

Second edition
2013-03-15

AMENDMENT 1
2014-11-15

**Intelligent transport systems —
Communications access for land
mobiles (CALM) — Access technology
support**

AMENDMENT 1

*Systèmes intelligents de transport — Accès aux communications des
services mobiles terrestres (CALM) — Support à la technologie d'accès*

AMENDEMENT 1



Reference number
ISO 21218:2013/Amd.1:2014(E)

© ISO 2014

STANDARDSISO.COM : Click to view the full PDF of ISO 21218:2013/Amd 1:2014



COPYRIGHT PROTECTED DOCUMENT

© ISO 2014

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
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

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 WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/TC 204, *Intelligent transport systems*.

STANDARDSISO.COM : Click to view the full PDF of ISO 21218:2013/Amd 1:2014

Intelligent transport systems — Communications access for land mobiles (CALM) — Access technology support

AMENDMENT 1

Page 2, Terms and definitions

Add at the end:

3.7

temps atomique international

time since 00:00:00 UTC, 1 January, 2004, identical with UTC except that no leap seconds need to be added

Page 3, Abbreviated terms

Add:

TAI Temps Atomique International

UTC Temps Universel Coordonné/Coordinated Universal Time

Page 5, 6.2.1.1 CI classes

Add at the end of the first sentence:

A CI shall support exactly one of these CI classes presented in Table 1.

Page 6, Table 1

Delete all occurrences of the term “wireless”.

Delete the character “w” in the terms contained in column “Communication interface class”

Replace the character sequence “lan” in the terms contained in the column “Communication interface class” by the character sequence “il”.

Page 6, last sentence (before Table 2 on page 7)

Delete the end of the sentence “in line with requirements presented in Table 3”.

Page 7, 6.2.1.3

Delete the whole 6.2.1.3.

Page 11, bullet c) sub-bullet 2)

Add:

“not applicable for receive-only CIs (CIC-I4)”.

Page 12, bullet d)

Add:

“not applicable for receive-only CIs (CIC-I4)”.

Page 18, third paragraph below Figure 7

Replace the first three sentences with the corrected versions:

“Each RX/TX-CI shall maintain at least one RX-VCI for reception of frames. There is a single UC-VCI for every known peer CI. The RX-VCI of a CI is shared by all of its TX-VCIs.”

Page 21, Table 4

Replace “GeneralizedTime” by “Time48IAT”.

Page 29, Table 10

Insert:

TIMEOUT	2	Lifetime of pending packet expired prior to transmission of the packet
PRIORITY	3	Priority of the packet is smaller than the minimum required priority given in I-Parameter “MinimumUserPriority”

Page 32, Table A.1

Delete information on I-Param No 1. This parameter is no longer supported.

Page 33, Table A.1

Delete information on I-Param No 14 and 15. This parameter is no longer supported.

Page 37-46, Annex B

Replace whole annex by the following one:

Annex B (normative)

ASN.1 definitions

B.1 Use of modules

The ASN.1 modules specified in Annex B.2 shall be used. ASN.1 BASIC-PER, UNALIGNED, as specified in ISO/IEC 8825-2, shall apply.

In order to achieve octet alignment enabling cheap implementations, “fill” bits were defined. All fill bits shall be set to the value ‘0’.

B.2 ASN.1 modules

```
CALM11sap {iso(1) standard(0) calm-11-sap(21218) asnm-1 (1)} DEFINITIONS AUTOMATIC
TAGS ::= BEGIN
```

```
-- EXPORTS;
```

```
IMPORTS
```

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

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

```
MediumCost FROM CITSapplReq {iso(1) standard(0) cits-applReq (17423) asnm-1 (1)}
```

```
LogicalChannelType FROM CITSapplMgmtComm {iso(1) standard(0) cits-applMgmt (17419) comm
```

```

(3)}
;

-- End of IMPORTS

Alt::=INTEGER(0..65535) -- Resolution 0,1 m.

CIaClass::=INTEGER{
    unknown          (0),
    ciac-1           (1),
    ciac-2           (2),
    ciac-3           (3)
} (0..255)

Ciclass::=INTEGER{
    unknown          (0),
    cic-l1           (1),
    cic-l2           (2),
    cic-l3           (3),
    cic-l4           (4),
    cic-l5           (5),
    cic-il1          (254),
    cic-il2          (255)
} (0..255)

Link-ID::=SEQUENCE{
    remoteCIID       EUI64,
    localCIID        EUI64
}
-- Link-ID
-- CI in peer ITS-S(s)
-- Unique ID of local CI

CIstatus::=INTEGER{
    not-existent     (0),
    existent         (1),
    unknown          (2),
    registered       (4),
    active           (8),
    connected        (16),
    suspended        (64),
    inactive         (128)
} (0..255)

CommProfile::=SEQUENCE (SIZE(0..255)) OF I-Param

Connect::=INTEGER{
    automatic        (0),
    manual           (255)
} (0..255)

IN-SAPaddress::=INTEGER(0..255)

DataRate::= INTEGER(0..4294967295)
-- in 100 bit/s

DataRatesNW::=SEQUENCE{
    minimum          DataRate,
    maximum          DataRate
}
-- available to the appl.
-- minimum possible value
-- maximum possible value

Directivity::=SEQUENCE{
    mode             DirMode,
    dirPredef        INTEGER(0..255),
    fill             BIT STRING (SIZE(7)),
    dirVar           SEQUENCE (SIZE(0..1)) OF DirVar
}
-- 0: see dirVar
-- >0: predefined direction

DirMode::=INTEGER{
    fixed            (0),
    tracking          (255)
} (0..255)

```

```

DirVar::=SEQUENCE{
    bsAzimuth      INTEGER(-32768..32767), -- -180 - +180
    bsElevation    INTEGER(-128..127),      -- -90 - +90
    openHorizontal  INTEGER(0..255),         -- 0 - 180
    openVertical    INTEGER(0..255)         -- 0 - 180
}

TxPowMax::=INTEGER(0..255) -- unit, minimum value and step size defined by access
technology (medium)

Errors::=SEQUENCE (SIZE(0..255)) OF SingleError

SingleError::=SEQUENCE{
    paramNo      INTEGER(0..255), -- ref number of parameter
    errStatus    ErrStatus        -- IS24102-3
}

Gs::=INTEGER(0..65535) -- Resolution 0,01 m/s.

KineVectIn::=SEQUENCE{
    dut          Time48IAT, -- date and universal time
    lat          Lat,       -- latitude
    lon          Lon,       -- longitude
    alt          Alt,       -- altitude / elevation
    gs           Gs,        -- ground speed
    tta          Tta -- true track angle
}

KineVectOut::=SEQUENCE{
    dut          Time48IAT, -- date and universal time
    fill         BIT STRING (SIZE(5)), -- used for octet alignment in PER
    options      SEQUENCE (SIZE(0..5)) OF KineVectOptions -- options
}

KineVectOptions::=CHOICE{
    Lat          Lat, -- latitude
    lon          Lon, -- longitude
    alt          Alt, -- altitude
    gs           Gs,  -- ground speed
    tta          Tta -- true track angle
}

Lat::=INTEGER(-2147483648..2147483647) -- equals  $\pm \pi/2$ 

Lon::=INTEGER(-2147483648..2147483647) -- equals  $\pm \pi$ 

LLServiceAddr::=SEQUENCE{
    csap         IN-SAPaddress,
    linkID       Link-ID
}

MACAddress::=OCTET STRING (SIZE(6))

MediumUsage::=SEQUENCE{
    receive      INTEGER(0..255),
    transmit     INTEGER(0..255)
}

MedType::=INTEGER{
    unknown      (0),
    any          (1),
    iso21212     (2), -- 2G
    iso21213     (3), -- 3G
    iso21214     (4), -- IR
    iso21215     (5), -- M5
    iso21216     (6), -- MM
    iso25112     (7), -- 802.16e
    iso25113     (8), -- HC-SDMA
    iso29283     (9), -- 802.20

```



```

iso17515          (10), -- LTE
iso15628          (128), -- DSRC
can               (254),
ethernet          (255)
} (0..255)

MedUseObsTime::=SEQUENCE {
    value          INTEGER(0..1023), -- valid parameter number (10 bits in PER)
    unit           TimeUnit
} -- 2 octets in PER

TimeUnit::=INTEGER{
    microseconds   (0),
    milliseconds   (1),
    seconds         (2),
    minutes         (3),
    hours           (4),
    days            (5),
    weeks           (6),
    months          (7),
    years           (8)
} (0..63) -- 6 bits in PER

Notify::=SEQUENCE (SIZE(0..255)) OF INTEGER(0..255) -- valid parameter number

PeerRXpower::=INTEGER(0..255)

PARAMS::= CLASS {
    &paramRef INTEGER(0..255),
    &Parameter
}

IPARAM::= PARAMS

I-Params IPARAM::={errors | commProfile | properties | manuDeviceID | iTs-scuId | medID |
localCIID | timeoutReg | macAddress | macAddrTemp | ciClass | ciaClass | ciStatus | notify
| medType | regInfo | connect | simPin | providerInfo | minUserPriority | queueLevel |
queueLowTh | queueAlarmTh | distancePeer | commRangeRef | timeOfLastRecep | inactTimeLimit
| mediumUsage | medUseObsTime | peerMAC | virtualCIs | minPrioCrossCI | rxSens | txPower |
txPowMax | peerRXpower | dataRate | dataRateNW | dataRatesNW | dataRateNWreq | directivity
| blockLength | freeAirTime | frameLengthMax | kineVectIn | kineVectOut | cost |
reliability | logicalChannel, ...}

I-Param::=SEQUENCE{
    paramNo        IPARAM.&paramRef({I-Params}),
    parameter      IPARAM.&Parameter({I-Params}{@paramNo})
}

errors            IPARAM::={&paramRef 0, &Parameter Errors}
commProfile        IPARAM::={&paramRef 2, &Parameter CommProfile}
properties         IPARAM::={&paramRef 3, &Parameter Properties}
manuDeviceID       IPARAM::={&paramRef 4, &Parameter UTF8String}
iTs-scuId          IPARAM::={&paramRef 5, &Parameter ITS-scuId}
medID              IPARAM::={&paramRef 6, &Parameter MedID}
localCIID          IPARAM::={&paramRef 7, &Parameter EUI64}
timeoutReg         IPARAM::={&paramRef 8, &Parameter INTEGER(0..255)}
macAddress         IPARAM::={&paramRef 9, &Parameter MACaddress}
macAddrTemp        IPARAM::={&paramRef 10, &Parameter MACaddress}
ciClass            IPARAM::={&paramRef 11, &Parameter Ciclass}
ciaClass           IPARAM::={&paramRef 12, &Parameter CIaClass}
ciStatus           IPARAM::={&paramRef 13, &Parameter Cistatus}
notify             IPARAM::={&paramRef 16, &Parameter Notify}
medType            IPARAM::={&paramRef 17, &Parameter MedType}
regInfo            IPARAM::={&paramRef 18, &Parameter RegInfo}
connect            IPARAM::={&paramRef 19, &Parameter Connect}
simPin             IPARAM::={&paramRef 20, &Parameter SimPin}
providerInfo       IPARAM::={&paramRef 21, &Parameter ProviderInfo}
minUserPriority     IPARAM::={&paramRef 22, &Parameter UserPriority}
queueLevel         IPARAM::={&paramRef 23, &Parameter QueueLevel}

```

-- DEFAULT 0

```

queueLowTh      IPARAM::={&paramRef 24, &Parameter QueueValue}
queueAlarmTh    IPARAM::={&paramRef 25, &Parameter QueueValue}
distancePeer    IPARAM::={&paramRef 26, &Parameter Distance}
commRangeRef    IPARAM::={&paramRef 27, &Parameter Distance}
timeOfLastRecep IPARAM::={&paramRef 28, &Parameter Time48IAT}
  -- 1 ms resolution
inactTimeLimit  IPARAM::={&paramRef 29, &Parameter INTEGER(0..65535)}
  -- milliseconds
mediumUsage     IPARAM::={&paramRef 30, &Parameter MediumUsage}
medUseObsTime   IPARAM::={&paramRef 31, &Parameter MedUseObsTime}
peerMAC         IPARAM::={&paramRef 32, &Parameter MACaddress}
virtualCIs      IPARAM::={&paramRef 33, &Parameter VirtualCIs}
minPrioCrossCI  IPARAM::={&paramRef 34, &Parameter UserPriority}
rxSens          IPARAM::={&paramRef 35, &Parameter INTEGER(0..255)}
txPower         IPARAM::={&paramRef 36, &Parameter INTEGER(0..255)}
txPowMax        IPARAM::={&paramRef 37, &Parameter TxPowMax}
peerRXpower     IPARAM::={&paramRef 38, &Parameter PeerRXpower}
dataRate        IPARAM::={&paramRef 39, &Parameter DataRate}
dataRateNW      IPARAM::={&paramRef 40, &Parameter DataRate}
dataRatesNW     IPARAM::={&paramRef 41, &Parameter DataRatesNW}
dataRateNWreq   IPARAM::={&paramRef 42, &Parameter DataRate}
directivity     IPARAM::={&paramRef 43, &Parameter Directivity}
blockLength     IPARAM::={&paramRef 44, &Parameter INTEGER(0..65535)}
freeAirTime     IPARAM::={&paramRef 45, &Parameter INTEGER(0..255)}
  -- measured in milliseconds
frameLengthMax  IPARAM::={&paramRef 46, &Parameter INTEGER(0..255)}
  -- measured in milliseconds
kineVectIn      IPARAM::={&paramRef 47, &Parameter KineVectIn}
kineVectOut     IPARAM::={&paramRef 48, &Parameter KineVectOut}
cost            IPARAM::={&paramRef 49, &Parameter MediumCost}
reliability     IPARAM::={&paramRef 50, &Parameter INTEGER(0..255)}
logicalChannel  IPARAM::={&paramRef 51, &Parameter LogicalChannels}

Time48IAT::= INTEGER { utcStartOf2004(0), oneMilliSecAfterUTCStartOf2004(1) }
(0..2814747976710655) -- International Atomic Time with one millisecond steps

I-ParamNo::=INTEGER{
  errors          (0),
  commProfile     (1),
  properties      (2),
  manuDeviceID    (3),
  iTS-scuId       (4),
  medID           (5),
  localCIID       (6),
  timeoutReg      (7),
  macAddress      (8),
  macAddrTemp     (9),
  ciClass         (10),
  ciaClass        (11),
  ciStatus        (12),
  notify          (13),
  medType         (14),
  regInfo         (15),
  connect         (16),
  simPin          (17),
  providerInfo    (18),
  minUserPriority (19),
  queueLevel      (20),
  queueLowTh      (21),
  queueAlarmTh    (22),
  distancePeer    (23),
  commRangeRef    (24),
  timeOfLastRecep (25),
  inactTimeLimit  (26),
  mediumUsage     (27),
  medUseObsTime   (28),
  peerMAC         (29),
  virtualCI       (30),
  minPrioCrossCI  (31),
  rxSens          (32),
  txPower         (33),

```

```

    txPowMax          (34),
    peerRXpower       (35),
    dataRate          (36),
    dataRateNW        (37),
    dataRatesNW       (38),
    dataRateNWreq     (39),
    directivity       (40),
    blockLength       (41),
    freeAirTime       (42),
    frameLengthMax    (43),
    kineVectIn        (44),
    kineVectOut       (45),
    cost              (46),
    reliability        (47),
    logicalChannel    (51)
} (0..255)

SimPin::=OCTET STRING

LogicalChannels::=SEQUENCE (SIZE(0..255)) OF LogicalChannelType

Distance::=INTEGER(0..65535)      -- measured in 1/10 m

MedID::=INTEGER(0..255)

Properties::=SEQUENCE (SIZE(0..255)) OF I-Param -- only selected parameters

ProviderInfo::=SEQUENCE{
    provName          OCTET STRING (SIZE(0..255)), -- Name of provider
    apn               OCTET STRING (SIZE(0..255)), -- Name of access point
    username          OCTET STRING (SIZE(0..255)), -- Log-in name of user
    password          OCTET STRING (SIZE(0..255)) -- Password for log-in
}

QueueLevel::=SEQUENCE{
    priority          UserPriority,
    level             QueueValue
}

QueueValue::=INTEGER(0..255)

RegInfo::=SEQUENCE{
    status            RegInfoStatus,
    limits            RegulatoryScheme
}

RegInfoStatus::=INTEGER{
    notApplicable    (0),
    invalid          (1),
    new              (254),
    valid            (255)
} (0..255)

REGULSCHEME::=CLASS {
    &regID            INTEGER(0..255),
    &RegInfo
}

RegulatoryScheme::=SEQUENCE{
    reguIdent        REGULSCHEME.&regID({RegulSchemes}),
    reguInfo         REGULSCHEME.&RegInfo({RegulSchemes}{@reguIdent})
}

RegulSchemes REGULSCHEME::={nullRegScheme, ...} -- to be filled

nullRegScheme REGULSCHEME::={&regID 0, &RegInfo NULL}

```

Tta::=INTEGER(0..65535) -- Resolution 0,1°.

UserPriority::=INTEGER(0..255)

VirtualCIs::=SEQUENCE (SIZE(0..65535))OF RemoteCIID

INSAP::= COMMUPDOWN

INsapspdown INSAP::={inUnitdataRq | inDataAckRq | inReplyAckRq | inReplyUpdateRq, ...}

INsapPrimitivesDown::=SEQUENCE{
 spRef INSAP.&primitiveRef({INSapspdown}),
 servPrimitive INSAP.&Primitive({INSapspdown}){@spRef})
 }

inUnitdataRq INSAP::={&primitiveRef 0, &Primitive IN-UNITDATA-request}
 inDataAckRq INSAP::={&primitiveRef 1, &Primitive IN-DATA-ACK-request}
 inReplyAckRq INSAP::={&primitiveRef 2, &Primitive IN-REPLY-ACK-request}
 inReplyUpdateRq INSAP::={&primitiveRef 3, &Primitive IN-REPLY-UPDATE-request}

IN-UNITDATA-request::=SEQUENCE{
 source-addr LLserviceAddr,
 dest-addr LLserviceAddr,
 data INdata,
 priority UserPriority,
 accessParams AccessParameters
 }

IN-DATA-ACK-request::=SEQUENCE{
 source-addr LLserviceAddr,
 dest-addr LLserviceAddr,
 data INdata,
 priority UserPriority,
 serviceClass MACServiceClass
 }

IN-REPLY-ACK-request::=SEQUENCE{
 source-addr LLserviceAddr,
 dest-addr LLserviceAddr,
 data INdata,
 priority UserPriority,
 serviceClass MACServiceClass
 }

IN-REPLY-UPDATE-request::=SEQUENCE{
 source-addr LLserviceAddr,
 data INdata
 }

INSAPUP::= COMMUPDOWN

INsapspsup INSAP::={inUnitdataInd | inUnitdataStatusInd | inDataAckInd | inDataAckStatusInd | inReplyInd | inReplyStatusInd | inReplyUpdateStatusInd, ...}

INsapPrimitivesUp::=SEQUENCE{
 spRef INSAP.&primitiveRef ({INSapspsup}),
 servPrimitive INSAP.&Primitive ({INSapspsup}){@spRef})
 }

inUnitdataInd INSAP::={&primitiveRef 0, &Primitive IN-UNITDATA-indication}
 inUnitdataStatusInd INSAP::={&primitiveRef 1, &Primitive IN-UNITDATA-STATUS-indication}
 inDataAckInd INSAP::={&primitiveRef 2, &Primitive IN-DATA-ACK-indication}
 inDataAckStatusInd INSAP::={&primitiveRef 3, &Primitive IN-DATA-ACK-STATUS-indication}
 inReplyInd INSAP::={&primitiveRef 4, &Primitive IN-REPLY-indication}
 inReplyStatusInd INSAP::={&primitiveRef 5, &Primitive IN-REPLY-STATUS-indication}
 inReplyUpdateStatusInd INSAP::={&primitiveRef 6, &Primitive IN-REPLY-UPDATE-STATUS-indication}

IN-UNITDATA-indication::=SEQUENCE{