

First edition
2013-07-01

AMENDMENT 1
2017-04

**Intelligent transport systems —
Communications access for land
mobiles (CALM) — ITS station
management —**

**Part 3:
Service access points**

AMENDMENT 1

*Systèmes intelligents de transport — Accès aux communications des
services mobiles terrestres (CALM) — Gestion des stations ITS —*

Partie 3: Points d'accès au service

AMENDEMENT 1



Reference number
ISO 24102-3:2013/Amd.1:2017(E)

© ISO 2017



COPYRIGHT PROTECTED DOCUMENT

© ISO 2017, Published in Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

Amendment 1 to ISO 24102-3:2013 was prepared by Technical Committee ISO/TC 204, *Intelligent transport systems*.

A list of all parts in the ISO 24102 series can be found on the ISO website.

STANDARDSISO.COM : Click to view the full PDF of ISO 24102-3:2013/Amd 1:2017

Intelligent transport systems — Communications access for land mobiles (CALM) — ITS station management —

Part 3: Service access points

AMENDMENT 1

Page 1, Scope

Add the following new list item with two sub-items after the last list item:

- the interfaces between the ITS-S application entity and
 - the ITS-S management entity (MA-SAP), and
 - the ITS-S security entity (SA-SAP).

Page 2, Abbreviated terms

Add the following abbreviated terms before MF-COMMAND:

- | | |
|------------|---|
| MA-COMMAND | command issued by the ITS-S management entity and sent to the ITS-S application entity via the MA-SAP |
| MA-REQUEST | command issued by the ITS-S management entity and sent to the ITS-S security entity via the MA-SAP |

Add the following abbreviated terms before SF-COMMAND:

- | | |
|------------|---|
| SA-COMMAND | command issued by the ITS-S security entity and sent to the ITS-S application entity via the SA-SAP |
| SA-REQUEST | command issued by the ITS-S application entity and sent to the ITS-S security entity via the SA-SAP |

Page 3, Requirements

In the second paragraph, add the following as the first item on the list:

- MA-SAP, SA-SAP,

At the end of the paragraph, before NOTE 1, add the following sentence:

The presentation of the specified services, service primitives and functions in ASN.1 shall be as specified in Annex A.

Replace NOTE 2 by the following:

NOTE 2 All interfaces towards the ITS-S applications are considered to be implemented in an API.

In the third paragraph, replace the first sentence with the following:

Means to secure the access to management functionality need to be specified within the global context of the BSMD security.

In the last paragraph, insert the following before the last item on the list:

— Clause 13 specifies the MA-SAP and SA-SAP.

Page 23

Insert as a new Clause 13 the following clause and subclauses:

13 Interfaces MA and SA towards the ITS-S application entity

13.1 SAPs and API

The API illustrated in Figure 1 contains the functionality of the interfaces MA, FA, and SA specified in ISO 21217. The MA-SAP and the SA-SAP are specified in this document.

Basically, there shall be different types of services that facilitate

- a) sending a command from the ITS-S management entity or the ITS-S security entity to the ITS applications entity, and
- b) receiving a request (command) from the ITS-S application entity by the ITS-S management entity or the ITS-S security entity.

Issuing of commands by the ITS-S management entity to the ITS applications entity shall be built on the service MA-COMMAND.

Receiving of requests from the ITS-S application entity by the ITS-S management entity shall be built on the service MA-REQUEST.

Issuing of commands by the ITS-S security entity to the ITS applications entity shall be built on the service SA-COMMAND.

Receiving of requests from the ITS-S application entity by the ITS-S security entity shall be built on the service SA-REQUEST.

Details of the service primitives shall be as specified in Annex A.

13.2 MA-COMMAND

13.2.1 MA-COMMANDs

Annex Q provides an overview and coding details on MA-COMMANDs that may be sent to the ITS-S application entity.

MA-COMMANDs shall be enabled by means of the service primitives MA-COMMAND.request and MA-COMMAND.confirm.

13.2.2 MA-COMMAND.request

The management service primitive MA-COMMAND.request allows the ITS-S management entity to trigger an action at the ITS-S application entity.

The parameters of the management service primitive MA-COMMAND.request are as follows:

```
MA-COMMAND.request    (  
                        CommandRef,  
                        MA-Command  
                        )
```

Table 33 — MA-COMMAND.request Parameter description

Name	Description
CommandRef	Unique cyclic reference number of command.
A-Command.No	See Annex Q.
A-Command.Value	See Annex Q.

An MA-COMMAND.request shall be generated by the ITS-S management entity when the ITS-S application entity shall perform an action.

On receipt of MA-COMMAND.request the requested action shall be performed.

13.2.3 MA-COMMAND.confirm

The service primitive MA-COMMAND.confirm reports the result of a previous MA-COMMAND.request.

The parameters of MA-COMMAND.confirm are as follows:

```
MA-COMMAND.confirm  (
    CommandRef,
    MA-CmdConfirm,
    ErrStatus
)
```

Table 34 — MA-COMMAND.confirm Parameter description

Name	Description
CommandRef	Unique cyclic reference number of command. Same value as in related MA-COMMAND.request.
MA-CmdConfirm.No	Reference number of command. Same value as A-Command.No in related MA-COMMAND.request.
MA-CmdConfirm.Value	Carries optional confirm data.
ErrStatus	Error/return code, see Table P.1

An MA-COMMAND.confirm shall be generated by the ITS-S application entity upon performance of a previous MA-COMMAND.request in case ErrStatus indicates error or MA-CmdConfirm is present. It may be generated in case ErrStatus indicates success or A-CmdConfirm is not present.

On receipt of MA-COMMAND.confirm, ErrStatus and MA-CmdConfirm shall be evaluated and a possible action shall be performed properly.

13.3 MA-REQUEST

13.3.1 MA-REQUESTs

Annex R provides an overview and coding details on MA-REQUESTs.

MA-REQUESTs shall be enabled by means of the service primitives MA-REQUEST.request and MA-REQUEST.confirm.

13.3.2 MA-REQUEST.request

The management service primitive MA-REQUEST.request allows the ITS-S application entity to trigger an action at the ITS-S management entity.

The parameters of MA-REQUEST.request are as follows:

```
MA-REQUEST.request    (
                        CommandRef,
                        MA-Request
                        )
```

Table 35 — MA-REQUEST.request Parameter description

Name	Description
CommandRef	Unique cyclic reference number of command.
MA-Request.No	See Annex R.
MA-Request.Value	See Annex R.

An MA-REQUEST.request shall be generated by the ITS-S application entity when the ITS-S management entity shall perform an action.

On receipt of MA-REQUEST.request the required action shall be performed.

13.3.3 MA-REQUEST.confirm

The service primitive MA-REQUEST.confirm reports the result of a previous MA-REQUEST.request.

The parameters of MA-REQUEST.confirm are as follows:

```
MA-REQUEST.confirm    (
                        CommandRef,
                        MA-ReqConfirm,
                        ErrStatus
                        )
```

Table 36 — MA-REQUEST.confirm Parameter description

Name	Description
CommandRef	Unique cyclic reference number of command. Same value as in related MA-REQUEST.request.
MA-ReqConfirm.No	Reference number of command. Same value as MA-Request.No in related MA-REQUEST.request.
MA-ReqConfirm.Value	Carries optional confirm data.
ErrStatus	Error/return code, see Table P.1.

An MA-REQUEST.confirm shall be generated by the ITS-S management entity upon performance of a previous MA-REQUEST.request in case ErrStatus indicates error or MA-ReqConfirm is present. It may be generated in case ErrStatus indicates success or MA-ReqConfirm is not present.

On receipt of MA-REQUEST.confirm, ErrStatus and MA-ReqConfirm shall be evaluated and a possible action shall be performed properly. Details are outside the scope of this document.

13.4 SA-COMMAND

13.4.1 SA-COMMANDs

Annex S provides an overview and coding details on SA-COMMANDs that may be sent to the ITS-S application entity.

SA-COMMANDs shall be enabled by means of the service primitives SA-COMMAND.request and SA-COMMAND.confirm.

13.4.2 SA-COMMAND.request

The management service primitive SA-COMMAND.request allows the ITS-S management entity to trigger an action at the ITS-S application entity.

The parameters of the management service primitive SA-COMMAND.request are as follows:

```
SA-COMMAND.request  (
    CommandRef,
    SA-Command
)
```

Table 37 — SA-COMMAND.request Parameter description

Name	Description
CommandRef	Unique cyclic reference number of command.
A-Command.No	See Annex S.
A-Command.Value	See Annex S.

An SA-COMMAND.request shall be generated by the ITS-S management entity when the ITS-S application entity shall perform an action.

On receipt of SA-COMMAND.request the requested action shall be performed.

13.4.3 SA-COMMAND.confirm

The service primitive SA-COMMAND.confirm reports the result of a previous SA-COMMAND.request.

The parameters of SA-COMMAND.confirm are as follows:

```
SA-COMMAND.confirm  (
    CommandRef,
    SA-CmdConfirm,
    ErrStatus
)
```

Table 38 — SA-COMMAND.confirm Parameter description

Name	Description
CommandRef	Unique cyclic reference number of command. Same value as in related SA-COMMAND.request.
SA-CmdConfirm.No	Reference number of command. Same value as A-Command.No in related SA-COMMAND.request.
SA-CmdConfirm.Value	Carries optional confirm data.
ErrStatus	Error/return code, see Table P.1.

An SA-COMMAND.confirm shall be generated by the ITS-S application entity upon performance of a previous SA-COMMAND.request in case ErrStatus indicates error or SA-CmdConfirm is present. It may be generated in case ErrStatus indicates success or A-CmdConfirm is not present.

On receipt of SA-COMMAND.confirm, ErrStatus and SA-CmdConfirm shall be evaluated and a possible action shall be performed properly.

13.5 SA-REQUEST

13.5.1 SA-REQUESTs

Annex T provides an overview and coding details on SA-REQUESTs.

SA-REQUESTs shall be enabled by means of the service primitives SA-REQUEST.request and SA-REQUEST.confirm.

13.5.2 SA-REQUEST.request

The management service primitive SA-REQUEST.request allows the ITS-S application entity to trigger an action at the ITS-S management entity.

The parameters of SA-REQUEST.request are as follows:

```
SA-REQUEST.request  (
    CommandRef,
    SA-Request
)
```

Table 39 — SA-REQUEST.request Parameter description

Name	Description
CommandRef	Unique cyclic reference number of command.
SA-Request.No	See Annex T.
SA-Request.Value	See Annex T.

An SA-REQUEST.request shall be generated by the ITS-S application entity when the ITS-S management entity shall perform an action.

On receipt of SA-REQUEST.request the required action shall be performed.

13.5.3 SA-REQUEST.confirm

The service primitive SA-REQUEST.confirm reports the result of a previous SA-REQUEST.request.

The parameters of SA-REQUEST.confirm are as follows:

```
SA-REQUEST.confirm  (
    CommandRef,
    SA-ReqConfirm,
    ErrStatus
)
```

Table 40 — SA-REQUEST.confirm Parameter description

Name	Description
CommandRef	Unique cyclic reference number of command. Same value as in related SA-REQUEST.request.
SA-ReqConfirm.No	Reference number of command. Same value as SA-Request.No in related SA-REQUEST.request.
SA-ReqConfirm.Value	Carries optional confirm data.
ErrStatus	Error/return code, see Table P.1.

An SA-REQUEST.confirm shall be generated by the ITS-S management entity upon performance of a previous SA-REQUEST.request in case ErrStatus indicates error or SA-ReqConfirm is present. It may be generated in case ErrStatus indicates success or SA-ReqConfirm is not present.

On receipt of SA-REQUEST.confirm, ErrStatus and SA-ReqConfirm shall be evaluated and a possible action shall be performed properly. Details are outside the scope of this document.

Page 23, Conformance

Renumber existing Clause 13 to Clause 14.

Replace NOTE with the following:

NOTE Service access points may become observable and thus testable in PDUs for ITS station-internal management communications specified in ISO 24102-4.

Page 24, Annex A

Replace the whole Annex A with the following:

Annex A
(normative)
ASN.1 modules

A.1 Overview

The following ASN.1 module is specified in this annex:

— CALMmsap {ISO (1) standard (0) calm-management (24102) msap (3) asnm-1 (1)}.

A.2 Module CALMmsap

This module specifies ASN.1 type definitions together with useful ASN.1 value definitions.

Unaligned packed encoding rules (PER) as specified in ISO/IEC 8825-2 shall be applied for this ASN.1 module.

In order to achieve octet alignment and enabling cheap implementations, “fill” bits were defined. All fill bits shall be set to the value ‘0’b. In case fill bits precede a CHOICE type, the CHOICE tag shall be evaluated only in case all fill bits show the value ‘0’b in order to identify a conflict with newer versions of the specification providing more choices.

```
CALMmsap {iso (1) standard (0) calm-management (24102) msap (3) asnm-1 (1)}

DEFINITIONS AUTOMATIC TAGS ::= BEGIN

IMPORTS

CIclass, Cistatus, Errors, INsapPrimitivesDown, INsapPrimitivesUp, I-Param, I-ParamNo,
KineVectOut, Link-ID, MACaddress, MedID, MedType, UserPriority FROM CALMllsap {iso (1)
standard (0) calm-ll-sap (21218) asnm-1 (1)}

ITS-scuId, ITS-SSI FROM CALMmanagement {iso (1) standard (0) calm-management (24102) local
(1) asnm-1 (1)}

IICrequestTX, IICresponseTX, IICrequestRX, IICresponseRX FROM CALMiiTsscu {iso (1)
standard (0) calm-management (24102) iitsscu (4) asnm-1 (1)}

GCctx, GCdeleteCmd, GCregerServer, GCregerServerConf, GCsam, GCsamctx, GCsamctxConf,
GCctxTxCmd, GCperiodCmd, CTXrxNot, SAMrxNot, GCupdateServer, GCupdateServerConf,
GCderegServer, GCderegServerConf, GCregerClient, GCregerClientConf, GCupdateClient,
GCupdateClientConf, GCderegClient, GCderegClientConf FROM CALMfsap {iso (1) standard (0)
calm-management (24102) fsap (5) asnm-1 (1)}

SetIPv6, SetConfIPv6, UpdateIPv6, DeleteIPv6, SetNotIPv6, UpdateNotIPv6, DeleteNotIPv6
FROM ITSipv6 {iso (1) standard (0) its-ipv6 (21210) asnm-1 (1)}

HostServiceInfo, NFSapPrimitivesDown, NFSapPrimitivesUp, SetFNTP, SetConfFNTP, UpdateFNTP,
DeleteFNTP, SetNotFNTP, UpdateNotFNTP, DeleteNotFNTP FROM CALMfntp {iso (1) standard (0)
calm-nonip (29281) fntp (1) asnm-1 (1)}
```

ITS-S-Appl-Reg, ITS-S-Appl-RegConf, ITS-S-Appl-RegFinal FROM CITSapplReq {iso (1) standard
(0) cits-applReq (17423) asnm-1 (1)}

;

-- End of IMPORTS

-- GENERIC OBJECT CLASSES

-- Communication-SAP generic OBJECT CLASS

```
COMMUPDOWN::=CLASS {
    &primitiveRef INTEGER (0..255),
    &Primitive
}
```

-- FA-SAP services up and down

FASAPDOWN::=COMMUPDOWN

```
FAsapPrimitivesDown::=SEQUENCE{
    spRef          FASAPDOWN.&primitiveRef({FAsapspdown}),
    servPrimitive  FASAPDOWN.&Primitive({FAsapspdown}{@spRef})
}
```

```
FAsapspdown FASAPDOWN::={gCsamctxConfFA | gCregServerFA | gCupdateServerFA |
gCderegServerFA | gCregClientFA | gCupdateClientFA | gCderegClientFA, ...}
```

```
gCsamctxConfFA  FASAPDOWN::={&primitiveRef 2, &Primitive GCsamctxConf}
gCregServerFA   FASAPDOWN::={&primitiveRef 3, &Primitive GCregServer}
gCupdateServerFA FASAPDOWN::={&primitiveRef 4, &Primitive GCupdateServer}
gCderegServerFA FASAPDOWN::={&primitiveRef 5, &Primitive GCderegServer}
gCregClientFA   FASAPDOWN::={&primitiveRef 6, &Primitive GCregClient}
gCupdateClientFA FASAPDOWN::={&primitiveRef 7, &Primitive GCupdateClient}
gCderegClientFA FASAPDOWN::={&primitiveRef 8, &Primitive GCderegClient}
```

FASAPUP::=COMMUPDOWN

```
FAsapPrimitivesUp::=SEQUENCE{
    spRef          FASAPUP.&primitiveRef({FAsapspsup}),
    servPrimitive  FASAPUP.&Primitive({FAsapspsup}{@spRef})
}
```

```
FAsapspsup FASAPUP::={gCsamctxFA | gCregServerConfFA | gCupdateServerConfFA |
gCderegServerConfFA | gCregClientConfFA | gCupdateClientConfFA | gCderegClientConfFA |
gCsamFA | gCctxFA, ...}
```

```
gCsamctxFA      FASAPUP::={&primitiveRef 2, &Primitive GCsamctx}
gCregServerConfFA FASAPUP::={&primitiveRef 3, &Primitive GCregServerConf}
gCupdateServerConfFA FASAPUP::={&primitiveRef 4, &Primitive GCupdateServerConf}
gCderegServerConfFA FASAPUP::={&primitiveRef 5, &Primitive GCderegServerConf}
gCregClientConfFA FASAPUP::={&primitiveRef 6, &Primitive GCregClientConf}
gCupdateClientConfFA FASAPUP::={&primitiveRef 7, &Primitive GCupdateClientConf}
gCderegClientConfFA FASAPUP::={&primitiveRef 8, &Primitive GCderegClientConf}
gCsamFA         FASAPUP::={&primitiveRef 9, &Primitive GCsam}
gCctxFA         FASAPUP::={&primitiveRef 10, &Primitive GCctx}
```

-- MX-SAP generic OBJECT CLASS

```
MXSERV::=CLASS {
    &mxref INTEGER(0..255),
    &MXParam
}
```

-- MF-SAP Service primitives --

-- MF-SAP Command.request --

MFSA-PR::=MXSERV

MF-Command MFSAP-CR::={simFUTcmd | simFLTcmd | gCsamctx | gCsam | gCctx | lDMnotifyCmd | legacyCICmd | stateCInotify | gCperiodCmd | gCctxTxCmd | gCdeleteCmd | iICrequestTX | iICresponseTX | testMF, ...} -- insert simFUTcmd once being defined and activate below

```
MF-Command-request::=SEQUENCE{
    commandRef      CommandRef,
    ref              MFSAP-CR.&mxref({MF-Command}),
    command-param    MFSAP-CR.&MXParam({MF-Command}{@ref})
}
```

```
simFUTcmd      MFSAP-CR::={&mxref 0, &MXParam SimFUTcmd}
simFLTcmd      MFSAP-CR::={&mxref 1, &MXParam SimFLTcmd}
gCsamctx       MFSAP-CR::={&mxref 2, &MXParam GCsamctx}
gCsam          MFSAP-CR::={&mxref 3, &MXParam GCsam}
gCctx          MFSAP-CR::={&mxref 4, &MXParam GCctx}
lDMnotifyCmd   MFSAP-CR::={&mxref 5, &MXParam lDMnotify}
legacyCICmd    MFSAP-CR::={&mxref 6, &MXParam LegacyCIreq}
stateCInotify  MFSAP-CR::={&mxref 7, &MXParam StateCInotify}
gCperiodCmd    MFSAP-CR::={&mxref 8, &MXParam GCperiodCmd}
gCctxTxCmd     MFSAP-CR::={&mxref 9, &MXParam GCctxTxCmd}
gCdeleteCmd    MFSAP-CR::={&mxref 10, &MXParam GCdeleteCmd}
iICrequestTX   MFSAP-CR::={&mxref 11, &MXParam iICrequestTX}
iICresponseTX  MFSAP-CR::={&mxref 12, &MXParam iICresponseTX}
testMF         MFSAP-CR::={&mxref 255, &MXParam EchoTest}
```

SimFUTcmd::=FAsapPrimitivesDown

SimFLTcmd::=NFsapPrimitivesUp

lDMnotify::=SEQUENCE (SIZE(0..255)) OF RadarView

```
RadarView::=SEQUENCE{
    peerITS-SSI    ITS-SSI
}
```

```
LegacyCIreq::=SEQUENCE{
    cIclass        Ciclass,
    legacyOption    INTEGER(0..255),
    linkId          Link-ID
}
```

```
StateCInotify::=SEQUENCE{
    linkId          Link-ID,
    cIstatus        Cistatus
}
```

EchoTest::=OCTET STRING (SIZE(0..255))

-- MF-SAP Command.confirm --

MFSAP-CC::=MXSERV

MF-CmdConfirm MFSAP-CC::={simFUTcmdConf | simFLTcmdConf | gCsamctxConf | gCsamConf | gCctxTxConf | lDMnotifyConf | legacyCICConf | stateCInotifyConf | gCperiodCmdConf | gCdeleteCmdConf | iICrequestTXConf | iICresponseTXConf | testMFConf, ...}

```
MF-Command-confirm::=SEQUENCE{
    commandRef      CommandRef,
    ref              MFSAP-CC.&mxref({MF-CmdConfirm}),
    cmdConfirm-param MFSAP-CC.&MXParam({MF-CmdConfirm}{@ref}),
    errStatus        ErrStatus
}
```

```
simFUTcmdConf  MFSAP-CC::={&mxref 0, &MXParam NULL}
simFLTcmdConf  MFSAP-CC::={&mxref 1, &MXParam NULL}
gCsamctxConf   MFSAP-CC::={&mxref 2, &MXParam GCsamctxConf}
gCsamConf      MFSAP-CC::={&mxref 3, &MXParam NULL}
gCctxTxConf    MFSAP-CC::={&mxref 4, &MXParam NULL}
lDMnotifyConf  MFSAP-CC::={&mxref 5, &MXParam NULL}
legacyCICConf  MFSAP-CC::={&mxref 6, &MXParam HostServiceInfo}
stateCInotifyConf MFSAP-CC::={&mxref 7, &MXParam NULL}
```

```
gCperiodCmdConf    MFSAP-CC:={&mxref 8, &MXParam NULL}
gCctxTxCmdConf     MFSAP-CC:={&mxref 9, &MXParam NULL}
gCdeleteCmdConf    MFSAP-CC:={&mxref 10, &MXParam NULL}
iICrequestTXConf   MFSAP-CC:={&mxref 11, &MXParam NULL}
iICresponseTXConf  MFSAP-CC:={&mxref 12, &MXParam NULL}
testMFConf         MFSAP-CC:={&mxref 255, &MXParam NULL}
```

-- MF-SAP Request.request --

MFSAP-RR:==MXSERV

MF-Request MFSAP-RR:={simFUTreq | simFLTreq | ITS-S-Appl-Reg | gCregServer | gCupdateServer | gCderegServer | gCregClient | gCupdateClient | gCderegClient | LDMregister | sAMrxNot | cTXrxNot | iICrequestRX | iICresponseRX | ITS-S-Appl-ReqFinal | testMFecho, ...}

```
MF-Request-request:==SEQUENCE{
    commandRef    CommandRef,
    ref           MFSAP-RR.&mxref({MF-Request}),
    request-param MFSAP-RR.&MXParam({MF-Request}){@ref})
}
```

```
simFUTreq          MFSAP-RR:={&mxref 0, &MXParam SimFUTreq}
simFLTreq          MFSAP-RR:={&mxref 1, &MXParam SimFLTreq}
ITS-S-Appl-Reg     MFSAP-RR:={&mxref 2, &MXParam ITS-S-Appl-Reg}
gCregServer        MFSAP-RR:={&mxref 3, &MXParam GCregServer}
gCupdateServer     MFSAP-RR:={&mxref 4, &MXParam GCupdateServer}
gCderegServer      MFSAP-RR:={&mxref 5, &MXParam GCderegServer}
gCregClient        MFSAP-RR:={&mxref 6, &MXParam GCregClient}
gCupdateClient     MFSAP-RR:={&mxref 7, &MXParam GCupdateClient}
gCderegClient      MFSAP-RR:={&mxref 8, &MXParam GCderegClient}
LDMregister        MFSAP-RR:={&mxref 9, &MXParam LDMregister}
sAMrxNot           MFSAP-RR:={&mxref 10, &MXParam sAMrxNot}
cTXrxNot           MFSAP-RR:={&mxref 11, &MXParam cTXrxNot}
iICrequestRX       MFSAP-RR:={&mxref 12, &MXParam iICrequestRX}
iICresponseRX      MFSAP-RR:={&mxref 13, &MXParam iICresponseRX}
ITS-S-Appl-ReqFinal MFSAP-RR:={&mxref 14, &MXParam ITS-S-Appl-ReqFinal}
testMFecho         MFSAP-RR:={&mxref 255, &MXParam TestMFecho}
```

SimFUTreq:==FasapPrimitivesUp

SimFLTreq:==NfsapPrimitivesDown

```
LDMregister:==SEQUENCE{
    ITS-scuId    ITS-scuId,
    reference    OCTET STRING (SIZE(0..65535))
}
```

```
TestMFecho:==SEQUENCE{
    sap    INTEGER{1 (70)} (0..255), -- indicating MF-SAP
    info  EchoTest
}
```

-- MF-SAP Request.confirm --

MFSAP-RC:==MXSERV

MF-ReqConfirm MFSAP-RC:={simFUTreqConf | simFLTreqConf | ITS-S-Appl-RegConf | gCregServerConf | gCupdateServerConf | gCderegServerConf | gCregClientConf | gCupdateClientConf | gCderegClientConf | LDMregisterConf | sAMrxNotConf | cTXrxNotConf | iICrequestRXConf | iICresponseRXConf | ITS-S-Appl-ReqFinalConf | testMFechoConf, ...}

```
MF-Request-confirm:==SEQUENCE{
    commandRef    CommandRef,
    ref           MFSAP-RC.&mxref({MF-ReqConfirm}),
    reqConfirm-param MFSAP-RC.&MXParam({MF-ReqConfirm}){@ref}),
    errStatus     ErrStatus
}
```

```
simFUTreqConf      MFSAP-RC:={&mxref 0, &MXParam NULL}
simFLTreqConf      MFSAP-RC:={&mxref 1, &MXParam NULL}
```

```

iTS-S-Appl-RegConf      MFSAP-RC::={&mxref 2, &MXParam NULL}
gCregServerConf         MFSAP-RC::={&mxref 3, &MXParam GCreateServerConf}
gCupdateServerConf      MFSAP-RC::={&mxref 4, &MXParam GCreateServerConf}
gCderegServerConf       MFSAP-RC::={&mxref 5, &MXParam GCreateServerConf}
gCregClientConf         MFSAP-RC::={&mxref 6, &MXParam GCreateClientConf}
gCupdateClientConf      MFSAP-RC::={&mxref 7, &MXParam GCreateClientConf}
gCderegClientConf       MFSAP-RC::={&mxref 8, &MXParam GCreateClientConf}
ldMregisterConf         MFSAP-RC::={&mxref 9, &MXParam ITS-scuId}
sAMrxNotConf           MFSAP-RC::={&mxref 10, &MXParam NULL}
cTXrxNotConf           MFSAP-RC::={&mxref 11, &MXParam NULL}
iICrequestRXConf        MFSAP-RC::={&mxref 12, &MXParam NULL}
iICresponseRXConf       MFSAP-RC::={&mxref 13, &MXParam NULL}
iTS-S-Appl-ReqFinalConf MFSAP-RC::={&mxref 14, &MXParam NULL}
testMFEchoConf          MFSAP-RC::={&mxref 255, &MXParam NULL}

-- MN-SAP Service primitives --
-- MN-SAP Command.request --

MNSAP-CR::=MXSERV

MN-Command MNSAP-CR::={simNUTcmd | simNLTcmd | fWTset | fWTupdate | fWTdelete | testMN,
...}

MN-Command-request::=SEQUENCE{
    commandRef      CommandRef,
    ref              MNSAP-CR.&mxref({MN-Command}),
    command-param    MNSAP-CR.&MXParam({MN-Command}){@ref}
}

simNUTcmd      MNSAP-CR::={&mxref 0, &MXParam SimNUTcmd}
simNLTcmd      MNSAP-CR::={&mxref 1, &MXParam SimNLTcmd}
fWTset         MNSAP-CR::={&mxref 2, &MXParam FWTset}
fWTupdate      MNSAP-CR::={&mxref 3, &MXParam FWTupdate}
fWTdelete      MNSAP-CR::={&mxref 4, &MXParam FWTdelete}
testMN         MNSAP-CR::={&mxref 255, &MXParam EchoTest}

SimNUTcmd::=NFsapPrimitivesDown
SimNLTcmd::=INSapPrimitivesUp

FWT::=CLASS {
    &fwtRef INTEGER(0..255),
    &Fwt
}

FWTset::=SEQUENCE{
    fwtNo      FWT.&fwtRef({NTprotsSet}),
    fwt        FWT.&Fwt({NTprotsSet}){@fwtNo}
}

NTprotsSet FWT::={fntpsset, ...}

FWTupdate::=SEQUENCE{
    fwtNo      FWT.&fwtRef({NTprotsUpdate}),
    fwt        FWT.&Fwt({NTprotsUpdate}){@fwtNo}
}

NTprotsUpdate FWT::={fntpupdate, ...}

FWTdelete::=SEQUENCE{
    fwtNo      FWT.&fwtRef({NTprotsDelete}),
    fwt        FWT.&Fwt({NTprotsDelete}){@fwtNo}
}

NTprotsDelete FWT::={fntpdelete, ...}

fntpsset      FWT::={&fwtRef 0, &Fwt SetFNTp}
fntpupdate    FWT::={&fwtRef 1, &Fwt UpdateFNTp}
fntpdelete    FWT::={&fwtRef 2, &Fwt DeleteFNTp}

-- MN-SAP Command.confirm --

```

MNSAP-CC::=MXSERV

MN-CmdConfirm MNSAP-CC::={simNUTcmdConf | simNLtcmdConf | fWTsetConf | fWTupdateConf | fWTdeleteConf | testMNConf, ...}

MN-Command-confirm::=SEQUENCE{
 commandRef CommandRef,
 ref MNSAP-CC.&mxref ({MN-CmdConfirm}),
 cmdConfirm-param MNSAP-CC.&MXParam ({MN-CmdConfirm} {@ref}),
 errStatus ErrStatus
 }

simNUTcmdConf MNSAP-CC::={&mxref 0, &MXParam NULL}
 simNLtcmdConf MNSAP-CC::={&mxref 1, &MXParam NULL}
 fWTsetConf MNSAP-CC::={&mxref 2, &MXParam FWTsetConf}
 fWTupdateConf MNSAP-CC::={&mxref 3, &MXParam FWTupdateConf}
 fWTdeleteConf MNSAP-CC::={&mxref 4, &MXParam FWTdeleteConf}
 testMNConf MNSAP-CC::={&mxref 255, &MXParam NULL}

FWTCONF::=CLASS {
 &fwtRef INTEGER(0..255),
 &Fwt
 }

FWTsetConf::=SEQUENCE{
 fwtNo FWTCONF.&fwtRef ({NTprotsSetConf}),
 fwt FWTCONF.&Fwt ({NTprotsSetConf} {@fwtNo})
 }

NTprotsSetConf FWTCONF::={fntpsetConf, ...}

FWTupdateConf::=SEQUENCE{
 fwtNo FWTCONF.&fwtRef ({NTprotsUpdateConf}),
 fwt FWTCONF.&Fwt ({NTprotsUpdateConf} {@fwtNo})
 }

NTprotsUpdateConf FWTCONF::={fntpupdateConf, ...}

FWTdeleteConf::=SEQUENCE{
 fwtNo FWTCONF.&fwtRef ({NTprotsDeleteConf}),
 fwt FWTCONF.&Fwt ({NTprotsDeleteConf} {@fwtNo})
 }

NTprotsDeleteConf FWTCONF::={fntpdeleteConf, ...}

fntpsetConf FWTCONF::={&fwtRef 0, &Fwt SetConfFNTp}
 fntpupdateConf FWTCONF::={&fwtRef 1, &Fwt NULL}
 fntpdeleteConf FWTCONF::={&fwtRef 2, &Fwt NULL}

-- MN-SAP Request request --

MNSAP-RR::=MXSERV

MN-Request MNSAP-RR::={simNUTreq | simNLtreq | fWTsetNot | fWTupdateNot | fWTdeleteNot | vCIcreatePeerMAC | its-ssiPeerNot | testMNecho, ...}

MN-Request-request::=SEQUENCE{
 commandRef CommandRef,
 ref MNSAP-RR.&mxref ({MN-Request}),
 request-param MNSAP-RR.&MXParam ({MN-Request} {@ref})
 }

simNUTreq MNSAP-RR::={&mxref 0, &MXParam SimNUTreq}
 simNLtreq MNSAP-RR::={&mxref 1, &MXParam SimNLtreq}
 fWTsetNot MNSAP-RR::={&mxref 2, &MXParam FWTsetNot}
 fWTupdateNot MNSAP-RR::={&mxref 3, &MXParam FWTupdateNot}
 fWTdeleteNot MNSAP-RR::={&mxref 4, &MXParam FWTdeleteNot}
 vCIcreatePeerMAC MNSAP-RR::={&mxref 5, &MXParam VCIcreatePeerMAC}
 its-ssiPeerNot MNSAP-RR::={&mxref 6, &MXParam Its-ssiPeerNot}
 testMNecho MNSAP-RR::={&mxref 255, &MXParam TestMNecho}

```

SimNUTreq::=NFsapPrimitivesUp

SimNLTreq::=INSapPrimitivesDown

FWTNOT::=CLASS {
    &fwtRef  INTEGER(0..255),
    &Fwt
}

FWTsetNot::=SEQUENCE{
    fwtNo      FWTNOT.&fwtRef({NTprotsSetNot}),
    fwt        FWTNOT.&Fwt({NTprotsSetNot}{@fwtNo})
}

NTprotsSetNot  FWTNOT::={fntpsetNot, ...}

FWTupdateNot::=SEQUENCE{
    fwtNo      FWTNOT.&fwtRef({NTprotsUpdateNot}),
    fwt        FWTNOT.&Fwt({NTprotsUpdateNot}{@fwtNo})
}

NTprotsUpdateNot  FWTNOT::={fntpupdateNot, ...}

FWTdeleteNot::=SEQUENCE{
    fwtNo      FWTNOT.&fwtRef({NTprotsDeleteNot}),
    fwt        FWTNOT.&Fwt({NTprotsDeleteNot}{@fwtNo})
}

NTprotsDeleteNot  FWTNOT::={fntpdeleteNot, ...}

fntpsetNot      FWTNOT::={&fwtRef 0, &Fwt SetNotFNTTP}
fntpupdateNot   FWTNOT::={&fwtRef 1, &Fwt UpdateNotFNTTP}
fntpdeleteNot   FWTNOT::={&fwtRef 2, &Fwt DeleteNotFNTTP}

VCicreatePeerMAC::=SEQUENCE{
    reference    INTEGER(0..255),
    linkId       Link-ID,
    peerMac      MACaddress
}

Its-ssiPeerNot::=SEQUENCE{
    sap          INTEGER(0..255),
    mACaddress    MACaddress,
    linkId        Link-ID,
    its-ssiData   ITS-SSI
}

StationType::=INTEGER{
    mobile        (0),
    fixed          (1),
    infrastructure (2),
    vehicle        (3),
    roadside       (4),
    central        (5),
    portable       (6),
    unknown        (255)
} (0..255)

StationID::=OCTET STRING (SIZE(4))

TestMNecho::=SEQUENCE{
    sap          INTEGER{n (78)} (0..255), -- indicating MN-SAP
    info         EchoTest
}

-- MN-SAP Request.confirm --

MNSAP-RC::=MXSERV

```

MN-ReqConfirm MNSAP-RC::={simNUTreqConf | simNLTreqConf | fWTsetNotConf | fWTupdateNotConf | fWTdeleteNotConf | vCIcreatePeerMACConf | its-ssiPeerNotConf | testMNechoConf, ...}

```
MN-Request-confirm::=SEQUENCE{
    commandRef      CommandRef,
    ref              MNSAP-RC.&mxref({MN-ReqConfirm}),
    reqConfirm-param MNSAP-RC.&MXParam({MN-ReqConfirm}{@ref}),
    errStatus        ErrStatus
}
```

```
simNUTreqConf      MNSAP-RC::={&mxref 0, &MXParam NULL}
simNLTreqConf      MNSAP-RC::={&mxref 1, &MXParam NULL}
fWTsetNotConf      MNSAP-RC::={&mxref 2, &MXParam NULL}
fWTupdateNotConf   MNSAP-RC::={&mxref 3, &MXParam NULL}
fWTdeleteNotConf   MNSAP-RC::={&mxref 4, &MXParam NULL}
vCIcreatePeerMACConf MNSAP-RC::={&mxref 5, &MXParam VCIpeerMAC}
its-ssiPeerNotConf MNSAP-RC::={&mxref 6, &MXParam NULL}
testMNechoConf     MNSAP-RC::={&mxref 255, &MXParam NULL}
```

```
VCIpeerMAC::=SEQUENCE{
    reference      INTEGER(0..255),
    linkId         Link-ID
}
```

-- MI-SAP Service primitives --

-- MI-SAP Command.request --

MISAP-CR::=MXSERV

MI-Command MISAP-CR::={simIUTcmd | regCmd | cIstate | wakeUp | rTScmd | rTSackCmd | cONcmd | rIcmd | manuCmd | vciCmd | monitor | unitDataCmd | testMI, ...}

```
MI-Command-request::=SEQUENCE{
    linkID          Link-ID,
    commandRef      CommandRef,
    ref              MISAP-CR.&mxref({MI-Command}),
    command-param   MISAP-CR.&MXParam({MI-Command}{@ref})
}
```

```
simIUTcmd      MISAP-CR::={&mxref 0, &MXParam SimIUTcmd}
regCmd         MISAP-CR::={&mxref 1, &MXParam RegCmd}
cIstate        MISAP-CR::={&mxref 2, &MXParam CIstateChng}
wakeUp         MISAP-CR::={&mxref 3, &MXParam WakeUp}
rTScmd         MISAP-CR::={&mxref 4, &MXParam RTScmd}
rTSackCmd      MISAP-CR::={&mxref 5, &MXParam RTSackCmd}
cONcmd         MISAP-CR::={&mxref 6, &MXParam CONcmd}
rIcmd          MISAP-CR::={&mxref 7, &MXParam RIcmd}
manuCmd        MISAP-CR::={&mxref 8, &MXParam OCTET STRING}
vciCmd         MISAP-CR::={&mxref 9, &MXParam VciCmd}
monitor        MISAP-CR::={&mxref 10, &MXParam Monitor}
unitDataCmd    MISAP-CR::={&mxref 11, &MXParam UnitData}
testMI         MISAP-CR::={&mxref 255, &MXParam EchoTest}
```

SimIUTcmd::=INsapPrimitivesDown

```
RegCmd::=SEQUENCE{
    scuId          ITS-scuId,
    medID          MedID
}
```

```
CIstateChng::=INTEGER{
    deregister      (0),
    activate        (4),
    resume          (8),
    connect         (16),
    disconnect      (32),
    suspend         (64),
    inactivate      (128)
} (0..255)
```

```

WakeUp ::= INTEGER (0..255)

RTScmd ::= SEQUENCE {
    reqID      ReqID,
    priority   UserPriority,
    seqNo      INTEGER (0..255),
    status     INTEGER {
        release      (0), -- release prioritization
        request      (16) -- request prioritization
    } (0..255)
}

ReqID ::= SEQUENCE {
    linkID Link-ID
}

RTSackCmd ::= SEQUENCE {
    priority   UserPriority,
    seqNo      INTEGER (0..255),
    status     INTEGER {
        ignored      (64), -- request ignored
        granted      (128) -- request granted
    } (0..255)
}

CONcmd ::= INTEGER {
    deleteAC      (0),
    connect        (1),
    disconnect     (255)
} (0..255)

RICmd ::= SEQUENCE {
    linkID Link-ID,
    ri      RI
}

RI ::= OCTET STRING (SIZE (0..65535))

VciCmd ::= SEQUENCE (SIZE (0..65535)) OF SEQUENCE {
    fill      BIT STRING (SIZE (7)),
    linkID    Link-ID,
    alive     BOOLEAN OPTIONAL
}

Monitor ::= SEQUENCE (SIZE (0..255)) OF SEQUENCE {
    paramNo    INTEGER (0..255), -- valid parameter number
    active     INTEGER {
        stop      (0),
        start     (255)
    } (0..255)
}

UnitData ::= SEQUENCE {
    sourceAddr Link-ID,
    destAddr   Link-ID,
    data       OCTET STRING (SIZE (0..65535)),
    priority   UserPriority,
    parameter  OCTET STRING (SIZE (0..65535)) -- tbd dependent on medium
}

-- MI-SAP Command.Confirm --

MI-Command-confirm ::= SEQUENCE {
    linkID      Link-ID,
    commandRef  CommandRef,
    errStatus   ErrStatus
}

-- MI-SAP Request.request --

```

MISAP-RR::=MXSERV

MI-Request MISAP-RR::={simIUTreq | regReq | prioReg | rTSreq | rTSackReq | rIreq | manuReq
| events | posUpdateReq | unitDataReq | testMIecho, ...}

MI-Request-request::=SEQUENCE{
linkID Link-ID,
commandRef CommandRef,
ref MISAP-RR.&mxref({MI-Request}),
request-param MISAP-RR.&MXParam({MI-Request}){@ref}}
}

simIUTreq MISAP-RR::={&mxref 0, &MXParam SimIUTreq}
regReq MISAP-RR::={&mxref 1, &MXParam RegReq}
prioReg MISAP-RR::={&mxref 2, &MXParam PrioReg}
rTSreq MISAP-RR::={&mxref 3, &MXParam RTSreq}
rTSackReq MISAP-RR::={&mxref 4, &MXParam RTSackReq}
rIreq MISAP-RR::={&mxref 5, &MXParam RIreq}
manuReq MISAP-RR::={&mxref 6, &MXParam OCTET STRING}
events MISAP-RR::={&mxref 7, &MXParam Events21218}
posUpdateReq MISAP-RR::={&mxref 8, &MXParam PosUpdateReq}
unitDataReq MISAP-RR::={&mxref 9, &MXParam UnitData}
testMIecho MISAP-RR::={&mxref 255, &MXParam TestMIecho}

SimIUTreq::=INsapPrimitivesUp

RegReq::=SEQUENCE{
medType MedType
}

PrioReg::=SEQUENCE{
interferers Interferers,
timeout INTEGER(0..255)
}

Interferers::=SEQUENCE (SIZE(0..255)) OF MedType

RTSreq::=SEQUENCE{
priority UserPriority,
seqNo INTEGER(0..255),
status INTEGER {
release (0) -- release prioritization
request (16) -- request prioritization
} (0..255)
}

RTSackReq::=SEQUENCE{
reqID ReqID,
seqNo INTEGER(0..255),
status INTEGER {
ignored (64), -- request ignored
granted (128) -- request granted
} (0..255)
}

RIreq::=SEQUENCE{
medType MedType, -- medium for retrieval
riAccess RIaccess -- request frame details
}

RIaccess::=OCTET STRING (SIZE(0..65535))

EVENT21218::=CLASS {
&eventRef INTEGER(0..255),
&Event21218
}

Events21218::=SEQUENCE{
eventNo EVENT21218.&eventRef({Events-21218}),
event EVENT21218.&Event21218({Events-21218} {@eventNo})
}

```

Events-21218 EVENT21218:={minUserPrio | txQueueThreshold | txQueueFull | vciCreated |
vciDeleted | paramMonitor | txQueueLow | vciReset, ...}

minUserPrio      EVENT21218:={&eventRef 0, &Event21218 E21218-0}
-- MinimumUserPriority
txQueueThreshold EVENT21218:={&eventRef 1, &Event21218 E21218-1}
-- TX queue at threshold
txQueueFull      EVENT21218:={&eventRef 2, &Event21218 E21218-2} -- TX queue full
vciCreated       EVENT21218:={&eventRef 3, &Event21218 E21218-3} -- VCI created
vciDeleted       EVENT21218:={&eventRef 4, &Event21218 E21218-4} -- VCI deleted
paramMonitor     EVENT21218:={&eventRef 5, &Event21218 E21218-5}
-- Automatic notification
txQueueLow       EVENT21218:={&eventRef 6, &Event21218 E21218-6}
-- TX queue below low thresh.
vciReset         EVENT21218:={&eventRef 7, &Event21218 E21218-7} -- VCI reset

E21218-0::=SEQUENCE{
    priority      UserPriority,
    linkID        Link-ID
}

E21218-1::=UserPriority
E21218-2::=UserPriority
E21218-3::=Link-ID
E21218-4::=Link-ID
E21218-5::=I-Param
E21218-6::=UserPriority
E21218-7::=Link-ID

PosUpdateReq::=INTEGER(0..65535) -- 0: Update disabled
-- >0: Update interval in ms

TestMIEcho::=SEQUENCE{
    sap          INTEGER{1 (73)} (0..255), -- indicating MI-SAP
    info         EchoTest
}

-- MI-SAP Request.confirm --

MI-Request-confirm::=SEQUENCE{
    linkID        Link-ID,
    commandRef     CommandRef,
    errStatus      ErrStatus
}

-- MI-SAP Set.request --

MI-Set-request::=SEQUENCE{
    linkID        Link-ID,
    commandRef     CommandRef,
    set-param      IParamList
}

IParamList::=SEQUENCE OF I-Param

-- MI-SAP Set.confirm --

MI-Set-confirm::=SEQUENCE{
    linkID        Link-ID,
    commandRef     CommandRef,
    errors         IErrorsList
}

```

IErrorsList::=SEQUENCE OF Errors

-- MI-SAP Get.request --

```
MI-Get-request::=SEQUENCE{
    linkID      Link-ID,
    commandRef  CommandRef,
    get-param-no IParamNoList
}
```

IParamNoList::=SEQUENCE OF I-ParamNo

-- MI-SAP Get.confirm --

```
MI-Get-confirm::=SEQUENCE{
    linkID      Link-ID,
    commandRef  CommandRef,
    get-param   IParamList
}
```

-- SF-SAP Service primitives --

-- SF-SAP Command.request --

-- activate the following once an SF-Command is defined
-- SFSAP-CR::=MXSERV

```
--SF-Command-request::=SEQUENCE{
--    commandRef      CommandRef,
--    ref              SFSAP-CR.&mxref({SF-Command}),
--    command-param    SFSAP-CR.&MXParam({SF-Command}{@ref})
--}
```

-- SF-Command SFSAP-CR::={, ...}

-- SF-SAP Command.confirm --

-- activate the following once an SF-CmdConfirm is defined
-- SFSAP-CC::=MXSERV

```
-- SF-Command-confirm::=SEQUENCE{
--    commandRef      CommandRef,
--    ref              SFSAP-CC.&mxref({SF-CmdConfirm}),
--    cmdConfirm-param SFSAP-CC.&MXParam({SF-CmdConfirm}{@ref}),
--    errStatus        ErrStatus
--}
```

-- SF-CmdConfirm SFSAP-CC::={, ...}

-- SF-SAP Request.request --

-- activate the following once an SF-Request is defined
-- SFSAP-RR::=MXSERV

```
-- SF-Request-request::=SEQUENCE{
--    commandRef      CommandRef,
--    ref              SFSAP-RR.&mxref({SF-Request}),
--    request-param    SFSAP-RR.&MXParam({SF-Request}{@ref})
--}
```

-- SF-Request SFSAP-RR::={, ...}

-- SF-SAP Request.confirm --

-- activate the following once an SN-ReqConfirm is defined
-- SFSAP-RC::=MXSERV

```
-- SF-Request-confirm::=SEQUENCE{
--    commandRef      CommandRef,
```

```

--      ref                SFSAP-RC.&mxref({SF-ReqConfirm}),
--      reqConfirm-param SFSAP-RC.&MXParam({SF-ReqConfirm}{@ref}),
--      errStatus        ErrStatus
--    }
--
-- SF-ReqConfirm SFSAP-RC::={, ...}

-- SN-SAP Service primitives --
-- SN-SAP Command.request --

-- activate the following once an SN-Command is defined
-- SNSAP-CR::=MXSERV
--
-- SN-Command-request::=SEQUENCE{
--   commandRef        CommandRef,
--   ref                SNSAP-CR.&mxref({SN-Command}),
--   command-param     SNSAP-CR.&MXParam({SN-Command}{@ref})
-- }
-- SN-Command SNSAP-CR::={, ...}

-- SN-SAP Command.confirm --

-- activate the following once an SN-CmdConfirm is defined
-- SNSAP-CC::=MXSERV
--
-- SN-Command-confirm::=SEQUENCE{
--   commandRef        CommandRef,
--   ref                SNSAP-CC.&mxref({SN-CmdConfirm}),
--   cmdConfirm-param  SNSAP-CC.&MXParam({SN-CmdConfirm}{@ref}),
--   errStatus        ErrStatus
-- }
--
-- SN-CmdConfirm SNSAP-CC::={, ...}

-- SN-SAP Request.request --

-- activate the following once an SN-Request is defined
-- SNSAP-RR::=MXSERV
--
-- SN-Request-request::=SEQUENCE{
--   commandRef        CommandRef,
--   ref                SNSAP-RR.&mxref({SN-Request}),
--   request-param     SNSAP-RR.&MXParam({SN-Request}{@ref})
-- }
--
-- SN-Request SNSAP-RR::={, ...}

-- SN-SAP Request.confirm --

-- activate the following once an SN-ReqConfirm is defined
-- SNSAP-RC::=MXSERV
--
-- SN-Request-confirm::=SEQUENCE{
--   commandRef        CommandRef,
--   ref                SNSAP-RC.&mxref({SN-ReqConfirm}),
--   reqConfirm-param  SNSAP-RC.&MXParam({SN-ReqConfirm}{@ref}),
--   errStatus        ErrStatus
-- }
--
-- SN-ReqConfirm SNSAP-RC::={, ...}

-- SI-SAP Service primitives --
-- SI-SAP Command.request --

-- activate the following once an SI-Command is defined
-- SISAP-CR::=MXSERV

```

```
--
-- SI-Command-request ::= SEQUENCE {
--     linkID          Link-ID,
--     commandRef      CommandRef,
--     ref             SISAP-CR.&mxref ({SI-Command}),
--     command-param   SISAP-CR.&MXParam ({SI-Command} {@ref})
-- }
--
-- SI-Command SISAP-CR ::= {, ...}

-- SI-SAP Command.confirm --

-- activate the following once an SI-CmdConfirm is defined
-- SISAP-CC ::= MXSERV
--
-- SI-Command-confirm ::= SEQUENCE {
--     linkID          Link-ID,
--     commandRef      CommandRef,
--     ref             SISAP-CC.&mxref ({SI-CmdConfirm}),
--     cmdConfirm-param SISAP-CC.&MXParam ({SI-CmdConfirm} {@ref}),
--     errStatus       ErrStatus
-- }
--
-- SI-CmdConfirm SISAP-CC ::= {, ...}

-- SI-SAP Request.request --

-- activate the following once an SI-Request is defined
-- SISAP-RR ::= MXSERV
--
-- SI-Request-request ::= SEQUENCE {
--     linkID          Link-ID,
--     commandRef      CommandRef,
--     ref             SISAP-RR.&mxref ({SI-Request}),
--     request-param   SISAP-RR.&MXParam ({SI-Request} {@ref})
-- }
--
-- SI-Request SISAP-RR ::= {, ...}

-- SI-SAP Request.confirm --

-- activate the following once an SI-CRqConfirm is defined
-- SISAP-RC ::= MXSERV
--
-- SI-Request-confirm ::= SEQUENCE {
--     linkID          Link-ID,
--     commandRef      CommandRef,
--     ref             SISAP-RC.&mxref ({SI-CmdConfirm}),
--     reqConfirm-param SISAP-RC.&MXParam ({SI-CmdConfirm} {@ref}),
--     errStatus       ErrStatus
-- }
--
-- SI-ReqConfirm SISAP-RC ::= {, ...}

-- MS-SAP Service primitives --
-- MS-SAP Command.request --

MSSAP-CR ::= MXSERV

MS-Command-request ::= SEQUENCE {
    commandRef      CommandRef,
    ref             MSSAP-CR.&mxref ({MS-Command}),
    command-param   MSSAP-CR.&MXParam ({MS-Command} {@ref})
}

MS-Command MSSAP-CR ::= {testMS, ...}

testMS MSSAP-CR ::= {&mxref 255, &MXParam EchoTest}
```

-- MS-SAP Command.confirm --

MSSAP-CC::=MXSERV

```

MS-Command-confirm:=SEQUENCE{
    commandRef      CommandRef,
    ref              MSSAP-CC.&mxref({MS-CmdConfirm}),
    cmdConfirm-param MSSAP-CC.&MXParam({MS-CmdConfirm}{@ref}),
    errStatus        ErrStatus
}

```

MS-CmdConfirm MSSAP-CC::={testMSConf, ...}

testMSConf MSSAP-CC::={&mxref 255, &MXParam NULL}

-- MS-SAP Request.request --

MSSAP-RR::=MXSERV

```

MS-Request-request:=SEQUENCE{
    commandRef      CommandRef,
    ref              MSSAP-RR.&mxref({MS-Request}),
    request-param    MSSAP-RR.&MXParam({MS-Request}{@ref})
}

```

```

MS-Request          MSSAP-RR::={ITS-S-Appl-ProcS-Reg | ITS-S-Appl-ProcS-ReqFinal |
testMSecho, ...}

```

```

ITS-S-Appl-ProcS-Reg    MSSAP-RR::={&mxref 1, &MXParam ITS-S-Appl-Reg}
ITS-S-Appl-ProcS-ReqFinal MSSAP-RR::={&mxref 13, &MXParam ITS-S-Appl-ReqFinal}
testMSecho              MSSAP-RR::={&mxref 255, &MXParam TestMSecho}

```

```

TestMSecho:=SEQUENCE{
    sap      INTEGER{s (83)} (0..255), -- indicating MS-SAP
    info     EchoTest
}

```

-- MS-SAP Request.confirm --

MSSAP-RC::=MXSERV

```

MS-Request-confirm:=SEQUENCE{
    commandRef      CommandRef,
    ref              MSSAP-RC.&mxref({MS-ReqConfirm}),
    reqConfirm-param MSSAP-RC.&MXParam({MS-ReqConfirm}{@ref}),
    errStatus        ErrStatus
}

```

```

MS-ReqConfirm          MSSAP-RC::={ITS-S-Appl-ProcS-RegConf | ITS-S-Appl-ProcS-ReqFinalConf |
testMSechoConf, ...}

```

```

ITS-S-Appl-ProcS-RegConf    MSSAP-RR::={&mxref 1, &MXParam NULL}
ITS-S-Appl-ProcS-ReqFinalConf MSSAP-RR::={&mxref 13, &MXParam NULL}
testMSechoConf              MSSAP-RR::={&mxref 255, &MXParam NULL}

```

-- MA-SAP service primitives**-- MA-SAP Command.request --**

MASAP-CR::=MXSERV

```

MA-Command-request:=SEQUENCE{
    commandRef      CommandRef,
    ref              MASAP-CR.&mxref({MA-Command}),
    command-param    MASAP-CR.&MXParam({MA-Command}{@ref})
}

```

MA-Command MASAP-CR::={testMA, ...}

```

testMA          MASAP-CR:={&mxref 255, &MXParam EchoTest}

-- MA-SAP Command.confirm --

MASAP-CC:={MXSERV

testMAConf  MASAP-CC:={&mxref 255, &MXParam NULL}

MA-Command-confirm:={SEQUENCE{
    commandRef      CommandRef,
    ref              MASAP-CC.&mxref({MA-CmdConfirm}),
    cmdConfirm-param MASAP-CC.&MXParam({MA-CmdConfirm}{@ref}),
    errStatus        ErrStatus
}

MA-CmdConfirm MASAP-CC:={testMAConf, ...}

-- MA-SAP Request.request --

MASAP-RR:={MXSERV

MA-Request-request:={SEQUENCE{
    commandRef      CommandRef,
    ref              MASAP-RR.&mxref({MA-Request}),
    request-param    MASAP-RR.&MXParam({MA-Request}{@ref})
}

MA-Request MASAP-RR:={ITS-S-Appl-Proc-Reg | ITS-S-Appl-Proc-ReqFinal | testMAecho, ...}

ITS-S-Appl-Proc-Reg      MASAP-RR:={&mxref 1, &MXParam ITS-S-Appl-Reg}
ITS-S-Appl-Proc-ReqFinal MASAP-RR:={&mxref 13, &MXParam ITS-S-Appl-ReqFinal}
testMAecho                MASAP-RR:={&mxref 255, &MXParam TestMAecho}

TestMAecho:={SEQUENCE{
    sap      INTEGER{a (65)} (0..255), -- indicating MA-SAP
    info     EchoTest
}

-- MA-SAP Request.confirm --

MASAP-RC:={MXSERV

MA-Request-confirm:={SEQUENCE{
    commandRef      CommandRef,
    ref              MASAP-RC.&mxref({MA-ReqConfirm}),
    reqConfirm-param MASAP-RC.&MXParam({MA-ReqConfirm}{@ref}),
    errStatus        ErrStatus
}

MA-ReqConfirm MASAP-RC:={ITS-S-Appl-Proc-RegConf | ITS-S-Appl-Proc-ReqFinalConf |
testMAechoConf, ...}

ITS-S-Appl-Proc-RegConf      MASAP-RC:={&mxref 1, &MXParam ITS-S-Appl-RegConf}
ITS-S-Appl-Proc-ReqFinalConf MASAP-RC:={&mxref 13, &MXParam NULL}
testMAechoConf              MASAP-RC:={&mxref 255, &MXParam NULL}

-- SA-SAP service primitives
-- SA-SAP Command.request --

-- activate the following once an SA-Command is defined
-- SASAP-CR:={MXSERV
--
-- SA-Command-request:={SEQUENCE{
--     commandRef      CommandRef,
--     ref              SASAP-CR.&mxref({SA-Command}),
--     command-param    SASAP-CR.&MXParam({SA-Command}{@ref})
-- }
--
-- SA-Command SASAP-CR:={, ...}

```

-- SA-SAP Command.confirm --

```
-- activate the following once an SA-CmdConfirm is defined
-- SASAP-CC::=MXSERV
--
-- SA-Command-confirm::=SEQUENCE{
--   commandRef      CommandRef,
--   ref              SASAP-CC.&mxref({SA-CmdConfirm}),
--   cmdConfirm-param SASAP-CC.&MXParam({SA-CmdConfirm}{@ref}),
--   errStatus        ErrStatus
-- }
--
-- SA-CmdConfirm SASAP-CC::={, ...}
```

-- SA-SAP Request.request --

```
-- activate the following once an SA-Request is defined
-- SASAP-RR::=MXSERV
--
-- SA-Request-request::=SEQUENCE{
--   commandRef      CommandRef,
--   ref              SASAP-RR.&mxref({SA-Request}),
--   request-param    SASAP-RR.&MXParam({SA-Request}{@ref})
-- }
--
-- SA-Request MASAP-RR::={, ...}
```

-- SA-SAP Request.confirm --

```
-- activate the following once an SA-ReqConfirm is defined
-- SASAP-RC::=MXSERV
--
-- SA-Request-confirm::=SEQUENCE{
--   commandRef      CommandRef,
--   ref              SASAP-RC.&mxref({SA-Request-confirm}),
--   reqConfirm-param SASAP-RC.&MXParam({SA-Request-confirm}{@ref}),
--   errStatus        ErrStatus
-- }
--
-- SA-ReqConfirm SASAP-RC::={, ...}
```

-- General types --

```
ErrStatus::=INTEGER{
  success          (0),
  unspecFailure    (1), -- unspecified failure
  invalParamNo     (2), -- invalid param no.
  invalParamVal    (3), -- invalid param value
  riViolation       (4), -- RI violation
  cmdReqNo         (5), -- invalid cmd/req no.
  cmdReqVal        (6), -- invalid cmd/req value
  accessViolation  (7), -- access violation
  invalidType      (8), -- invalid cmd/req type
  sequenceError    (9), -- cmd/req not applicable in the given context
  nonavailValue    (10), -- value required in cmd/req is not available
  hardwareFailure  (255) -- unspecified hardware failure
} (0..255)
```

```
CommandRef::=INTEGER(0..65535)
```

-- Values

```
version INTEGER(0..255)::=2 -- Version of this ASN.1 module
```

```
/*
The ASN.1 specification has been checked for conformance to the ASN.1
standards by OSS ASN.1 Syntax Checker, and by OSS ASN-1STEP
*/
```

END

Page 35, Annex B

Change Annex B from a normative annex to an informative annex.

Change all “shall be” statements in the whole annex to simple “is” statements.

Page 36, Table B.1

Delete the last row in the table.

Insert the following two new rows at the beginning of the table:

SimFUTcmd	Command to simulate an access to the ITS-S facilities layer via FA-SAP. Applicable only in test mode ASN.1: SimFUTcmd / -
SimFLTcmd	Command to simulate an access to the ITS-S facilities layer via NF-SAP. Applicable only in test mode ASN.1: SimFLTcmd / -

Page 37, B.2

Insert the following two new subclauses after B.2.1:

B.2.2 SimFUTcmd

MF-COMMAND “SimFUTcmd” is used by the ITS-S management to simulate an access to the ITS-S facilities layer via FA-SAP.

Table B.2 — MF-COMMAND.request for MF-COMMAND SimFUTcmd

ASN.1 Type	Description
MF-Command. simFUTcmd	FA-SAP service primitive received at the ITS-S facilities layer. Requires activation of test mode.

B.2.3 SimFLTcmd

MF-COMMAND “SimFLTcmd” is used by the ITS-S management to simulate an access to the ITS-S facilities layer via NF-SAP.

Table B.31 — MF-COMMAND.request for MF-COMMAND SimFLTcmd

ASN.1 Type	Description
MF-Command. simFLTcmd	NF-SAP service primitive received at the ITS-S facilities layer. Requires activation of test mode.