diagnostic information, Specific information)
Abort	`MAP_UABORT_IND`	(User reason, Diagnostic information, Specific information)
Abort	`MAP_PABORT_IND`	(Provider reason, Source)
Notice	`MAP_NOTICE_IND`	()

Table 8. Secure Transport Primitives.

SERVICE	**PRIMITIVE**	**PARAMETERS**
Class 1	`MAP_STC1_REQ`	
	`MAP_STC1_IND`	
Class 2	`MAP_STC2_REQ`	
	`MAP_STC2_IND`	
Class 3	`MAP_STC3_REQ`	
	`MAP_STC3_IND`	
Class 4	`MAP_STC4_REQ`	
	`MAP_STC4_IND`	

4.1 Management Primitives

These primitives appy to all protocol classes.

4.1.1 Information Request

MAP_INFO_REQ

This primitive requests the MAP provider to return the values of all supported protocol parameters (see Section 4.1.2 [Information Acknowledgement], page 32), and also the current state of the MAP provider.[1]

Format

The format of the message is one 'M_PROTO' message block and its structure is as follows:

```
typedef struct {
        ulong PRIM_type;    /* always MAP_INFO_REQ */
} MAP_info_req_t;
```

Parameters

PRIM_type:
Indicates the primitive type.

Valid States

This primitive is valid in any state where a local acknowledgement is not pending.

New State

The new state remains unchanged from the old state.

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledgements upon receipt of the pirmitive:

— **Successful:** Acknowledgement of the primitive with the MAP_INFO_ACK primitive described in Section 4.1.2 [Information Acknowledgement], page 32.

— **Non-fatal Errors:** There are no errors associated with the issuance of this primitive.

[1] The MAP_INFO_REQ and MAP_INFO_ACK primitives have no effect on the state of the MAP provider and do not appear in the state tables in Appendix B [State/Event Tables], page 389.

4.1.2 Information Acknowledgement

MAP_INFO_ACK

This primitive indicates to the MAP users any relevant protocol-dependent parameters.[1] It should be initiated in response to the `MAP_INFO_REQ` primitive described above in Section 4.1.1 [Information Request], page 31.

Format

The format of the message is one '`M_PCPROTO`' message block. The format of the '`M_PCPROTO`' message block is as follows:

```
typedef struct {
    ulong PRIM_type;        /* always MAP_INFO_ACK */
    ulong TSDU_size;        /* maximum TSDU size */
    ulong ETSDU_size;       /* maximum ETSDU size */
    ulong CDATA_size;       /* connect data size  */
    ulong DDATA_size;       /* disconnect data size */
    ulong ADDR_size;        /* address size */
    ulong ADDR_length;      /* address length */
    ulong ADDR_offset;      /* address offset */
    ulong QOS_length;       /* default QOS values length */
    ulong QOS_offset;       /* default QOS values offset */
    ulong QOS_range_length; /* QOS range length */
    ulong QOS_range_offset; /* QOS range offset */
    ulong OPTIONS_flags;    /* bit masking for options */
    ulong TIDU_size;        /* transaction interface data size */
    long  SERV_type;        /* service type */
    ulong CURRENT_state;    /* current state */
    ulong PROVIDER_type;    /* type of provider */
    ulong NODU_size;        /* optimal TSDU size */
    ulong PROTOID_length;   /* length of bound protocol ids */
    ulong PROTOID_offset;   /* offset of bound protocol ids */
    ulong MAPI_version;     /* supported MAPI version number */
} MAP_info_ack_t;
```

Parameters

PRIM_type Indicates the primitive type.

TSDU_size Specifies the maximum size (in octets) of Transaction Service User Data supported by the MAP provider.

ETSDU_size
Specifies the maximum size (in octets) of Expedited Transaction Service User Data supported by the MAP provider.

CDATA_size
Specifies the maximum number of octets of data that may be associated with a transaction initiation primitive for operations class 1, 2 and 3 operation.

DDATA_size
Specifies the maximum number of octets of data that may be associated with a transaction termination primitive for operations class 1, 2 and 3 operation.

[1]

ADDR_size
Specifies the maximum size (in decimal digits) of a network address.

ADDR_length, ADDR_offset
Specifies the length in bytes and offset from the beginning of the `M_PCPROTO` message block of the network address bound on the stream on which the `MAP_INFO_REQ` was issued (a network address is bound to a stream via a `MAP_BIND_REQ`).

QOS_length, QOS_offset
QOS_range_length, QOS_range_offset
OPTIONS_flags
TIDU_size[2]
This is the size of the MAP protocol interface data unit, and should not exceed the tunable system limit, if non-zero, for the size of a STREAMS message.

SERV_type This field specified the service type supported by the MAP provider, and is one of the following:

CURRENT_state
THis is the current state of the MAP provider.

PROVIDER_type
This field specifies additional properties specific to the MAP provider and may alter the way the MAP user communicates. MAP providers supporting the features of XTI in XPG4 and beyond must send up a version number as specified below. The following flags may be set by the provider:

`SENDZERO` This flag indicates that the MAP provider support the sending of zero-length (component-less) TSDUs.

`XPG4_1` This indicates that the MAP provider conforms to XTI in XPG4 and supports primitives `MAP_ADDR_REQ` and `MAP_ADDR_ACK`.

TODU_size
PROTOID_length, PROTOID_offset
MAPI_version

Flags

Valid States

New State

Rules

The following rules apply when the type of service is operations class 4:

— The *ETSDU_size*, *CDATA_size* and *DDATA_size* fields should be '`-2`'.

— The *TSDU_size* should be equal to the *TIDU_size*.

[2] This is the amount of user components that may be present in a single service specific primitive.

4.1.3 Bind Application Context Request

MAP_BIND_REQ

This primitive requests that the MAP provider bind a protocol address to the *stream*, negotiate the number of dialogue open indications allowed to be outstanding by the MAP provider for the specified protocol address, and activate[1] the *stream* associated with the protocol address.

Format

The format of the message is one 'M_PROTO' message block. The format of the 'M_PROTO' message block is as follows:

```
typedef struct {
    ulong PRIM_type;                /* always MAP_BIND_REQ */
    ulong ADDR_length;              /* length of protocol address */
    ulong ADDR_offset;              /* offset of protocol address */
    unsigned long DIAIND_number;    /* requested number of dialogue indications to be queued */
} MAP_bind_req_t;
```

Parameters

PRIM_type
Indicates the primitive type.

ADDR_length
The length[2] of the protocol address to be bound to the *stream*.

ADDR_offset
The offset from the beiginning of the 'M_PROTO' block where the protocol address begins. The address in the 'M_PROTO' message block is however, aligned the same as it was received from the MAP user.

DIAIND_number
[3] The requested number of connection indications[4] allowed to be outstanding by the MAP provider for the specified protocol address.

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to geenrate one of the following acknowledgements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitive:

[1] A *stream* is viewed as active when the MAP provider may receive and transmit components (Trasaction Protocol Data Units) associated with the *stream*

[2] All lengths, offsets and sizes in all structures refer to the number of bytes.

[3] This field should be ignored by those application contexts providing operations class 4 service.

[4] If the number outstanding connect indications equals *CONIND_number*, the MAP provider need not discard further incoming connect indications, but may chose to queue them internally until the number of outstanding connect indications drops below CONNIND_number.

— **Successful:** Correct acknowledgement of the primitive is indicated with the `MAP_BIND_ACK` primitive described in Section 4.1.4 [Bind Application Context Acknowledgement], page 36.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TBADADDR` This indicates that the protocol address was in an incorrect format or the address contained illegal information. It is not indended to indicate protocol errors.

`TNOADDR` This indicates that the MAP provider could not allocate an address.

`TACCES` This indicates that the user did not have proper permissions for the use of the requested address.

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

`TADDRBUSY` This indicated that the requested address is already in use.

4.1.4 Bind Application Context Acknowledgement

MAP_BIND_ACK

This primitive indicates to the MAP user that the specified protoocl address has been bound to the *stream*, that the specified number of dialogue open indications are allowed to be queued by the MAP provider for th specified protocol address, and that the *stream* associated with the specified protocol address has ben activated.

Format

The format of the message is one '`M_PCPROTO`' message block. The format of the '`M_PCPROTO`' message block is as follows:

```
typedef struct {
    long PRIM_type;         /* always MAP_BIND_ACK */
    long ADDR_length;       /* length of address -- see note in sec. 1.4 */
    long ADDR_offset;       /* offset of address */
    ulong DIAIND_number;    /* dialogue open indications to be queued */
} MAP_bind_ack_t;
```

Parameters

PRIM_type
ADDR_length
ADDR_offset
DIAIND_number

Flags

Valid States

New State

Rules

The following rules apply to the binding of the specified protocol address to the *stream*:

— If the *ADDR_length*

4.1.5 Unbind Application Context Request

MAP_UNBIND_REQ

This primitive requests that the MAP provider unbind the protocol address associated with the *stream* and deactivate the *stream*.

Format

The format of the message is one 'M_PROTO' message block. The format of the 'M_PROTO' mesage block is as follows:

```
typedef struct {
    long PRIM_type;         /* always MAP_UNBIND_REQ */
} MAP_unbind_req_t;
```

Parameters

PRIM_type:
Indicates primitive type.

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate the following acknowledgements upon receipt of the primitive and that the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Successful:** Correct acknowledgement of the primitive is indicated with the MAP_OK_ACK primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the MAP_ERROR_ACK primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

TOUTSTATE The primitive would place the MAP interface out of state.

TSYSERR A system error has occurred and the UNIX system error is indicated in the primitive.

4.1.6 Options Management Request

MAP_OPTMGMT_REQ

This primitive allows the MAP user to manage the options associated with the *stream*.

Format

The format of the message is one 'M_PROTO' message block. The format of the 'M_PROTO' message block is as follows:

```
typedef struct {
    long PRIM_type;         /* always MAP_OPTMGMT_REQ */
    long OPT_length;        /* options length */
    long OPT_offset;        /* options offset */
    long MGMT_flags;        /* flags */
} MAP_optmgmt_req_t;
```

Parameters

PRIM_type
Indicates the primitive type.

OPT_length
The length of the protocol options associated with the primitive.

OPT_offset The offset from the beginning of the 'M_PROTO' message block where the options begin. The proper alignment of options is not guaranteed. The options are, however, aligned the same as they were received from the MAP user.

MGMT_flags
The flags which define the request made by the MAP user.

Flags

T_NEGOTIATE
Negotiate and set the options with the MAP provider.

T_CHECK Check the validity of the specified options.

T_CURRENT Return the options currently in effect.

T_DEFAULT Return the default options.

Rules

For the rules governing the requests made by this primitive, see the MAP_OPTMGMT_ACK primitive described in Section 4.1.9 [Options Management Acknowledgement], page 42.

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledgements upon receipt of the primitive and that the MAP user await the acknowledgement before issuing any othe primitives:

— **Successful:** Acknowledgement of the primitive with the MAP_OPTMGMT_ACK primitive described in Section 4.1.9 [Options Management Acknowledgement], page 42.

— **Non-fatal errors:** These errors will be indicated with the MAP_ERROR_ACK primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

TACCES — This indicates that the user did not have proper permissions for the use of the requested options.

TOUTSTATE — The primitive would place the MAP interface out of state.

TBADOPT — This indicates that the options as specified were in an incorrect format, or they contained illegal information.

TBADFLAG — This indicates that the flags as specified were incorrect or illegal.

TSYSERR — A system error has occured and the UNIX system error is indicated in the primitive.

TNOTSUPPORT — This MAP provider does not support the requested flag (T_CHECK or T_CURRENT).

4.1.7 Get Protocol Address Request

MAP_ADDR_REQ

This primitive requests that the MAP provider return the local protocol address that is bound to the *stream* and the address of the remote MAP entity if a dialogue has been established.

Format

The format of the message is one 'M_PROTO' message block. The format of the 'M_PROTO' message block is as follows:

```
typedef struct {
    long PRIM_type;     /* always MAP_ADDR_REQ */
} MAP_addr_req_t;
```

Parameters

PRIM_type
Indicates the primitive type.

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Successful:** Correct acknowledgement of the primitive is indicated with the MAP_ADDR_ACK primitive described in Section 4.1.8 [Get Protocol Address Acknowledgement], page 41.

— **Non-fatal errors:** There are no errors associated with the issuance of this primitive.

4.1.8 Get Protocol Address Acknowledgement

MAP_ADDR_ACK

Format

Parameters

Flags

Valid States

New State

4.1.9 Options Management Acknowledgement

MAP_OPTMGMT_ACK

Format

Parameters

Flags

Valid States

New State

4.1.10 Error Acknowledgement

MAP_ERROR_ACK

Format

Parameters

Valid Error Codes

Valid States

New State

4.1.11 Successful Receipt Acknowledgement

MAP_OK_ACK

Format

Parameters

Valid States

New State

4.2 Association Establishment Phase

4.2.1 Association Establishment Request

MAP_OPEN_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Successful Association Establishment:**
- **Unsuccessful Association Establishment:**
- **Non-fatal errors:**

4.2.2 Association Establishment Indication

MAP_OPEN_IND

Format

Parameters

Flags

Valid States

New State

4.2.3 Association Establishment Response

MAP_OPEN_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.2.4 Association Establishment Confirmation

MAP_OPEN_CON

Format

Parameters

Flags

Valid States

New State

4.3 Component Transfer Phase

4.3.1 Component Delimitation Request

MAP_DELIM_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.3.2 Component Delimitation Indication

MAP_DELIM_IND

Format

Parameters

Flags

Valid States

New State

4.4 Association Termination Phase

4.4.1 Association Release Request

MAP_CLOSE_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.4.2 Association Release Indication

MAP_CLOSE_IND

Format

Parameters

Flags

Valid States

New State

4.4.3 Association Abort Request

MAP_ABORT_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.4.4 Association User Abort Indication

MAP_UABORT_IND

Format

Parameters

Flags

Valid States

New State

4.4.5 Association Provider Abort Indication

MAP_PABORT_IND

Format

Parameters

Flags

Valid States

New State

4.4.6 Association Notice Indication

MAP_NOTICE_IND

Format

Parameters

Flags

Valid States

New State

4.5 Secure Transport Primitives

4.5.1 Secure Transport Class 1 Request

MAP_STC1_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.5.2 Secure Transport Class 1 Indication

MAP_STC1_IND

Format

Parameters

Flags

Valid States

New State

4.5.3 Secure Transport Class 2 Request

MAP_STC2_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.5.4 Secure Transport Class 2 Indication

MAP_STC2_IND

Format

Parameters

Flags

Valid States

New State

4.5.5 Secure Transport Class 3 Request

MAP_STC3_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.5.6 Secure Transport Class 3 Indication

MAP_STC3_IND

Format

Parameters

Flags

Valid States

New State

4.5.7 Secure Transport Class 4 Request

MAP_STC4_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.5.8 Secure Transport Class 4 Indication

MAP_STC4_IND

Format

Parameters

Flags

Valid States

New State

4.6 Service Specific Primitives

4.6.1 Activate Supplementary Service Request

MAP_ACTSS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.2 Activate Supplementary Service Indication

MAP_ACTSS_IND

Format

Parameters

Flags

Valid States

New State

4.6.3 Activate Supplementary Service Response

MAP_ACTSS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.4 Activate Supplementary Service Confirmation

MAP_ACTSS_CON

Format

Parameters

Flags

Valid States

New State

4.6.5 Activate Traceec Mode Request

MAP_ACTTM_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.6 Activate Traceec Mode Indication

MAP_ACTTM_IND

Format

Parameters

Flags

Valid States

New State

4.6.7 Activate Traceec Mode Response

MAP_ACTTM_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.8 Activate Traceec Mode Confirmation

MAP_ACTTM_CON

Format

Parameters

Flags

Valid States

New State

4.6.9 Alert Service Center Request

MAP_ALERTSC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.10 Alert Service Center Indication

MAP_ALERTSC_IND

Format

Parameters

Flags

Valid States

New State

4.6.11 Alert Service Center Response

MAP_ALERTSC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.12 Alert Service Center Confirmation

MAP_ALERTSC_CON

Format

Parameters

Flags

Valid States

New State

4.6.13 Allocate Handover Number Request

MAP_ALLOCHN_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.14 Allocate Handover Number Indication

MAP_ALLOCHN_IND

Format

Parameters

Flags

Valid States

New State

4.6.15 Allocate Handover Number Response

MAP_ALLOCHN_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.16 Allocate Handover Number Confirmation

MAP_ALLOCHN_CON

Format

Parameters

Flags

Valid States

New State

4.6.17 Any Time Modification Request

MAP_ANYMOD_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.18 Any Time Modification Indication

MAP_ANYMOD_IND

Format

Parameters

Flags

Valid States

New State

4.6.19 Any Time Modification Response

MAP_ANYMOD_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.20 Any Time Modification Confirmation

MAP_ANYMOD_CON

Format

Parameters

Flags

Valid States

New State

4.6.21 Any Time Interrogation Request

MAP_ANYQRY_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.22 Any Time Interrogation Indication

MAP_ANYQRY_IND

Format

Parameters

Flags

Valid States

New State

4.6.23 Any Time Interrogation Response

MAP_ANYQRY_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.24 Any Time Interrogation Confirmation

MAP_ANYQRY_CON

Format

Parameters

Flags

Valid States

New State

4.6.25 Any Time Subscription Interrogation Request

MAP_ANYSUB_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.26 Any Time Subscription Interrogation Indication

MAP_ANYSUB_IND

Format

Parameters

Flags

Valid States

New State

4.6.27 Any Time Subscription Interrogation Response

MAP_ANYSUB_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.28 Any Time Subscription Interrogation Confirmation

MAP_ANYSUB_CON

Format

Parameters

Flags

Valid States

New State

4.6.29 Authentication Failure Report Request

MAP_AUTHFAIL_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.30 Authentication Failure Report Indication

MAP_AUTHFAIL_IND

Format

Parameters

Flags

Valid States

New State

4.6.31 Authentication Failure Report Response

MAP_AUTHFAIL_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.32 Authentication Failure Report Confirmation

MAP_AUTHFAIL_CON

Format

Parameters

Flags

Valid States

New State

4.6.33 Authenticate Request

MAP_AUTH_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.34 Authenticate Indication

MAP_AUTH_IND

Format

Parameters

Flags

Valid States

New State

4.6.35 Authenticate Response

MAP_AUTH_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.36 Authenticate Confirmation

MAP_AUTH_CON

Format

Parameters

Flags

Valid States

New State

4.6.37 Cancel Location Request

MAP_CANCLOC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.38 Cancel Location Indication

MAP_CANCLOC_IND

Format

Parameters

Flags

Valid States

New State

4.6.39 Cancel Location Response

MAP_CANCLOC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.40 Cancel Location Confirmation

MAP_CANCLOC_CON

Format

Parameters

Flags

Valid States

New State

4.6.41 Reset Request

MAP_RESET_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.42 Reset Indication

MAP_RESET_IND

Format

Parameters

Flags

Valid States

New State

4.6.43 Reset Response

MAP_RESET_CON

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.44 Reset Confirmation

MAP_RESET_RES

Format

Parameters

Flags

Valid States

New State

4.6.45 Restore Request

MAP_RESTORE_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.46 Restore Indication

MAP_RESTORE_IND

Format

Parameters

Flags

Valid States

New State

4.6.47 Restore Response

MAP_RESTORE_CON

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.48 Restore Confirmation

MAP_RESTORE_RES

Format

Parameters

Flags

Valid States

New State

4.6.49 Resume Call Handling Request

MAP_RESUME_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.50 Resume Call Handling Indication

MAP_RESUME_IND

Format

Parameters

Flags

Valid States

New State

4.6.51 Resume Call Handling Confirmation

MAP_RESUME_RES

Format

Parameters

Flags

Valid States

New State

4.6.52 Resume Call Handling Response

MAP_RESUME_CON

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.53 Search for MS Request

MAP_SEARCH_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.54 Search for MS Indication

MAP_SEARCH_IND

Format

Parameters

Flags

Valid States

New State

4.6.55 Search for MS Confirmation

MAP_SEARCH_RES

Format

Parameters

Flags

Valid States

New State

4.6.56 Search for MS Response

MAP_SEARCH_CON

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.57 Check IEMI Request

MAP_CKIEMI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.58 Check IEMI Indication

MAP_CKIEMI_IND

Format

Parameters

Flags

Valid States

New State

4.6.59 Check IEMI Response

MAP_CKIEMI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.60 Check IEMI Confirmation

MAP_CKIEMI_CON

Format

Parameters

Flags

Valid States

New State

4.6.61 Deactivate Supplementary Service Request

MAP_DEACTSS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.62 Deactivate Supplementary Service Indication

MAP_DEACTSS_IND

Format

Parameters

Flags

Valid States

New State

4.6.63 Deactivate Supplementary Service Response

MAP_DEACTSS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.64 Deactivate Supplementary Service Confirmation

MAP_DEACTSS_CON

Format

Parameters

Flags

Valid States

New State

4.6.65 Deactivate Trace Mode Request

MAP_DEACTTM_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.66 Deactivate Trace Mode Indication

MAP_DEACTTM_IND

Format

Parameters

Flags

Valid States

New State

4.6.67 Deactivate Trace Mode Response

MAP_DEACTTM_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.68 Deactivate Trace Mode Confirmation

MAP_DEACTTM_CON

Format

Parameters

Flags

Valid States

New State

4.6.69 Delete Subscriber Data Request

MAP_DELSUBD_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.70 Delete Subscriber Data Indication

MAP_DELSUBD_IND

Format

Parameters

Flags

Valid States

New State

4.6.71 Delete Subscriber Data Response

MAP_DELSUBD_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.72 Delete Subscriber Data Confirmation

MAP_DELSUBD_CON

Format

Parameters

Flags

Valid States

New State

4.6.73 Erase CC Entry Request

MAP_ERASECC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.74 Erase CC Entry Indication

MAP_ERASECC_IND

Format

Parameters

Flags

Valid States

New State

4.6.75 Erase CC Entry Response

MAP_ERASECC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.76 Erase CC Entry Confirmation

MAP_ERASECC_CON

Format

Parameters

Flags

Valid States

New State

4.6.77 Erase Supplementatry Service Request

MAP_ERASESS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.78 Erase Supplementatry Service Indication

MAP_ERASESS_IND

Format

Parameters

Flags

Valid States

New State

4.6.79 Erase Supplementatry Service Response

MAP_ERASESS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.80 Erase Supplementatry Service Confirmation

MAP_ERASESS_CON

Format

Parameters

Flags

Valid States

New State

4.6.81 Failure Report Request

MAP_FAILURE_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.82 Failure Report Indication

MAP_FAILURE_IND

Format

Parameters

Flags

Valid States

New State

4.6.83 Failure Report Response

MAP_FAILURE_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.84 Failure Report Confirmation

MAP_FAILURE_CON

Format

Parameters

Flags

Valid States

New State

4.6.85 Forward Access Signalling Request

MAP_FAS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.86 Forward Access Signalling Indication

MAP_FAS_IND

Format

Parameters

Flags

Valid States

New State

4.6.87 Forward Access Signalling Response

MAP_FAS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.88 Forward Access Signalling Confirmation

MAP_FAS_CON

Format

Parameters

Flags

Valid States

New State

4.6.89 Forward Check SS Indication Request

MAP_FCKSSIND_REQ

Format

Parameters

Flags

Valid States

New State

4.6.90 Forward Check SS Indication Indication

MAP_FCKSSIND_IND

Format

Parameters

Flags

Valid States

New State

4.6.91 Forward Check SS Indication Response

MAP_FCKSSIND_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.92 Forward Check SS Indication Confirmation

MAP_FCKSSIND_CON

Format

Parameters

Flags

Valid States

New State

4.6.93 Forward Group Call Signalling Request

MAP_FGRCSIG_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.94 Forward Group Call Signalling Indication

MAP_FGRCSIG_IND

Format

Parameters

Flags

Valid States

New State

4.6.95 Forward Group Call Signalling Response

MAP_FGRCSIG_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.96 Forward Group Call Signalling Confirmation

MAP_FGRCSIG_CON

Format

Parameters

Flags

Valid States

New State

4.6.97 Forward New IMSI Request

MAP_FNEWIMSI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.98 Forward New IMSI Indication

MAP_FNEWIMSI_IND

Format

Parameters

Flags

Valid States

New State

4.6.99 Forward New IMSI Response

MAP_FNEWIMSI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.100 Forward New IMSI Confirmation

MAP_FNEWIMSI_CON

Format

Parameters

Flags

Valid States

New State

4.6.101 Obtain IEMI Request

MAP_GETIEMI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.102 Obtain IEMI Indication

MAP_GETIEMI_IND

Format

Parameters

Flags

Valid States

New State

4.6.103 Obtain IEMI Response

MAP_GETIEMI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.104 Obtain IEMI Confirmation

MAP_GETIEMI_CON

Format

Parameters

Flags

Valid States

New State

4.6.105 Get Password Request

MAP_GETPASS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.106 Get Password Indication

MAP_GETPASS_IND

Format

Parameters

Flags

Valid States

New State

4.6.107 Get Password Response

MAP_GETPASS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.108 Get Password Confirmation

MAP_GETPASS_CON

Format

Parameters

Flags

Valid States

New State

4.6.109 Inform Service Center Request

MAP_INFORMSC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.110 Inform Service Center Indication

MAP_INFORMSC_IND

Format

Parameters

Flags

Valid States

New State

4.6.111 Inform Service Center Response

MAP_INFORMSC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.112 Inform Service Center Confirmation

MAP_INFORMSC_CON

Format

Parameters

Flags

Valid States

New State

4.6.113 Insert Subscriber Data Request

MAP_INSSUBSD_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.114 Insert Subscriber Data Indication

MAP_INSSUBSD_IND

Format

Parameters

Flags

Valid States

New State

4.6.115 Insert Subscriber Data Response

MAP_INSSUBSD_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.116 Insert Subscriber Data Confirmation

MAP_INSSUBSD_CON

Format

Parameters

Flags

Valid States

New State

4.6.117 Iterrogate Supplementary Service Request

MAP_INTERGSS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.118 Iterrogate Supplementary Service Indication

MAP_INTERGSS_IND

Format

Parameters

Flags

Valid States

New State

4.6.119 Iterrogate Supplementary Service Response

MAP_INTERGSS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.120 Iterrogate Supplementary Service Confirmation

MAP_INTERGSS_CON

Format

Parameters

Flags

Valid States

New State

4.6.121 IST Alert Request

MAP_ISTALERT_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.122 IST Alert Indication

MAP_ISTALERT_IND

Format

Parameters

Flags

Valid States

New State

4.6.123 IST Alert Response

MAP_ISTALERT_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.124 IST Alert Confirmation

MAP_ISTALERT_CON

Format

Parameters

Flags

Valid States

New State

4.6.125 IST Command Request

MAP_ISTCMD_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.126 IST Command Indication

MAP_ISTCMD_IND

Format

Parameters

Flags

Valid States

New State

4.6.127 IST Command Response

MAP_ISTCMD_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.128 IST Command Confirmation

MAP_ISTCMD_CON

Format

Parameters

Flags

Valid States

New State

4.6.129 Note MM Event Request

MAP_MMEVENT_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.130 Note MM Event Indication

MAP_MMEVENT_IND

Format

Parameters

Flags

Valid States

New State

4.6.131 Note MM Event Response

MAP_MMEVENT_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.132 Note MM Event Confirmation

MAP_MMEVENT_CON

Format

Parameters

Flags

Valid States

New State

4.6.133 Mobile Originated Forward Short Message Request

MAP_MOFSMS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.134 Mobile Originated Forward Short Message Indication

MAP_MOFSMS_IND

Format

Parameters

Flags

Valid States

New State

4.6.135 Mobile Originated Forward Short Message Response

MAP_MOFSMS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.136 Mobile Originated Forward Short Message Confirmation

MAP_MOFSMS_CON

Format

Parameters

Flags

Valid States

New State

4.6.137 Note MS Present for GRPS Request

MAP_MSGRPSPRES_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.138 Note MS Present for GRPS Indication

MAP_MSGRPSPRES_IND

Format

Parameters

Flags

Valid States

New State

4.6.139 Note MS Present for GRPS Response

MAP_MSGRPSPRES_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.140 Note MS Present for GRPS Confirmation

MAP_MSGRPSPRES_CON

Format

Parameters

Flags

Valid States

New State

4.6.141 Mobile Terminated Forward Short Message Request

MAP_MTFSMS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.142 Mobile Terminated Forward Short Message Indication

MAP_MTFSMS_IND

Format

Parameters

Flags

Valid States

New State

4.6.143 Mobile Terminated Forward Short Message Response

MAP_MTFSMS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.144 Mobile Terminated Forward Short Message Confirmation

MAP_MTFSMS_CON

Format

Parameters

Flags

Valid States

New State

4.6.145 Page Request

MAP_PAGE_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.146 Page Indication

MAP_PAGE_IND

Format

Parameters

Flags

Valid States

New State

4.6.147 Page Response

MAP_PAGE_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.148 Page Confirmation

MAP_PAGE_CON

Format

Parameters

Flags

Valid States

New State

4.6.149 Prepare Group Call Request

MAP_PREPGC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.150 Prepare Group Call Indication

MAP_PREPGC_IND

Format

Parameters

Flags

Valid States

New State

4.6.151 Prepare Group Call Response

MAP_PREPGC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.152 Prepare Group Call Confirmation

MAP_PREPGC_CON

Format

Parameters

Flags

Valid States

New State

4.6.153 Prepare Handover Request

MAP_PREPHO_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.154 Prepare Handover Indication

MAP_PREPHO_IND

Format

Parameters

Flags

Valid States

New State

4.6.155 Prepare Handover Response

MAP_PREPHO_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE`	The primitive would place the MAP interface out of state.
`TSYSERR`	A system error has occured and the UNIX System error in indicated in the primitive.

4.6.156 Prepare Handover Confirmation

MAP_PREPHO_CON

Format

Parameters

Flags

Valid States

New State

4.6.157 Prepare Subsequent Handover Request

MAP_PREPSH_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.158 Prepare Subsequent Handover Indication

MAP_PREPSH_IND

Format

Parameters

Flags

Valid States

New State

4.6.159 Prepare Subsequent Handover Response

MAP_PREPSH_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.160 Prepare Subsequent Handover Confirmation

MAP_PREPSH_CON

Format

Parameters

Flags

Valid States

New State

4.6.161 Process Access Signalling Request

MAP_PROCAS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.162 Process Access Signalling Indication

MAP_PROCAS_IND

Format

Parameters

Flags

Valid States

New State

4.6.163 Process Access Signalling Response

MAP_PROCAS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.164 Process Access Signalling Confirmation

MAP_PROCAS_CON

Format

Parameters

Flags

Valid States

New State

4.6.165 Process Group Call Signalling Request

MAP_PROCGC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.166 Process Group Call Signalling Indication

MAP_PROCGC_IND

Format

Parameters

Flags

Valid States

New State

4.6.167 Process Group Call Signalling Response

MAP_PROCGC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.168 Process Group Call Signalling Confirmation

MAP_PROCGC_CON

Format

Parameters

Flags

Valid States

New State

4.6.169 Process Unstructured SS Request

MAP_PROCUSS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.170 Process Unstructured SS Indication

MAP_PROCUSS_IND

Format

Parameters

Flags

Valid States

New State

4.6.171 Process Unstructured SS Response

MAP_PROCUSS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.172 Process Unstructured SS Confirmation

MAP_PROCUSS_CON

Format

Parameters

Flags

Valid States

New State

4.6.173 Provide IMSI Request

MAP_PROVIMSI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.174 Provide IMSI Indication

MAP_PROVIMSI_IND

Format

Parameters

Flags

Valid States

New State

4.6.175 Provide IMSI Response

MAP_PROVIMSI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.176 Provide IMSI Confirmation

MAP_PROVIMSI_CON

Format

Parameters

Flags

Valid States

New State

4.6.177 Provide Roaming Number Request

MAP_PROVRN_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.178 Provide Roaming Number Indication

MAP_PROVRN_IND

Format

Parameters

Flags

Valid States

New State

4.6.179 Provide Roaming Number Response

MAP_PROVRN_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.180 Provide Roaming Number Confirmation

MAP_PROVRN_CON

Format

Parameters

Flags

Valid States

New State

4.6.181 Provide Subscriber Location Request

MAP_PROVSLOC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.182 Provide Subscriber Location Indication

MAP_PROVSLOC_IND

Format

Parameters

Flags

Valid States

New State

4.6.183 Provide Subscriber Location Response

MAP_PROVSLOC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.184 Provide Subscriber Location Confirmation

MAP_PROVSLOC_CON

Format

Parameters

Flags

Valid States

New State

4.6.185 Provide Subscriber Information Request

MAP_PROVSUBI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.186 Provide Subscriber Information Indication

MAP_PROVSUBI_IND

Format

Parameters

Flags

Valid States

New State

4.6.187 Provide Subscriber Information Response

MAP_PROVSUBI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the MAP_OK_ACK primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the MAP_ERROR_ACK primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

TOUTSTATE The primitive would place the MAP interface out of state.

TSYSERR A system error has occured and the UNIX System error in indicated in the primitive.

4.6.188 Provide Subscriber Information Confirmation

MAP_PROVSUBI_CON

Format

Parameters

Flags

Valid States

New State

4.6.189 Purge MS Request

MAP_PURGEMS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.190 Purge MS Indication

MAP_PURGEMS_IND

Format

Parameters

Flags

Valid States

New State

4.6.191 Purge MS Response

MAP_PURGEMS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.192 Purge MS Confirmation

MAP_PURGEMS_CON

Format

Parameters

Flags

Valid States

New State

4.6.193 Ready for SM Request

MAP_RDYSM_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.194 Ready for SM Indication

MAP_RDYSM_IND

Format

Parameters

Flags

Valid States

New State

4.6.195 Ready for SM Response

MAP_RDYSM_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.196 Ready for SM Confirmation

MAP_RDYSM_CON

Format

Parameters

Flags

Valid States

New State

4.6.197 Register CC Entry Request

MAP_REGCC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.198 Register CC Entry Indication

MAP_REGCC_IND

Format

Parameters

Flags

Valid States

New State

4.6.199 Register CC Entry Response

MAP_REGCC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.200 Register CC Entry Confirmation

MAP_REGCC_CON

Format

Parameters

Flags

Valid States

New State

4.6.201 Register Password Request

MAP_REGPW_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.202 Register Password Indication

MAP_REGPW_IND

Format

Parameters

Flags

Valid States

New State

4.6.203 Register Password Response

MAP_REGPW_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.204 Register Password Confirmation

MAP_REGPW_CON

Format

Parameters

Flags

Valid States

New State

4.6.205 Register Supplementary Service Request

MAP_REGSS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.206 Register Supplementary Service Indication

MAP_REGSS_IND

Format

Parameters

Flags

Valid States

New State

4.6.207 Register Supplementary Service Response

MAP_REGSS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.208 Register Supplementary Service Confirmation

MAP_REGSS_CON

Format

Parameters

Flags

Valid States

New State

4.6.209 Remote User Free Request

MAP_REMUSRFREE_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.210 Remote User Free Indication

MAP_REMUSRFREE_IND

Format

Parameters

Flags

Valid States

New State

4.6.211 Remote User Free Response

MAP_REMUSRFREE_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.212 Remote User Free Confirmation

MAP_REMUSRFREE_CON

Format

Parameters

Flags

Valid States

New State

4.6.213 Send Identification Request

MAP_SENDID_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.214 Send Identification Indication

MAP_SENDID_IND

Format

Parameters

Flags

Valid States

New State

4.6.215 Send Identification Response

MAP_SENDID_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.216 Send Identification Confirmation

MAP_SENDID_CON

Format

Parameters

Flags

Valid States

New State

4.6.217 Set Ciphering Mode Request

MAP_SETCM_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.218 Set Ciphering Mode Indication

MAP_SETCM_IND

Format

Parameters

Flags

Valid States

New State

4.6.219 Set Ciphering Mode Response

MAP_SETCM_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.220 Set Ciphering Mode Confirmation

MAP_SETCM_CON

Format

Parameters

Flags

Valid States

New State

4.6.221 Set Reporting State Request

MAP_SETREP_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.222 Set Reporting State Indication

MAP_SETREP_IND

Format

Parameters

Flags

Valid States

New State

4.6.223 Set Reporting State Response

MAP_SETREP_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.224 Set Reporting State Confirmation

MAP_SETREP_CON

Format

Parameters

Flags

Valid States

New State

4.6.225 Report Short Message Delivery Status Request

MAP_SMDELIVSTAT_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE`	The primitive would place the MAP interface out of state.
`TSYSERR`	A system error has occured and the UNIX System error in indicated in the primitive.

4.6.226 Report Short Message Delivery Status Indication

MAP_SMDELIVSTAT_IND

Format

Parameters

Flags

Valid States

New State

4.6.227 Report Short Message Delivery Status Response

MAP_SMDELIVSTAT_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.228 Report Short Message Delivery Status Confirmation

MAP_SMDELIVSTAT_CON

Format

Parameters

Flags

Valid States

New State

4.6.229 Send Authentication Information Request

MAP_SENDAUTHI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.230 Send Authentication Information Indication

MAP_SENDAUTHI_IND

Format

Parameters

Flags

Valid States

New State

4.6.231 Send Authentication Information Response

MAP_SENDAUTHI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.232 Send Authentication Information Confirmation

MAP_SENDAUTHI_CON

Format

Parameters

Flags

Valid States

New State

4.6.233 Send End Signal Request

MAP_SENDEND_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.234 Send End Signal Indication

MAP_SENDEND_IND

Format

Parameters

Flags

Valid States

New State

4.6.235 Send End Signal Response

MAP_SENDEND_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.236 Send End Signal Confirmation

MAP_SENDEND_CON

Format

Parameters

Flags

Valid States

New State

4.6.237 Send Group Call End Signal Request

MAP_SENDGCEND_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.238 Send Group Call End Signal Indication

MAP_SENDGCEND_IND

Format

Parameters

Flags

Valid States

New State

4.6.239 Send Group Call End Signal Response

MAP_SENDGCEND_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.240 Send Group Call End Signal Confirmation

MAP_SENDGCEND_CON

Format

Parameters

Flags

Valid States

New State

4.6.241 Send Routing Information for GPRS Request

MAP_SENDGPRSRI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.242 Send Routing Information for GPRS Indication

MAP_SENDGPRSRI_IND

Format

Parameters

Flags

Valid States

New State

4.6.243 Send Routing Information for GPRS Response

MAP_SENDGPRSRI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.244 Send Routing Information for GPRS Confirmation

MAP_SENDGPRSRI_CON

Format

Parameters

Flags

Valid States

New State

4.6.245 Send Handover Report Request

MAP_SENDHOREP_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.246 Send Handover Report Indication

MAP_SENDHOREP_IND

Format

Parameters

Flags

Valid States

New State

4.6.247 Send Handover Report Response

MAP_SENDHOREP_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.248 Send Handover Report Confirmation

MAP_SENDHOREP_CON

Format

Parameters

Flags

Valid States

New State

4.6.249 Send IMSI Request

MAP_SENDIMSI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.250 Send IMSI Indication

MAP_SENDIMSI_IND

Format

Parameters

Flags

Valid States

New State

4.6.251 Send IMSI Response

MAP_SENDIMSI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.252 Send IMSI Confirmation

MAP_SENDIMSI_CON

Format

Parameters

Flags

Valid States

New State

4.6.253 Send Routing Information for LCS Request

MAP_SENDLCSRI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.254 Send Routing Information for LCS Indication

MAP_SENDLCSRI_IND

Format

Parameters

Flags

Valid States

New State

4.6.255 Send Routing Information for LCS Response

MAP_SENDLCSRI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.256 Send Routing Information for LCS Confirmation

MAP_SENDLCSRI_CON

Format

Parameters

Flags

Valid States

New State

4.6.257 Send Mobile Originated SMS Information Request

MAP_SENDMOSMSI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.258 Send Mobile Originated SMS Information Indication

MAP_SENDMOSMSI_IND

Format

Parameters

Flags

Valid States

New State

4.6.259 Send Mobile Originated SMS Information Response

MAP_SENDMOSMSI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.260 Send Mobile Originated SMS Information Confirmation

MAP_SENDMOSMSI_CON

Format

Parameters

Flags

Valid States

New State

4.6.261 Send Mobile Terminated SMS Information Request

MAP_SENDMTSMSI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.262 Send Mobile Terminated SMS Information Indication

MAP_SENDMTSMSI_IND

Format

Parameters

Flags

Valid States

New State

4.6.263 Send Mobile Terminated SMS Information Response

MAP_SENDMTSMSI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.264 Send Mobile Terminated SMS Information Confirmation

MAP_SENDMTSMSI_CON

Format

Parameters

Flags

Valid States

New State

4.6.265 Send Routing Information Request

MAP_SENDRI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.266 Send Routing Information Indication

MAP_SENDRI_IND

Format

Parameters

Flags

Valid States

New State

4.6.267 Send Routing Information Response

MAP_SENDRI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.268 Send Routing Information Confirmation

MAP_SENDRI_CON

Format

Parameters

Flags

Valid States

New State

4.6.269 Send Routing Information for SM Request

MAP_SENDSMSRI_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.270 Send Routing Information for SM Indication

MAP_SENDSMSRI_IND

Format

Parameters

Flags

Valid States

New State

4.6.271 Send Routing Information for SM Response

MAP_SENDSMSRI_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.272 Send Routing Information for SM Confirmation

MAP_SENDSMSRI_CON

Format

Parameters

Flags

Valid States

New State

4.6.273 SS Invocation Notification Request

MAP_SSINV_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.274 SS Invocation Notification Indication

MAP_SSINV_IND

Format

Parameters

Flags

Valid States

New State

4.6.275 SS Invocation Notification Response

MAP_SSINV_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.276 SS Invocation Notification Confirmation

MAP_SSINV_CON

Format

Parameters

Flags

Valid States

New State

4.6.277 Status Report Request

MAP_STATUS_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.278 Status Report Indication

MAP_STATUS_IND

Format

Parameters

Flags

Valid States

New State

4.6.279 Status Report Response

MAP_STATUS_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.280 Status Report Confirmation

MAP_STATUS_CON

Format

Parameters

Flags

Valid States

New State

4.6.281 Subscriber Location Report Request

MAP_SUBLOCREP_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.282 Subscriber Location Report Indication

MAP_SUBLOCREP_IND

Format

Parameters

Flags

Valid States

New State

4.6.283 Subscriber Location Report Response

MAP_SUBLOCREP_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.284 Subscriber Location Report Confirmation

MAP_SUBLOCREP_CON

Format

Parameters

Flags

Valid States

New State

4.6.285 Note Subscriber Data Modified Request

MAP_SUBSDATAMOD_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.286 Note Subscriber Data Modified Indication

MAP_SUBSDATAMOD_IND

Format

Parameters

Flags

Valid States

New State

4.6.287 Note Subscriber Data Modified Response

MAP_SUBSDATAMOD_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.288 Note Subscriber Data Modified Confirmation

MAP_SUBSDATAMOD_CON

Format

Parameters

Flags

Valid States

New State

4.6.289 Trace Subscriber Activity Request

MAP_TRACESA_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.290 Trace Subscriber Activity Indication

MAP_TRACESA_IND

Format

Parameters

Flags

Valid States

New State

4.6.291 Trace Subscriber Activity Response

MAP_TRACESA_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.292 Trace Subscriber Activity Confirmation

MAP_TRACESA_CON

Format

Parameters

Flags

Valid States

New State

4.6.293 Update GPRS Location Request

MAP_UDGRPSLOC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.294 Update GPRS Location Indication

MAP_UDGRPSLOC_IND

Format

Parameters

Flags

Valid States

New State

4.6.295 Update GPRS Location Response

MAP_UDGRPSLOC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.296 Update GPRS Location Confirmation

MAP_UDGRPSLOC_CON

Format

Parameters

Flags

Valid States

New State

4.6.297 Update Location Request

MAP_UDLOC_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE` The primitive would place the MAP interface out of state.

`TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.298 Update Location Indication

MAP_UDLOC_IND

Format

Parameters

Flags

Valid States

New State

4.6.299 Update Location Response

MAP_UDLOC_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.300 Update Location Confirmation

MAP_UDLOC_CON

Format

Parameters

Flags

Valid States

New State

4.6.301 Unstructured Supplementary Service Notify Request

MAP_USSDNTFY_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

— **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.

— **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

`TOUTSTATE`	The primitive would place the MAP interface out of state.
`TSYSERR`	A system error has occured and the UNIX System error in indicated in the primitive.

4.6.302 Unstructured Supplementary Service Notify Indication

MAP_USSDNTFY_IND

Format

Parameters

Flags

Valid States

New State

4.6.303 Unstructured Supplementary Service Notify Response

MAP_USSDNTFY_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.304 Unstructured Supplementary Service Notify Confirmation

MAP_USSDNTFY_CON

Format

Parameters

Flags

Valid States

New State

4.6.305 Unstructured Supplementary Service Request

MAP_USSDREQ_REQ

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.306 Unstructured Supplementary Service Indication

MAP_USSDREQ_IND

Format

Parameters

Flags

Valid States

New State

4.6.307 Unstructured Supplementary Service Response

MAP_USSDREQ_RES

Format

Parameters

Flags

Valid States

New State

Acknowledgements

This primitive requires the MAP provider to generate one of the following acknowledegements upon receipt of the primitive, and the MAP user must wait for the acknowledgement before issuing any other primitives:

- **Success:** Acknowledgement of the primitive with the `MAP_OK_ACK` primitive described in Section 4.1.11 [Successful Receipt Acknowledgement], page 44.
- **Non-fatal errors:** These errors will be indicated with the `MAP_ERROR_ACK` primitive described in Section 4.1.10 [Error Acknowledgement], page 43. The allowable errors are as follows:

  `TOUTSTATE` The primitive would place the MAP interface out of state.

  `TSYSERR` A system error has occured and the UNIX System error in indicated in the primitive.

4.6.308 Unstructured Supplementary Service Confirmation

MAP_USSDREQ_CON

Format

Parameters

Flags

Valid States

New State

5 Diagnostics Requirements

Two error handling facilities should be provided to the MAP service user: one to handle non-fatal errors, and the other to handle fatal errors.

5.1 Non-Fatal Error Handling Facility

These are errors that do not change the state of the MAP service interface as seen by the MAP service user and provide the user with the option of reissuing the MAP primitive with the corrected options specification. The non-fatal error handling is provided only to those primitives that require acknowledgements, and uses the `MAP_ERROR_ACK` to report these errors. These errors retain the state of the MAP service interface the same as it was before the MAP provider received the primitive that was in error. Syntax errors and rule violations are reported via the non-fatal error handling facility.

5.2 Fatal Error Handling Facility

These errors are issued by the MAP provider when it detects errors that are not correctable by the MAP user, or if it is unable to report a correctible error to the MAP service user. Fatal errors are indicated via the STREAMS message type `M_ERROR` with the UNIX system error `EPROTO`. The `M_ERROR` STREAMS message type will result in the failure of all the UNIX system calls on the stream. The MAP service user can recover from a fatal error by having all the processes close the files associated with the stream, and then reopening them for processing.

6 References

1. CCITT X.213, (Geneva, 1986), "Network Service Definition for Open Systems Interconnection (OSI) for CCITT Applications," (Grey Book).
2. ISO 8348 — "Information Processing Systems — Data Communications — Newtork Service Definition," 4/15/87
3. ISO 8348/AD — "Information Processing Systems — Data Communications — Newtork Service Definition — Addendum 1: Connectionless Mode Transmission," 4/15/87
4. ISO 8373 — "Information Processing Systems — Data Communicatoins Protocol for Providing the Connectionless Mode Network Service," SC6 N4542
5. ISO 8208 — "Information Processing Systems — X.25 Packet Level Protocol for Data Terminal Equipment," 9/1/87
6. ISO 8878 — "Information Processing Systems — Data Communications — Use of X.25 to Provide the OSI Connection-Mode Network Service," 9/1/87
7. System V Interface Definition, Issue 2 – Volume 3
8. CCITT X.210, (Geneva 1984), "Open Systems Interconnection (OSI) Layer Service Definition Conventions," (Red Book)

Addendum for 3GPP Conformance

Appendix A Mapping MAPI Primitives to 3GPP TS 29.002

Table A-1 shows the mapping of the MAPI primitives to the MAP service definition primitives listed in 3GPP TS 29.002 Release 6.6.0.

Table A-1. Mapping of MAPI Primitives to 3GPP TS 29.002

MAP Primitive	3GPP Service Primitive
`MAP_OPEN_REQ`	MAP-OPEN **req**
`MAP_OPEN_IND`	MAP-OPEN **ind**
`MAP_OPEN_RES`	MAP-OPEN **rsp**
`MAP_OPEN_CON`	MAP-OPEN **cnf**
`MAP_CLOSE_REQ`	MAP-CLOSE **req**
`MAP_CLOSE_IND`	MAP-CLOSE **ind**
`MAP_DELIM_REQ`	MAP-DELIMITER **req**
`MAP_DELIM_IND`	MAP-DELIMITER **ind**
`MAP_ABORT_REQ`	MAP-U-ABORT **req**
`MAP_UABORT_IND`	MAP-U-ABORT **ind**
`MAP_PABORT_IND`	MAP-P-ABORT **ind**
`MAP_NOTICE_IND`	MAP-NOTICE **ind**
`MAP_STC1_REQ`	MAP-SECURE-TRANSPORT-CLASS-1 **req**
`MAP_STC1_IND`	MAP-SECURE-TRANSPORT-CLASS-1 **ind**
`MAP_STC2_REQ`	MAP-SECURE-TRANSPORT-CLASS-2 **req**
`MAP_STC2_IND`	MAP-SECURE-TRANSPORT-CLASS-2 **ind**
`MAP_STC3_REQ`	MAP-SECURE-TRANSPORT-CLASS-3 **req**
`MAP_STC3_IND`	MAP-SECURE-TRANSPORT-CLASS-3 **ind**
`MAP_STC4_REQ`	MAP-SECURE-TRANSPORT-CLASS-4 **req**
`MAP_STC4_IND`	MAP-SECURE-TRANSPORT-CLASS-4 **ind**
`MAP_ACTSS_REQ`	MAP-ACTIVATE-SS **req**
`MAP_ACTSS_IND`	MAP-ACTIVATE-SS **ind**
`MAP_ACTSS_RES`	MAP-ACTIVATE-SS **rsp**
`MAP_ACTSS_CON`	MAP-ACTIVATE-SS **cnf**
`MAP_ACTTM_REQ`	MAP-ACTIVATE-TRACE-MODE **req**
`MAP_ACTTM_IND`	MAP-ACTIVATE-TRACE-MODE **ind**
`MAP_ACTTM_RES`	MAP-ACTIVATE-TRACE-MODE **rsp**
`MAP_ACTTM_CON`	MAP-ACTIVATE-TRACE-MODE **cnf**
`MAP_ALERTSC_REQ`	MAP-ALERT-SERVICE-CENTRE **req**
`MAP_ALERTSC_IND`	MAP-ALERT-SERVICE-CENTRE **ind**
`MAP_ALERTSC_RES`	MAP-ALERT-SERVICE-CENTRE **rsp**
`MAP_ALERTSC_CON`	MAP-ALERT-SERVICE-CENTRE **cnf**
`MAP_ALLOCHN_REQ`	MAP-ALLOCATE-HANDOVER-NUMBER **req**
`MAP_ALLOCHN_IND`	MAP-ALLOCATE-HANDOVER-NUMBER **ind**
`MAP_ALLOCHN_RES`	MAP-ALLOCATE-HANDOVER-NUMBER **rsp**
`MAP_ALLOCHN_CON`	MAP-ALLOCATE-HANDOVER-NUMBER **cnf**
`MAP_ANYMOD_REQ`	MAP-ANY-TIME-MODIFICATION **req**
`MAP_ANYMOD_IND`	MAP-ANY-TIME-MODIFICATION **ind**
`MAP_ANYMOD_RES`	MAP-ANY-TIME-MODIFICATION **rsp**
`MAP_ANYMOD_CON`	MAP-ANY-TIME-MODIFICATION **cnf**
`MAP_ANYQRY_REQ`	MAP-ANY-TIME-INTERROGATION **req**
`MAP_ANYQRY_IND`	MAP-ANY-TIME-INTERROGATION **ind**
`MAP_ANYQRY_RES`	MAP-ANY-TIME-INTERROGATION **rsp**

`MAP_ANYQRY_CON`	MAP-ANY-TIME-INTERROGATION **cnf**
`MAP_ANYSUB_REQ`	MAP-ANY-TIME-SUBSCRIPTION-INTERROGATION **req**
`MAP_ANYSUB_IND`	MAP-ANY-TIME-SUBSCRIPTION-INTERROGATION **ind**
`MAP_ANYSUB_RES`	MAP-ANY-TIME-SUBSCRIPTION-INTERROGATION **rsp**
`MAP_ANYSUB_CON`	MAP-ANY-TIME-SUBSCRIPTION-INTERROGATION **cnf**
`MAP_AUTH_REQ`	MAP-AUTHENTICATE **req**
`MAP_AUTH_IND`	MAP-AUTHENTICATE **ind**
`MAP_AUTH_RES`	MAP-AUTHENTICATE **rsp**
`MAP_AUTH_CON`	MAP-AUTHENTICATE **cnf**
`MAP_AUTHFAIL_REQ`	MAP-AUTHENTICATION-FAILURE-REPORT **req**
`MAP_AUTHFAIL_IND`	MAP-AUTHENTICATION-FAILURE-REPORT **ind**
`MAP_AUTHFAIL_RES`	MAP-AUTHENTICATION-FAILURE-REPORT **rsp**
`MAP_AUTHFAIL_CON`	MAP-AUTHENTICATION-FAILURE-REPORT **cnf**
`MAP_CANCLOC_REQ`	MAP-CANCEL-LOCATION **req**
`MAP_CANCLOC_IND`	MAP-CANCEL-LOCATION **ind**
`MAP_CANCLOC_RES`	MAP-CANCEL-LOCATION **rsp**
`MAP_CANCLOC_CON`	MAP-CANCEL-LOCATION **cnf**
`MAP_CKIEMI_REQ`	MAP-CHECK-IEMI **req**
`MAP_CKIEMI_IND`	MAP-CHECK-IEMI **ind**
`MAP_CKIEMI_RES`	MAP-CHECK-IEMI **rsp**
`MAP_CKIEMI_CON`	MAP-CHECK-IEMI **cnf**
`MAP_DEACTSS_REQ`	MAP-DEACTIVATE-SS **req**
`MAP_DEACTSS_IND`	MAP-DEACTIVATE-SS **ind**
`MAP_DEACTSS_RES`	MAP-DEACTIVATE-SS **rsp**
`MAP_DEACTSS_CON`	MAP-DEACTIVATE-SS **cnf**
`MAP_DEACTTM_REQ`	MAP-DEACTIVATE-TRACE-MODE **req**
`MAP_DEACTTM_IND`	MAP-DEACTIVATE-TRACE-MODE **ind**
`MAP_DEACTTM_RES`	MAP-DEACTIVATE-TRACE-MODE **rsp**
`MAP_DEACTTM_CON`	MAP-DEACTIVATE-TRACE-MODE **cnf**
`MAP_DELSUBD_REQ`	MAP-DELETE-SUBSCRIBER-DATA **req**
`MAP_DELSUBD_IND`	MAP-DELETE-SUBSCRIBER-DATA **ind**
`MAP_DELSUBD_RES`	MAP-DELETE-SUBSCRIBER-DATA **rsp**
`MAP_DELSUBD_CON`	MAP-DELETE-SUBSCRIBER-DATA **cnf**
`MAP_ERASECC_REQ`	MAP-ERASE-CC-ENTRY **req**
`MAP_ERASECC_IND`	MAP-ERASE-CC-ENTRY **ind**
`MAP_ERASECC_RES`	MAP-ERASE-CC-ENTRY **rsp**
`MAP_ERASECC_CON`	MAP-ERASE-CC-ENTRY **cnf**
`MAP_ERASESS_REQ`	MAP-ERASE-SS **req**
`MAP_ERASESS_IND`	MAP-ERASE-SS **ind**
`MAP_ERASESS_RES`	MAP-ERASE-SS **rsp**
`MAP_ERASESS_CON`	MAP-ERASE-SS **cnf**
`MAP_FAILURE_REQ`	MAP-FAILURE-REPORT **req**
`MAP_FAILURE_IND`	MAP-FAILURE-REPORT **ind**
`MAP_FAILURE_RES`	MAP-FAILURE-REPORT **rsp**
`MAP_FAILURE_CON`	MAP-FAILURE-REPORT **cnf**
`MAP_FAS_REQ`	MAP-FORWARD-ACCESS-SIGNALLING **req**
`MAP_FAS_IND`	MAP-FORWARD-ACCESS-SIGNALLING **ind**
`MAP_FAS_RES`	MAP-FORWARD-ACCESS-SIGNALLING **rsp**
`MAP_FAS_CON`	MAP-FORWARD-ACCESS-SIGNALLING **cnf**

`MAP_FCKSSIND_REQ`	MAP-FORWARD-CHECK-SS-INDICATION **req**
`MAP_FCKSSIND_IND`	MAP-FORWARD-CHECK-SS-INDICATION **ind**
`MAP_FCKSSIND_RES`	MAP-FORWARD-CHECK-SS-INDICATION **rsp**
`MAP_FCKSSIND_CON`	MAP-FORWARD-CHECK-SS-INDICATION **cnf**
`MAP_FGRCSIG_REQ`	MAP-FORWARD-GROUP-CALL-SIGNALLING **req**
`MAP_FGRCSIG_IND`	MAP-FORWARD-GROUP-CALL-SIGNALLING **ind**
`MAP_FGRCSIG_RES`	MAP-FORWARD-GROUP-CALL-SIGNALLING **rsp**
`MAP_FGRCSIG_CON`	MAP-FORWARD-GROUP-CALL-SIGNALLING **cnf**
`MAP_FNEWIMSI_REQ`	MAP-FORWARD-NEW-IMSI **req**
`MAP_FNEWIMSI_IND`	MAP-FORWARD-NEW-IMSI **ind**
`MAP_FNEWIMSI_RES`	MAP-FORWARD-NEW-IMSI **rsp**
`MAP_FNEWIMSI_CON`	MAP-FORWARD-NEW-IMSI **cnf**
`MAP_GETIEMI_REQ`	MAP-OBTAIN-IEMI **req**
`MAP_GETIEMI_IND`	MAP-OBTAIN-IEMI **ind**
`MAP_GETIEMI_RES`	MAP-OBTAIN-IEMI **rsp**
`MAP_GETIEMI_CON`	MAP-OBTAIN-IEMI **cnf**
`MAP_GETPASS_REQ`	MAP-GET-PASSWORD **req**
`MAP_GETPASS_IND`	MAP-GET-PASSWORD **ind**
`MAP_GETPASS_RES`	MAP-GET-PASSWORD **rsp**
`MAP_GETPASS_CON`	MAP-GET-PASSWORD **cnf**
`MAP_INFORMSC_REQ`	MAP-INFORM-SERVICE-CENTER **req**
`MAP_INFORMSC_IND`	MAP-INFORM-SERVICE-CENTER **ind**
`MAP_INFORMSC_RES`	MAP-INFORM-SERVICE-CENTER **rsp**
`MAP_INFORMSC_CON`	MAP-INFORM-SERVICE-CENTER **cnf**
`MAP_INSSUBSD_REQ`	MAP-INSERT-SUBSCRIBER-DATA **req**
`MAP_INSSUBSD_IND`	MAP-INSERT-SUBSCRIBER-DATA **ind**
`MAP_INSSUBSD_RES`	MAP-INSERT-SUBSCRIBER-DATA **rsp**
`MAP_INSSUBSD_CON`	MAP-INSERT-SUBSCRIBER-DATA **cnf**
`MAP_INTERGSS_REQ`	MAP-INTERROGATE-SS **req**
`MAP_INTERGSS_IND`	MAP-INTERROGATE-SS **ind**
`MAP_INTERGSS_RES`	MAP-INTERROGATE-SS **rsp**
`MAP_INTERGSS_CON`	MAP-INTERROGATE-SS **cnf**
`MAP_ISTALERT_REQ`	MAP-IST-ALERT **req**
`MAP_ISTALERT_IND`	MAP-IST-ALERT **ind**
`MAP_ISTALERT_RES`	MAP-IST-ALERT **rsp**
`MAP_ISTALERT_CON`	MAP-IST-ALERT **cnf**
`MAP_ISTCMD_REQ`	MAP-IST-COMMAND **req**
`MAP_ISTCMD_IND`	MAP-IST-COMMAND **ind**
`MAP_ISTCMD_RES`	MAP-IST-COMMAND **rsp**
`MAP_ISTCMD_CON`	MAP-IST-COMMAND **cnf**
`MAP_MMEVENT_REQ`	MAP-NOTE-MM-EVENT **req**
`MAP_MMEVENT_IND`	MAP-NOTE-MM-EVENT **ind**
`MAP_MMEVENT_RES`	MAP-NOTE-MM-EVENT **rsp**
`MAP_MMEVENT_CON`	MAP-NOTE-MM-EVENT **cnf**
`MAP_MOFSMS_REQ`	MAP-MO-FORWARD-SHORT-MESSAGE **req**
`MAP_MOFSMS_IND`	MAP-MO-FORWARD-SHORT-MESSAGE **ind**
`MAP_MOFSMS_RES`	MAP-MO-FORWARD-SHORT-MESSAGE **rsp**
`MAP_MOFSMS_CON`	MAP-MO-FORWARD-SHORT-MESSAGE **cnf**
`MAP_MSGRPSPRES_REQ`	MAP-NOTE-MS-PRESENT-FOR-GPRS **req**

MAP_MSGRPSPRES_IND	MAP-NOTE-MS-PRESENT-FOR-GPRS **ind**
MAP_MSGRPSPRES_RES	MAP-NOTE-MS-PRESENT-FOR-GPRS **rsp**
MAP_MSGRPSPRES_CON	MAP-NOTE-MS-PRESENT-FOR-GPRS **cnf**
MAP_MTFSMS_REQ	MAP-MT-FORWARD-SHORT-MESSAGE **req**
MAP_MTFSMS_IND	MAP-MT-FORWARD-SHORT-MESSAGE **ind**
MAP_MTFSMS_RES	MAP-MT-FORWARD-SHORT-MESSAGE **rsp**
MAP_MTFSMS_CON	MAP-MT-FORWARD-SHORT-MESSAGE **cnf**
MAP_PAGE_REQ	MAP-PAGE **req**
MAP_PAGE_IND	MAP-PAGE **ind**
MAP_PAGE_RES	MAP-PAGE **rsp**
MAP_PAGE_CON	MAP-PAGE **cnf**
MAP_PREPGC_REQ	MAP-PREPARE-GROUP-CALL **req**
MAP_PREPGC_IND	MAP-PREPARE-GROUP-CALL **ind**
MAP_PREPGC_RES	MAP-PREPARE-GROUP-CALL **rsp**
MAP_PREPGC_CON	MAP-PREPARE-GROUP-CALL **cnf**
MAP_PREPHO_REQ	MAP-PREPARE-HANDOVER **req**
MAP_PREPHO_IND	MAP-PREPARE-HANDOVER **ind**
MAP_PREPHO_RES	MAP-PREPARE-HANDOVER **rsp**
MAP_PREPHO_CON	MAP-PREPARE-HANDOVER **cnf**
MAP_PREPSH_REQ	MAP-PREPARE-SUBSEQUENT-HANDOVER **req**
MAP_PREPSH_IND	MAP-PREPARE-SUBSEQUENT-HANDOVER **ind**
MAP_PREPSH_RES	MAP-PREPARE-SUBSEQUENT-HANDOVER **rsp**
MAP_PREPSH_CON	MAP-PREPARE-SUBSEQUENT-HANDOVER **cnf**
MAP_PROCAS_REQ	MAP-PROCESS-ACCESS-SIGNALLING **req**
MAP_PROCAS_IND	MAP-PROCESS-ACCESS-SIGNALLING **ind**
MAP_PROCAS_RES	MAP-PROCESS-ACCESS-SIGNALLING **rsp**
MAP_PROCAS_CON	MAP-PROCESS-ACCESS-SIGNALLING **cnf**
MAP_PROCGC_REQ	MAP-PROCESS-GROUP-CALL-SIGNALLING **req**
MAP_PROCGC_IND	MAP-PROCESS-GROUP-CALL-SIGNALLING **ind**
MAP_PROCGC_RES	MAP-PROCESS-GROUP-CALL-SIGNALLING **rsp**
MAP_PROCGC_CON	MAP-PROCESS-GROUP-CALL-SIGNALLING **cnf**
MAP_PROCUSS_REQ	MAP-PROCESS-UNSTRUCTURED-SS-REQUEST **req**
MAP_PROCUSS_IND	MAP-PROCESS-UNSTRUCTURED-SS-REQUEST **ind**
MAP_PROCUSS_RES	MAP-PROCESS-UNSTRUCTURED-SS-REQUEST **rsp**
MAP_PROCUSS_CON	MAP-PROCESS-UNSTRUCTURED-SS-REQUEST **cnf**
MAP_PROVIMSI_REQ	MAP-PROVIDE-IMSI **req**
MAP_PROVIMSI_IND	MAP-PROVIDE-IMSI **ind**
MAP_PROVIMSI_RES	MAP-PROVIDE-IMSI **rsp**
MAP_PROVIMSI_CON	MAP-PROVIDE-IMSI **cnf**
MAP_PROVRN_REQ	MAP-PROVIDE-ROAMING-NUMBER **req**
MAP_PROVRN_IND	MAP-PROVIDE-ROAMING-NUMBER **ind**
MAP_PROVRN_RES	MAP-PROVIDE-ROAMING-NUMBER **rsp**
MAP_PROVRN_CON	MAP-PROVIDE-ROAMING-NUMBER **cnf**
MAP_PROVSLOC_REQ	MAP-PROVIDE-SUBSCRIBER-LOCATION **req**
MAP_PROVSLOC_IND	MAP-PROVIDE-SUBSCRIBER-LOCATION **ind**
MAP_PROVSLOC_RES	MAP-PROVIDE-SUBSCRIBER-LOCATION **rsp**
MAP_PROVSLOC_CON	MAP-PROVIDE-SUBSCRIBER-LOCATION **cnf**
MAP_PROVSUBI_REQ	MAP-PROVIDE-SUBSCRIBER-INFO **req**
MAP_PROVSUBI_IND	MAP-PROVIDE-SUBSCRIBER-INFO **ind**

`MAP_PROVSUBI_RES`	MAP-PROVIDE-SUBSCRIBER-INFO **rsp**
`MAP_PROVSUBI_CON`	MAP-PROVIDE-SUBSCRIBER-INFO **cnf**
`MAP_PURGEMS_REQ`	MAP-PURGE-MS **req**
`MAP_PURGEMS_IND`	MAP-PURGE-MS **ind**
`MAP_PURGEMS_RES`	MAP-PURGE-MS **rsp**
`MAP_PURGEMS_CON`	MAP-PURGE-MS **cnf**
`MAP_RDYSM_REQ`	MAP-READY-FOR-SM **req**
`MAP_RDYSM_IND`	MAP-READY-FOR-SM **ind**
`MAP_RDYSM_RES`	MAP-READY-FOR-SM **rsp**
`MAP_RDYSM_CON`	MAP-READY-FOR-SM **cnf**
`MAP_REGCC_REQ`	MAP-REGISTER-CC-ENTRY **req**
`MAP_REGCC_IND`	MAP-REGISTER-CC-ENTRY **ind**
`MAP_REGCC_RES`	MAP-REGISTER-CC-ENTRY **rsp**
`MAP_REGCC_CON`	MAP-REGISTER-CC-ENTRY **cnf**
`MAP_REGPW_REQ`	MAP-REGISTER-PASSWORD **req**
`MAP_REGPW_IND`	MAP-REGISTER-PASSWORD **ind**
`MAP_REGPW_RES`	MAP-REGISTER-PASSWORD **rsp**
`MAP_REGPW_CON`	MAP-REGISTER-PASSWORD **cnf**
`MAP_REGSS_REQ`	MAP-REGISTER-SS **req**
`MAP_REGSS_IND`	MAP-REGISTER-SS **ind**
`MAP_REGSS_RES`	MAP-REGISTER-SS **rsp**
`MAP_REGSS_CON`	MAP-REGISTER-SS **cnf**
`MAP_REMUSRFREE_REQ`	MAP-REMOTE-USER-FREE **req**
`MAP_REMUSRFREE_IND`	MAP-REMOTE-USER-FREE **ind**
`MAP_REMUSRFREE_RES`	MAP-REMOTE-USER-FREE **rsp**
`MAP_REMUSRFREE_CON`	MAP-REMOTE-USER-FREE **cnf**
`MAP_RESET_REQ`	MAP-RESET **req**
`MAP_RESET_IND`	MAP-RESET **ind**
`MAP_RESET_RES`	MAP-RESET **rsp**
`MAP_RESET_CON`	MAP-RESET **cnf**
`MAP_RESTORE_REQ`	MAP-RESTORE-DATA **req**
`MAP_RESTORE_IND`	MAP-RESTORE-DATA **ind**
`MAP_RESTORE_RES`	MAP-RESTORE-DATA **rsp**
`MAP_RESTORE_CON`	MAP-RESTORE-DATA **cnf**
`MAP_RESUME_REQ`	MAP-RESUME-CALL-HANDLING **req**
`MAP_RESUME_IND`	MAP-RESUME-CALL-HANDLING **ind**
`MAP_RESUME_RES`	MAP-RESUME-CALL-HANDLING **rsp**
`MAP_RESUME_CON`	MAP-RESUME-CALL-HANDLING **cnf**
`MAP_SEARCH_REQ`	MAP-SEARCH-FOR-MS **req**
`MAP_SEARCH_IND`	MAP-SEARCH-FOR-MS **ind**
`MAP_SEARCH_RES`	MAP-SEARCH-FOR-MS **rsp**
`MAP_SEARCH_CON`	MAP-SEARCH-FOR-MS **cnf**
`MAP_SENDAUTHI_REQ`	MAP-SEND-AUTHENTICATION-INFO **req**
`MAP_SENDAUTHI_IND`	MAP-SEND-AUTHENTICATION-INFO **ind**
`MAP_SENDAUTHI_RES`	MAP-SEND-AUTHENTICATION-INFO **rsp**
`MAP_SENDAUTHI_CON`	MAP-SEND-AUTHENTICATION-INFO **cnf**
`MAP_SENDEND_REQ`	MAP-SEND-END-SIGNAL **req**
`MAP_SENDEND_IND`	MAP-SEND-END-SIGNAL **ind**
`MAP_SENDEND_RES`	MAP-SEND-END-SIGNAL **rsp**

`MAP_SENDEND_CON`	MAP-SEND-END-SIGNAL **cnf**
`MAP_SENDGCEND_REQ`	MAP-SEND-GROUP-CALL-END-SIGNAL **req**
`MAP_SENDGCEND_IND`	MAP-SEND-GROUP-CALL-END-SIGNAL **ind**
`MAP_SENDGCEND_RES`	MAP-SEND-GROUP-CALL-END-SIGNAL **rsp**
`MAP_SENDGCEND_CON`	MAP-SEND-GROUP-CALL-END-SIGNAL **cnf**
`MAP_SENDGPRSRI_REQ`	MAP-SEND-ROUTING-INFO-FOR-GPRS **req**
`MAP_SENDGPRSRI_IND`	MAP-SEND-ROUTING-INFO-FOR-GPRS **ind**
`MAP_SENDGPRSRI_RES`	MAP-SEND-ROUTING-INFO-FOR-GPRS **rsp**
`MAP_SENDGPRSRI_CON`	MAP-SEND-ROUTING-INFO-FOR-GPRS **cnf**
`MAP_SENDHOREP_REQ`	MAP-SEND-HANDOVER-REPORT **req**
`MAP_SENDHOREP_IND`	MAP-SEND-HANDOVER-REPORT **ind**
`MAP_SENDHOREP_RES`	MAP-SEND-HANDOVER-REPORT **rsp**
`MAP_SENDHOREP_CON`	MAP-SEND-HANDOVER-REPORT **cnf**
`MAP_SENDID_REQ`	MAP-SEND-IDENTIFICATION **req**
`MAP_SENDID_IND`	MAP-SEND-IDENTIFICATION **ind**
`MAP_SENDID_RES`	MAP-SEND-IDENTIFICATION **rsp**
`MAP_SENDID_CON`	MAP-SEND-IDENTIFICATION **cnf**
`MAP_SENDIMSI_REQ`	MAP-SEND-IMSI **req**
`MAP_SENDIMSI_IND`	MAP-SEND-IMSI **ind**
`MAP_SENDIMSI_RES`	MAP-SEND-IMSI **rsp**
`MAP_SENDIMSI_CON`	MAP-SEND-IMSI **cnf**
`MAP_SENDLCSRI_REQ`	MAP-SEND-ROUTING-INFO-FOR-LCS **req**
`MAP_SENDLCSRI_IND`	MAP-SEND-ROUTING-INFO-FOR-LCS **ind**
`MAP_SENDLCSRI_RES`	MAP-SEND-ROUTING-INFO-FOR-LCS **rsp**
`MAP_SENDLCSRI_CON`	MAP-SEND-ROUTING-INFO-FOR-LCS **cnf**
`MAP_SENDMOSMSI_REQ`	MAP-SEND-INFO-FOR-MO-SMS **req**
`MAP_SENDMOSMSI_IND`	MAP-SEND-INFO-FOR-MO-SMS **ind**
`MAP_SENDMOSMSI_RES`	MAP-SEND-INFO-FOR-MO-SMS **rsp**
`MAP_SENDMOSMSI_CON`	MAP-SEND-INFO-FOR-MO-SMS **cnf**
`MAP_SENDMTSMSI_REQ`	MAP-SEND-INFO-FOR-MT-SMS **req**
`MAP_SENDMTSMSI_IND`	MAP-SEND-INFO-FOR-MT-SMS **ind**
`MAP_SENDMTSMSI_RES`	MAP-SEND-INFO-FOR-MT-SMS **rsp**
`MAP_SENDMTSMSI_CON`	MAP-SEND-INFO-FOR-MT-SMS **cnf**
`MAP_SENDRI_REQ`	MAP-SEND-ROUTING-INFORMATION **req**
`MAP_SENDRI_IND`	MAP-SEND-ROUTING-INFORMATION **ind**
`MAP_SENDRI_RES`	MAP-SEND-ROUTING-INFORMATION **rsp**
`MAP_SENDRI_CON`	MAP-SEND-ROUTING-INFORMATION **cnf**
`MAP_SENDSMSRI_REQ`	MAP-SEND-ROUTING-INFO-FOR-SM **req**
`MAP_SENDSMSRI_IND`	MAP-SEND-ROUTING-INFO-FOR-SM **ind**
`MAP_SENDSMSRI_RES`	MAP-SEND-ROUTING-INFO-FOR-SM **rsp**
`MAP_SENDSMSRI_CON`	MAP-SEND-ROUTING-INFO-FOR-SM **cnf**
`MAP_SETCM_REQ`	MAP-SET-CIPHERING-MODE **req**
`MAP_SETCM_IND`	MAP-SET-CIPHERING-MODE **ind**
`MAP_SETCM_RES`	MAP-SET-CIPHERING-MODE **rsp**
`MAP_SETCM_CON`	MAP-SET-CIPHERING-MODE **cnf**
`MAP_SETREP_REQ`	MAP-SET-REPORTING-STATE **req**
`MAP_SETREP_IND`	MAP-SET-REPORTING-STATE **ind**
`MAP_SETREP_RES`	MAP-SET-REPORTING-STATE **rsp**
`MAP_SETREP_CON`	MAP-SET-REPORTING-STATE **cnf**

`MAP_SMDELIVSTAT_REQ`	MAP-REPORT-SM-DELIVER-STATUS **req**
`MAP_SMDELIVSTAT_IND`	MAP-REPORT-SM-DELIVER-STATUS **ind**
`MAP_SMDELIVSTAT_RES`	MAP-REPORT-SM-DELIVER-STATUS **rsp**
`MAP_SMDELIVSTAT_CON`	MAP-REPORT-SM-DELIVER-STATUS **cnf**
`MAP_SSINV_REQ`	MAP-SS-INVOCATION-NOTIFICATION **req**
`MAP_SSINV_IND`	MAP-SS-INVOCATION-NOTIFICATION **ind**
`MAP_SSINV_RES`	MAP-SS-INVOCATION-NOTIFICATION **rsp**
`MAP_SSINV_CON`	MAP-SS-INVOCATION-NOTIFICATION **cnf**
`MAP_STATUS_REQ`	MAP-STATUS-REPORT **req**
`MAP_STATUS_IND`	MAP-STATUS-REPORT **ind**
`MAP_STATUS_RES`	MAP-STATUS-REPORT **rsp**
`MAP_STATUS_CON`	MAP-STATUS-REPORT **cnf**
`MAP_SUBLOCREP_REQ`	MAP-SUBSCRIBER-LOCATION-REPORT **req**
`MAP_SUBLOCREP_IND`	MAP-SUBSCRIBER-LOCATION-REPORT **ind**
`MAP_SUBLOCREP_RES`	MAP-SUBSCRIBER-LOCATION-REPORT **rsp**
`MAP_SUBLOCREP_CON`	MAP-SUBSCRIBER-LOCATION-REPORT **cnf**
`MAP_SUBSDATAMOD_REQ`	MAP-NOTE-SUBSCRIBER-DATA-MODIFIED **req**
`MAP_SUBSDATAMOD_IND`	MAP-NOTE-SUBSCRIBER-DATA-MODIFIED **ind**
`MAP_SUBSDATAMOD_RES`	MAP-NOTE-SUBSCRIBER-DATA-MODIFIED **rsp**
`MAP_SUBSDATAMOD_CON`	MAP-NOTE-SUBSCRIBER-DATA-MODIFIED **cnf**
`MAP_TRACESA_REQ`	MAP-TRACE-SUBSCRIBER-ACTIVITY **req**
`MAP_TRACESA_IND`	MAP-TRACE-SUBSCRIBER-ACTIVITY **ind**
`MAP_TRACESA_RES`	MAP-TRACE-SUBSCRIBER-ACTIVITY **rsp**
`MAP_TRACESA_CON`	MAP-TRACE-SUBSCRIBER-ACTIVITY **cnf**
`MAP_UDGRPSLOC_REQ`	MAP-UPDATE-GPRS-LOCATION **req**
`MAP_UDGRPSLOC_IND`	MAP-UPDATE-GPRS-LOCATION **ind**
`MAP_UDGRPSLOC_RES`	MAP-UPDATE-GPRS-LOCATION **rsp**
`MAP_UDGRPSLOC_CON`	MAP-UPDATE-GPRS-LOCATION **cnf**
`MAP_UDLOC_REQ`	MAP-UPDATE-LOCATION **req**
`MAP_UDLOC_IND`	MAP-UPDATE-LOCATION **ind**
`MAP_UDLOC_RES`	MAP-UPDATE-LOCATION **rsp**
`MAP_UDLOC_CON`	MAP-UPDATE-LOCATION **cnf**
`MAP_USSDNTFY_REQ`	MAP-UNSTRUCTURED-SS-NOTIFY **req**
`MAP_USSDNTFY_IND`	MAP-UNSTRUCTURED-SS-NOTIFY **ind**
`MAP_USSDNTFY_RES`	MAP-UNSTRUCTURED-SS-NOTIFY **rsp**
`MAP_USSDNTFY_CON`	MAP-UNSTRUCTURED-SS-NOTIFY **cnf**
`MAP_USSDREQ_REQ`	MAP-UNSTRUCTURED-SS-REQUEST **req**
`MAP_USSDREQ_IND`	MAP-UNSTRUCTURED-SS-REQUEST **ind**
`MAP_USSDREQ_RES`	MAP-UNSTRUCTURED-SS-REQUEST **rsp**
`MAP_USSDREQ_CON`	MAP-UNSTRUCTURED-SS-REQUEST **cnf**

A.1 Application Contexts

Table 1. Application Contexts

Ver.	Application Contexts	Operations
v3	locationCancellationContext	cancelLocation
v3	equipmentMngtContext	checkIEMI

v2	imsiRetrievalContext	sendIMSI
v3	infoRetrievalContext	sendAuthenticationInfo
v3	interVlrInfoRetrievalContext	sendIdentification
v3	handoverControlContext	prepareHandover, forwareAccessSignalling, sendEndSignal, processAccessSignalling, prepareSubsequentHandover
v3	mwdMngtContext	readyForSM
v3	msPurgingContext	purgeMS
v2	shortMsgAlertContext	alertServiceCentre
v2	resetContext	reset
v2	networkUnstructuredSsContext	processUnstructuredSS-Request, unstructuredSS-Request, unstructuredSS-Notify
v3	tracingContext	activateTraceMode, deactivateTraceMode
v2	networkFunctionalSsContext	registerSS, eraseSS, activateSS, deactivateSS, registerPassword, interrogateSS, getPassword
v3	shortMsgMO-RelayContext	mo-forwardSM
v3	shortMsgMT-RelayContext	mt-forwardSM
v3	shortMsgGatewayContext	sendRoutingInfoForSM, reportSM-DeliveryStatus, InformServiceCentre
v3	networkLocUpContext	updateLocation, forwardCheckSs-Indication, restoreData, insertSubscriberData, activateTraceMode
v3	gprsLocationUpdateContext	updateGprsLocation, insertSubscriberData, activateTraceMode
v3	subscriberDataMngtContext	insertSubscriberData, deleteSubscriberData
v3	roamingNumberEnquiryContext	provideRoamingNumber
v3	locationInfoRetrievalContext	sendRoutingInfo
v3	gprsNotifyContext	noteMsPresentForGprs
v4	gprsLocationInfoRetrievalContext	sendRoutingInfoForGprs
v3	failureReportContext	failureReport
v4	callControlTransferContext	resumeCallHandling
v3	susbcriberInfoEnquiryContext	provideSubscriberInfo
v3	anyTimeEnquiryContext	anyTimeInterrogation

v3	anyTimeInfoHandlingContext	anyTimeSubscriptionInterrogation, anyTimeModification
v3	ss-InvocationNotificationContext	ss-InvocationNotification
v3	groupCallControlContext	prepareGroupCall, processGroupCallSignalling, forwardGroupCallSignalling, sendGroupCallEndSignal
v3	reportingContext	setReportingState, statusReport, remoteUserFree
v3	callCompletionContext	registerCC-Entry, eraceCC-Entry
v3	isAlertingContext	isAlert
v3	ServiceTerminationContext	istCommand
v3	locationSvcEnquiryContext	provideSubscriberLocation, subscriberLocationReport
v3	locationSvcGatewayContext	sendRoutingInfoForLCS
v3	mm-EventReportingContext	noteMM-Event
v3	subscriberDataModificationNotificationContext	noteSubscriberDataModified
v3	authenticationFailureReportContext	authenticationFailureReport
v3	secureTransportHandlingContext	secureTransportClass1, secureTransportClass2, secureTransportClass3, secureTransportClass4

Appendix B State/Event Tables

Appendix C Primitive Precedence Tables

Appendix D MAPI Header File Listing

Licenses

All code presented in this manual is licensed under the [GNU Affero General Public License], page 395. The text of this manual is licensed under the [GNU Free Documentation License], page 405, with no invariant sections, no front-cover texts and no back-cover texts. Please note, however, that it is just plain wrong to modify statements of, or attribute statements to, the Author or *OpenSS7 Corporation*.

GNU Affero General Public License

Preamble

The GNU Affero General Public License is a free, copyleft license for software and other kinds of works, specifically designed to ensure cooperation with the community in the case of network server software.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, our General Public Licenses are intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

Developers that use our General Public Licenses protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License which gives you legal permission to copy, distribute and/or modify the software.

A secondary benefit of defending all users' freedom is that improvements made in alternate versions of the program, if they receive widespread use, become available for other developers to incorporate. Many developers of free software are heartened and encouraged by the resulting cooperation. However, in the case of software used on network servers, this result may fail to come about. The GNU General Public License permits making a modified version and letting the public access it on a server without ever releasing its source code to the public.

The GNU Affero General Public License is designed specifically to ensure that, in such cases, the modified source code becomes available to the community. It requires the operator of a network server to provide the source code of the modified version running there to the users of that server. Therefore, public use of a modified version, on a publicly accessible server, gives the public access to the source code of the modified version.

An older license, called the Affero General Public License and published by Affero, was designed to accomplish similar goals. This is a different license, not a version of the Affero GPL, but Affero has released a new version of the Affero GPL which permits relicensing under this license.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

   "This License" refers to version 3 of the GNU Affero General Public License.

   "Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

   "The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

   To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

   A "covered work" means either the unmodified Program or a work based on the Program.

   To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

   To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

   An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

   A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

   The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

   All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

   You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

   Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

   No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

   When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   a. The work must carry prominent notices stating that you modified it, and giving a relevant date.

b. The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

c. You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d. If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a. Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b. Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c. Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d. Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e. Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

   "Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

   When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a. Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
b. Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
c. Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
d. Limiting the use for publicity purposes of names of licensors or authors of the material; or
e. Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
f. Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

    Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

    An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

    You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

    A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

    A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

    Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

    In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

    If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others’ Freedom.

    If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Remote Network Interaction; Use with the GNU General Public License.

    Notwithstanding any other provision of this License, if you modify the Program, your modified version must prominently offer all users interacting with it remotely through a network (if your version supports such interaction) an opportunity to receive the Corresponding Source of your version by providing access to the Corresponding Source from a network server at no charge, through some standard or customary means of facilitating copying of software. This Corresponding Source shall include the Corresponding Source for any work covered by version 3 of the GNU General Public License that is incorporated pursuant to the following paragraph.

    Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the work with which it is combined will remain governed by version 3 of the GNU General Public License.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of the GNU Affero General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU Affero General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU Affero General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU Affero General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

    THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

    If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and a brief idea of what it does.
Copyright (C) year name of author

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU Affero General Public License as published by
the Free Software Foundation, either version 3 of the License, or (at
your option) any later version.

This program is distributed in the hope that it will be useful, but
WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License
along with this program.  If not, see http://www.gnu.org/licenses/.
```

Also add information on how to contact you by electronic and paper mail.

If your software can interact with users remotely through a network, you should also make sure that it provides a way for users to get its source. For example, if your program is a web application, its interface could display a "Source" link that leads users to an archive of the code. There are many ways you could offer source, and different solutions will be better for different programs; see section 13 for the specific requirements.

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU AGPL, see `http://www.gnu.org/licenses/`.

GNU Free Documentation License

GNU FREE DOCUMENTATION LICENSE
Version 1.3, 3 November 2008

0. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

   This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

   We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

   This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

   A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

   A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

   The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTEX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

   You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

   You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

   If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

   If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

   It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

   You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

   A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

   B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

   C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

   D. Preserve all the copyright notices of the Document.

   E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

   F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements.”

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
  Copyright (C)  year  your name.
  Permission is granted to copy, distribute and/or modify this document
  under the terms of the GNU Free Documentation License, Version 1.3
  or any later version published by the Free Software Foundation;
  with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
  Texts.  A copy of the license is included in the section entitled ``GNU
  Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

```
    with the Invariant Sections being list their titles, with
    the Front-Cover Texts being list, and with the Back-Cover Texts
    being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Index