INTERNATIONAL STANDARD

ISO 24102-2

First edition
2015-12-01

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

Part 2: Remote management of ITS-SCUs

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

Partie 2: Gestion à distance des SCUs-ITS

Reference number
ISO 24102-2:2015(E)

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

Contents

Page

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*.

This first edition, together with ISO 24102-1, ISO 24102-3, ISO 24102-4, ISO 24102-5 and ISO 24102-6, cancels and replaces ISO 24102:2010, which has been technically revised.

ISO 24102 consists of the following parts, under the general title *Intelligent transport systems — Communications access for land mobiles (CALM) — ITS station management*:

— *Part 1: Local management*

— *Part 2: Remote management of ITS-SCUs*

— *Part 3: Service access points*

— *Part 4: Station-internal management communications*

— *Part 5: Fast service advertisement protocol (FSAP)*

— *Part 6: Path and flow management*

Introduction

This part of ISO 24102 is part of a family of International Standards for communications access for land mobiles (CALM). An introduction to the whole set of International Standards for Intelligent Transport Systems (ITS) is provided in ISO 21217.

This part of ISO 24102 is the second part of a multipart International Standard which determines remote management of an ITS station unit (ITS-SU) with the ITS station and communication architecture specified in ISO 21217 and illustrated in Figure 1, and operated as a bounded secured managed domain (BSME).

Remote ITS station management has the purpose of

— setting, updating, and deletion of configuration and operation information in an ITS station communication units (ITS-SCU) of an ITS station unit (ITS-SU) specified in ISO 21217, e.g. information on policies and regulations, security related information, accounting information, access layer parameters (see Reference [1]),

— installation, update, and deinstallation of persistent information in an ITS-SCU, e.g. ITS-S application processes specified in ISO 21217, ITS-S communication protocols, and

— notification and retrieval of management information, e.g. log files of events, alarms generated by the ITS-SCU(s) of an ITS-SU.

By this, it covers the five management areas identified in ISO/IEC 7498-4.

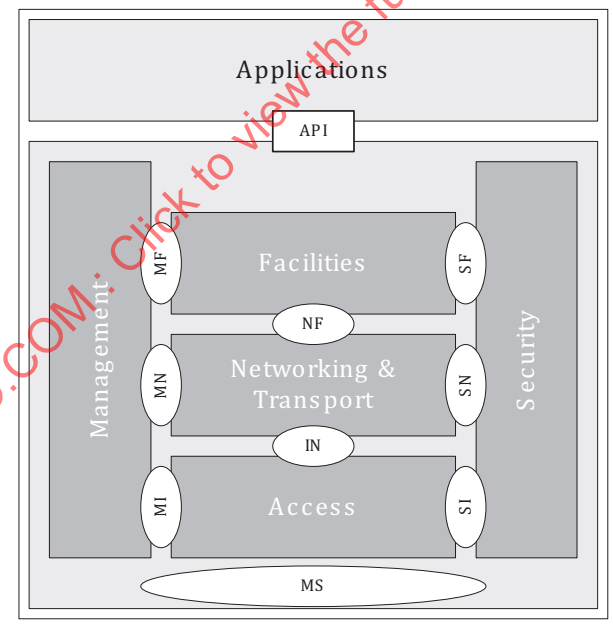

Figure 1 — ITS station reference architecture

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

Part 2: Remote management of ITS-SCUs

1 Scope

This part of ISO 24102 provides specifications for Intelligent Transport Systems (ITS) station management to be compliant with the ITS station reference architecture and the set of related standards from ISO/TC 204.

Remote ITS station management is specified by means of protocol data units (PDUs) and procedures of the "Remote ITS Station Management Protocol" (RSMP) related to managed objects in an ITS station unit. Distinction is made between managed ITS station units (management clients) and managing remote ITS station units (management servers).

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 21217, *Intelligent transport systems — Communications access for land mobiles (CALM) — Architecture*

ISO 24102-1, *Intelligent transport systems — Communications access for land mobiles (CALM) — ITS station management — Part 1: Local management*

ISO 24102-3, *Intelligent transport systems — Communications access for land mobiles (CALM) — ITS station management — Part 3: Service access points*

ISO 24102-4, *Intelligent transport systems — Communications access for land mobiles (CALM) — ITS station management — Part 4: Station-internal management communications*

ISO/IEC 7498-4, *Information processing systems — Open Systems Interconnection — Basic Reference Model — Part 4: Management framework*

ISO/IEC 8825-2, *Information technology — ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 21217, ISO 24102-1, ISO 24102-3, ISO 24102-4, ISO/IEC 7498-4, and the following apply.

3.1
remote management client
ITS station communication unit in which remote ITS station management is performed by a *remote management server* (3.2)

3.2
remote management server
entity performing remote ITS station management in an ITS station communication unit

4 Symbols and abbreviated terms

For the purposes of this document, the abbreviated terms given in ISO 21217, ISO 24102-1, ISO 24102-3, ISO/IEC 7498-4, and the following apply.

BSME	Bounded Secured Managed Entity (from ISO 21217)
IICP	ITS station-Internal management Communications Protocol (from ISO 24102-4)
FSAP	Fast Service Advertisement Protocol (from ISO 24102-5)
ITS	Intelligent Transport Systems
ITS-SCU	ITS Station Communication Unit (from ISO 21217)
ITS-SCU-CMC	ITS-SCU Configuration Management Centre (from CEN/ISO/TS 17419)
ITS-SU	ITS Station Unit (from ISO 21217)
RMC	Remote Management Client
RMCH	Remote Management Communication Handler
RMPE	Remote Management Protocol Execution
RMS	Remote Management Server
RSMP	Remote ITS station Management Protocol

5 Requirements

The ITS station management entity shall provide the functionality specified in the various parts of this multipart International Standard.

1) The functionality of local ITS station management specified in ISO 24102-1.
2) The functionality of remote ITS station management specified in this part of ISO 24102.
3) The functionality of management service access points specified in ISO 24102-3.
4) The functionality of ITS station-internal management communications specified in ISO 24102-4.
5) The functionality of the “Fast Service Advertisement Protocol” (FSAP) specified in ISO 24102-5.
6) The functionality of the path and flow management specified in ISO 24102-6.

Means to secure the access to management functionality need to be specified within the global context of ITS security. Details are outside the scope of this part of ISO 24102.

Detailed mandatory requirements are specified in the following clauses of this part of ISO 24102.

— Clause 6 presents the remote management architecture.
— Clause 7 specifies remote management protocol data units.
— Clause 8 specifies service primitive functions.
— Clause 9 specifies remote management procedures.
— Clause 10 specifies details needed for the Fast Service Advertisement Protocol (FSAP).
— Annex A specifies the ASN.1 module for remote management.

— Annex B proposes settings of communication service parameters used for automatic selection of communication profiles specified in CEN/ISO/TS 17423.

6 Remote management architecture

6.1 Functionality

The "Remote ITS Station Management Protocol" (RSMP) specified in this part of ISO 24102 has the purpose of

— setting, updating, and deletion of configuration and operation information in an ITS station communication unit (ITS-SCU) of an ITS station unit (ITS-SU) specified in ISO 21217, e.g. information on policies and regulations (CEN/ISO/TS 17419), security related information, accounting information, access layer parameters (see Reference [1]), etc.,

— installation, update, and deinstallation of persistent information in an ITS-SCU, e.g. ITS-S application processes, ITS-S communication protocols, and

— notification and retrieval of management information, e.g. log files of events, alarms generated by the ITS-SCU of an ITS-SU.

By this, it covers the five management areas identified in ISO/IEC 7498-4.

Remote ITS station management covers a set of management processes where ITS station units (ITS-SU) acting as remote management servers (RMS) manage ITS station communication units (ITS-SCU) of managed ITS-SUs acting as remote management clients (RMC).

An RMS is associated with an ITS-SCU configuration management centre identified in CEN/ISO/TS 17419. An RMS may be implemented, e.g. in a roadside ITS sub-system or in a central ITS sub-system.

Remote ITS station management is applied to managed objects (according to ISO/IEC 7498-4) in remote management sessions. Such sessions may be initiated

— by the RMS (server initiated session), e.g. by means of the Fast Service Advertisement Protocol (FSAP) (see Reference [2]) or by direct IPv6 based access, or

— by the RMC (client initiated session), typically using IPv6 communications, as illustrated in Figure 2 (server initiated session using FSAP), in Figure 3 (direct server initiated session), and in Figure 4 (client initiated session).

The mechanisms specified in this part of ISO 24102 enable future specifications of remote management features in separate standards or by means of registries.

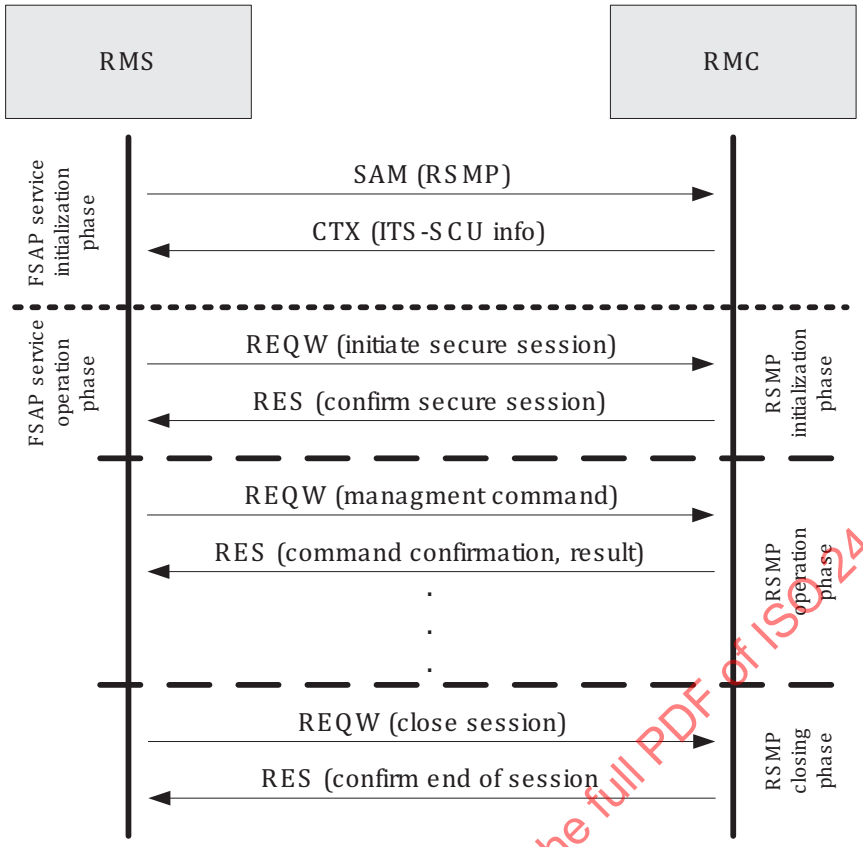

Figure 2 — Server initiated session (example with FSAP)

SAM and CTX specified in Reference [2] with details specified in this part of ISO 24102 are used in the example of Figure 2 to prepare for the secured management session. During the FSAP service operation phase, first, a secure session is requested from the RMS which is acknowledged by the RMC. After successful establishment of a session with mutual authentication of RMS and RMC with optional agreement on encryption of the management data to be exchanged in the session, the RMS may send out a sequence of management commands, each of which is acknowledged by the RMC providing also optional result data. Finally, the RMS closes the session, which also is acknowledged by the RMC. Subsequent to this, no more management data can be exchanged.

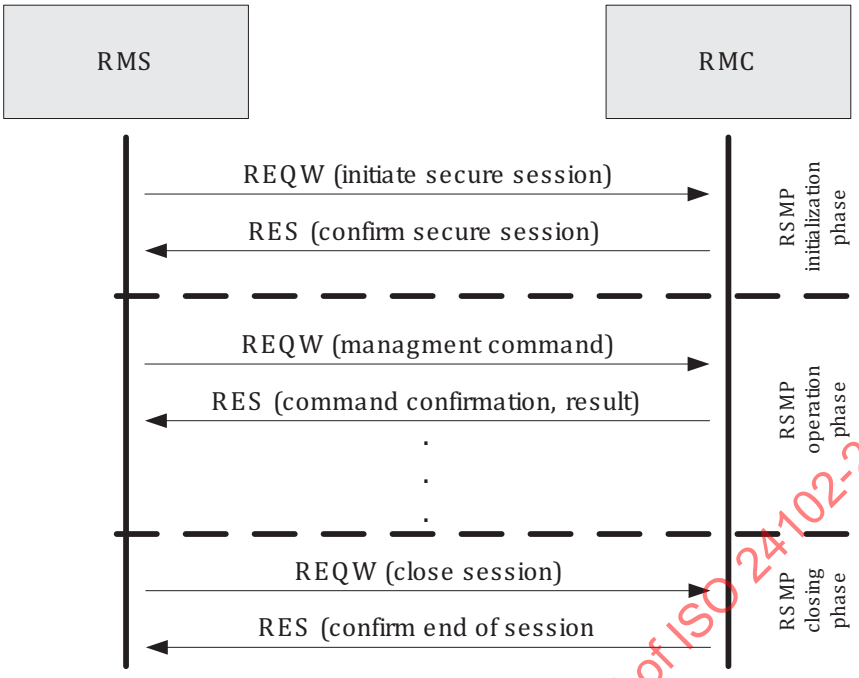

Figure 3 — Direct server initiated session

In the example of Figure 3, an RMS directly initiates a secure session with an RMC. After confirmation of the secure session by the RMC, the RMS runs and closes the secure session as illustrated above for the direct server initiated session.

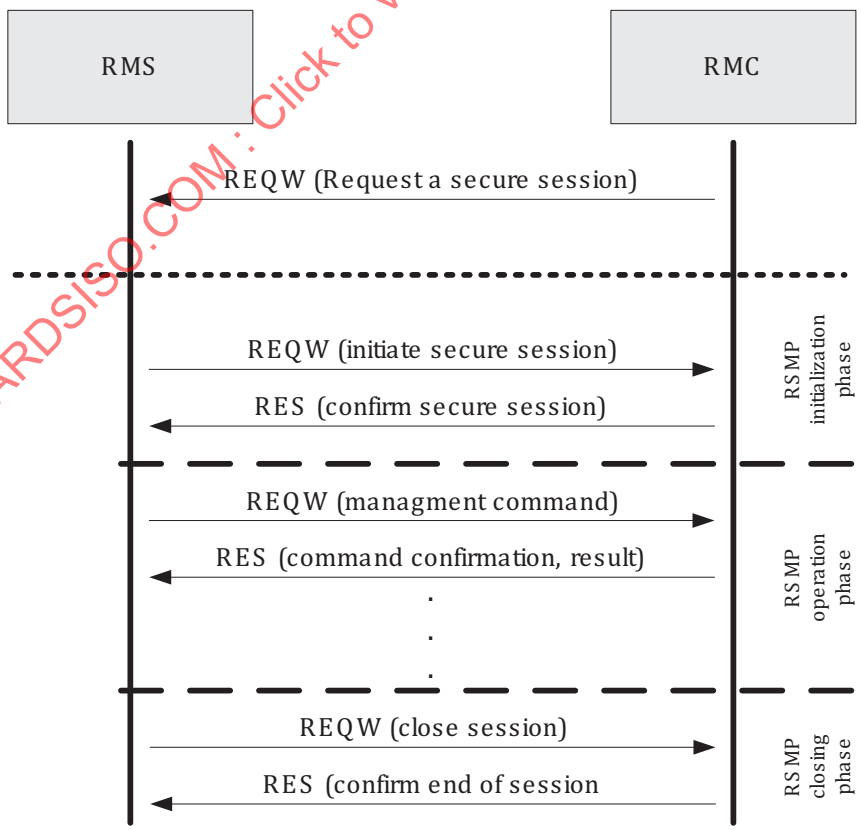

Figure 4 — Client initiated session

In the example of Figure 4, upon an event internal to an RMC, an RMC notifies the need for a secure session to the RMS. Then the RMS initiates, runs, and closes the secure session.

6.2 ITS station architecture

The "Remote ITS-station Management Protocol" (RSMP) consists of two functional blocks, i.e.

— the ITS-S application process "Remote Management Protocol Execution" (RMPE) with a registered ITS-AID, and

— the ITS-S facility "Remote Management Communication Handler" (RMCH) using a well-known registered ITS port number PORT_RSM and dynamically assigned ITS port numbers (see Reference [3]). The value of PORT_RSM is 32763.

The allocation of these functional blocks in the ITS station architecture specified in ISO 21217 is presented in Figure 5. Globally, unique identifiers are specified in CEN/ISO/TS 17419.

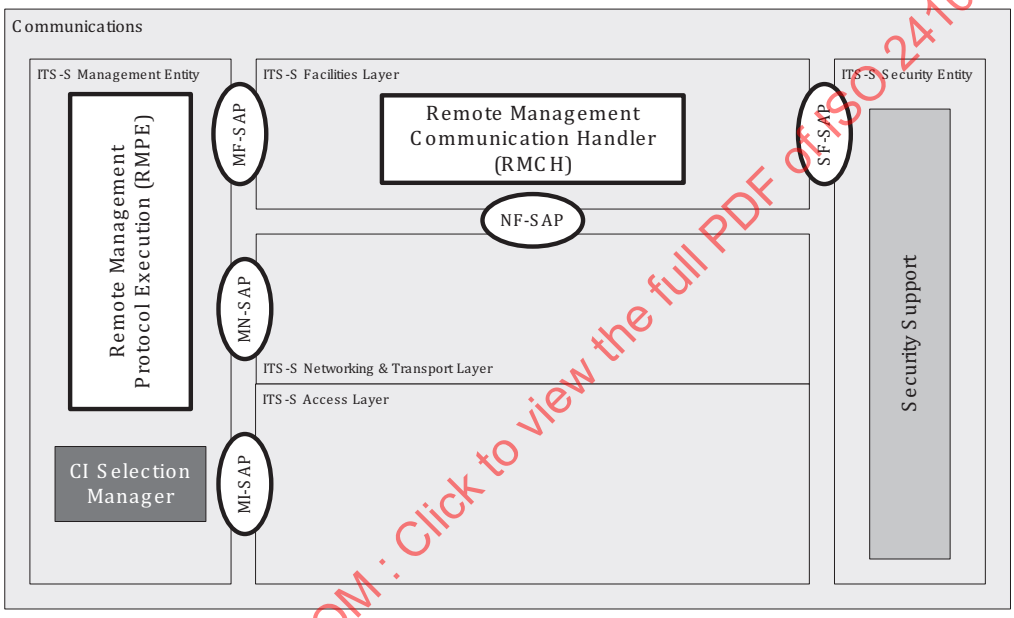

Figure 5 — Functional blocks of RSMP

The RMCH is located in the ITS-S facilities layer.

The RMPE is located in the ITS-S management entity.

RMCH and RMPE are connected via the MF-SAP services MF-COMMAND and MF-REQUEST with service primitive functions specified in Clause 8.

6.3 Distributed implementation of an ITS-S

The "Remote ITS-station Management Protocol" (RSMP) supports distributed implementations of ITS-S roles identified in ISO 21217, i.e. several ITS-SCUs per ITS-SU. The RMCH thus may communicate via the ITS station-internal network with an ITS-SCU providing the link to the peer ITS station unit. Details depend on the ITS-S networking and transport layer protocol used and are outside the scope of this part of ISO 24102.

6.4 RMPE

"Remote Management Protocol Execution" (RMPE) is an ITS-S application process located in the ITS-S management entity. There are two distinct instantiations of the RMPE, i.e. the server instantiation and

the client instantiation. There is exactly one instantiation of RMPE in each ITS-SCU of an ITS-SU. The RMPE cannot manage ITS-SCUs in which it is not instantiated.

NOTE For more information on ITS-S application processes, read ISO 21217.

Management activities include the following:

— updating firmware in the ITS-SCU;

— maintenance of ITS-S application processes;

— new installations;

— updates of existing installations;

— deletion of existing installations;

— maintenance of communication, management, and security protocols;

— new installations;

— updates of existing installations;

— deletion of existing installations;

— maintenance of configuration parameters;

— setting of parameter values and other information;

— retrieval of parameter values and other information, e.g. logfiles;

— maintenance of security related managed objects;

6.5 RMCH

The "Remote Management Communication Handler" (RMCH) is a communication facility located in the ITS-S facilities layer. The RMCH

— receives service data units which contain "RMCH Protocol Data Units" (RMCH-PDUs) illustrated in Figure 7 from peer ITS-SUs,

— exchanges RMPE-PDUs illustrated in Figure 7 with the RMPE via the MF-SAP,

— transmits RMCH-PDUs to peer ITS-SUs, and

— uses services from the ITS-S security entity via SF-SAP service primitives to authenticate peer ITS station units, and to optionally encrypt and decrypt RMPE-PDUs.

The well-known ITS port PORT_RSM (see Reference [3]) is used by the following:

a) an RMS for transmission of a message

   — as a source port number, and

   — as a destination port number in case of direct session initiation, and only in the REQW (initiate secure session) message shown in Figure 3. With the REQW (initiate secure session), an RMS requests an RMC to use a specific dynamically-assigned port number for subsequent communications.

b) an RMC for transmission of a message

   — as a destination port number in case of direct session initiation, and only in the "notify need for secure session" message shown in Figure 4.

Client instantiations use dynamically-assigned port numbers as source port number.

7 Remote management protocol data units

Remote station management uses the protocol data units, data elements, and security elements illustrated in Figure 6, Figure 7, Figure 8, and Figure 9. The ASN.1 presentation of these PDUs is specified in Annex A.

RMCH - Request/Response:

SecHeader	*OCTET STRING containing RSMP-Request or RSMP-Response* optionally encrypted	SecTrailer

Figure 6 — RMCH protocol data units

RMCH-Request and RMCH-Response messages encapsulate RSMP-Request and RSMP-Response messages, respectively, between a security header and a security trailer. RSMP-Request and RSMP-Response messages may be encrypted as indicated in the SecHeader fields.

RSMP-Request:

SessionID	PDU-Counter	PDU-ID (0)	Length of remainder	Request Data

RSMP-Response:

SessionID	PDU-Counter	PDU-ID (1)	Length of remainder	Response Data

Figure 7 — RSMP protocol data units RSMP-Request and RSMP-Response

RSMP-Request messages of ASN.1 type `RSMPmessage` are sent by RMSs and by RMCs to request a remote management session as illustrated in Figure 4. RSMP-Response messages of ASN.1 type `RSMPmessage` are sent only by RMCs.

The SessionID identifies uniquely a session for a specific ITS-SCU and a specific management centre. The value zero is used by a RMC in the RSMP-Request message requesting initiation of a secure session by an RMS.

The PDU-Counter distinguishes PDUs uniquely in a session. The value presented in an RSMP-Request is used in the corresponding RSMP-Response.

The PDU-ID distinguishes RSMP-Request and RSMP-Response messages.

Request Data:

RqDataID	Length of RqData	RqData

Response Data:

RsDataID	Length of RsData	RsData	Error Status

Figure 8 — Request and response data elements `RequestData` and `ResponseData`

RqDataIDs are given in the ASN.1 type `RefRSMPREQ` and RsDataIDs are given in the ASN.1 type `RefRSMPRES`. These IDs uniquely identify RqData and RsData, respectively.

The ErrorStatus is given by the ASN.1 type `RSMPErrStatus`.

SecHeader:

SecHeadID	Length of SecHead	SecHead

SecTrailer:

SecTrailID	Length of SecTrail	SecTrail

Figure 9 — Security header and trailer `SecHeader` and `SecTrailer`

SecHeadIDs and SecTrailIDs are given in the ASN.1 type `RefSECRSMP`. Not applying any security is given by the ASN.1 value `c-noSecurity`; the corresponding SecHead and SecTrail are of ASN.1 type `NullType`.

8 Service primitive functions

8.1 Generic service primitives

Service primitives MF-COMMAND.request, MF-COMMAND.confirm, MF-REQUEST.request, MF-REQUEST.confirm, SF-REQUEST.request, and SF-REQUEST.confirm are specified in ISO 24102-3. This part of ISO 24102 specifies functions for these generic service primitives applicable for RSMP.

8.2 MF-SAP service primitive functions

8.2.1 Transmission request of RSMP-Request and RSMP-Response

The service primitive MF-COMMAND.request is used. ASN.1 types and values for the applicable function shall be as specified in Table 1, with ASN.1 details specified in Annex A.

Table 1 — Transmission request of RSMP-Request and RSMP-Response

MF-Command-request		
&mxref	**&MXParam**	**Description**
`c-rsmpMessageTX =13`	`RSMPmessageTX`	Request to transmit RSMP-Request and RSMP-Response to a specific peer station
MF-Command-confirm		
&mxref	**&MXParam**	**Description**
`c-rsmpMessageTXconf =13`	`NullType`	The MF-Command-request `RSMPmessageTX` shall be confirmed with `NullType` and with ErrStatus as specified in ISO 24102-3.

8.2.2 Notification of reception of RSMP-Request and RSMP-Response

The service primitive MF-REQUEST.request is used. ASN.1 types and values for the applicable functions shall be as specified in Table 2, with ASN.1 details specified in Annex A.

Table 2 — Notification of reception of RSMP-Request

MF-Request-request		
&mxref	**&MXParam**	**Description**
c-rsmpMessageRX =15	RSMPmessageRX	Notification of reception of RSMP-Request and RSMP-Response from specific peer station
MF-Request-confirm		
&mxref	**&MXParam**	**Description**
c-rsmpMessageRXconf =15	NullType	The request RSMPmessageRX shall be confirmed with NullType and with ErrStatus as specified in ISO 24102-3.

8.3 SF-SAP service primitive functions

8.3.1 Security procedure applied to RSMP-Request and RSMP-Response

The service primitive SF-REQUEST.request is used. ASN.1 types and values for the applicable functions shall be as specified in Table 3, with ASN.1 details specified in Annex A.

Table 3 — Security procedure applied to RSMP-Request

SF-Request-request		
&mxref	**&MXParam**	**Description**
c-secRSMPmessageTX =2	SecRSMPmessageTX	Request to secure RSMP-Request prior to transmission to a specific peer station
SF-Request-confirm		
&mxref	**&MXParam**	**Description**
c-secRSMPmessageTXConf =2	RMCHmessage	The SF-Request-request SecRSMPmessageTX shall be confirmed with the secured copy of RSMP-Request, i.e. the RMCH-Request and with ErrStatus as specified in ISO 24102-3.

8.3.2 Security procedure applied to RMCH-Request and RSMP-Response

The service primitive SF-Request.request is used. ASN.1 types and values for the applicable functions shall be as specified in Table 4, with ASN.1 details specified in Annex A.

Table 4 — Security procedure applied to RMCH-Request and RMCH response

SF-Request-request		
&mxref	**&MXParam**	**Description**
c-secRMCHrX =3	SecRMCHrX	Request to secure a received RMCH-Request from a specific peer station prior to forwarding to the RMPE
SF-Request-confirm		
&mxref	**&MXParam**	**Description**
c-secRMCHrXConf =3	RSMPmessage	The SF-Request-request SecRMCHrX shall be confirmed with the secured copy of RMCH-Request, i.e. the RSMP-Request and with ErrStatus as specified in ISO 24102-3

9 Remote management procedures

9.1 Remote management session initiation

9.1.1 Initiation by server

An RMS may initiate a remote management session upon purpose. In order to start a remote management session, the RMS shall send the RSMP-Request(`rmsInitSession`) to the RMC in the desired ITS-SCU.

Upon reception of the RSMP-Request(`rmsInitSession`), the RMC shall acknowledge the request with the RSMP-Response(`rmsInitSessionRs`).

9.1.2 Initiation by client

An RMC may initiate a remote management session upon purpose. In order to request initiation of a remote management session, the RMC shall send the RSMP-Request(`RMCreqSession`) to the desired management centre. This request may be repeated until the RMC receives an RSMP-Request(`rmsInitSession`) from the selected RMS starting the requested session.

Upon reception of the RSMP-Request(`rmsInitSession`), the RMC shall acknowledge the request with the RSMP-Response(`rmsInitSessionRs`).

9.1.3 RSMP session identifier

Every remote management session is identified by an "RSMP Session Identifer" SessionID of ASN.1 type `RsmpSessionID` assigned by the RMC. The values of SessionID shall be unique for a given RMC. The value zero shall be used by a RMC in the RSMP-Request message requesting initiation of a secure session by a RMS as specified in Clause 7.

9.1.4 RSMP session security

Details of security schemes to be applied for a remote management session may be negotiated between RMS and RMC. This negotiation is performed using the RSMP-Request and RSMP-Request PDUs specified in Clause 7. Details of these PDUs will be specified in a future version of this part of ISO 24102.

9.2 Remote management session closure

9.2.1 Active closure

As soon as no further remote management actions are needed in a remote management session, the RMS shall close the session. Closure of a remote management session shall be indicated by the RMS by sending the RSMP-Request(`rmsCloseSession`) to the RMC.

Upon reception of the RSMP-Request(`RMScloseSession`), the RMC shall acknowledge the request with the RSMP-Response(`rmsCloseSessionRs`), and shall treat the remote management session as closed.

9.2.2 Timeout

In case an RMC does not receive RSMP-Requests from an RMS during an open remote management session for a time period larger than the time indicated in the management parameter RSMPtimeout of ASN.1 type `RSMPtimeout`, the RMC shall close the remote management session on its own.

In order to keep a session alive, an RMS may periodically send the Ping-command RSMP-Request(`pingRq`) with arbitrary data which shall be acknowledged by the RMS with the Ping-command RSMP-Response(`pingRs`) returning the received arbitrary data.

9.2.3 No active session

As long as an RMC is not in an active remote management session with an RMS, it shall ignore all RSMP-Request messages except the one to start a remote management session and an optionally repeated RSMP-Request(`rmsCloseSession`).

9.3 Firmware update

A partial or complete update of the firmware of an ITS-SCU may be requested by the RMS by sending the RSMP-Request(`firmwareUpdate`) message. Upon reception of the RSMP-Request(`firmwareUpdate`) message, the RMC shall first acknowledge reception of the RSMP-Request(`firmwareUpdate`) message by sending the RSMP-Response(`firmwareUpdateRs`) message, and shall then perform the requested firmware update in line with requirements on secure operation of the platform in which the ITS-SCU is installed. Upon automatic restart of the ITS-SCU after the firmware update, the ITS-SCU shall request a remote management session at the RMS to confirm success of the operation, and to close correctly the session.

As long as a firmware update session is active, other management sessions are prohibited.

9.4 Maintenance of ITS-S protocols

ITS-S protocols are protocols different to ITS-S application processes which reside in the ITS-S access layer, the ITS-S networking and transport layer, the ITS-S facilities layer, the ITS-S management entity, and the ITS-S security entity.

Maintenance of an ITS-S protocols may be performed with RSMP-Request(`ProtMaintenance`) distinguishing three types of maintenance operation.

a) Installation of a new ITS-S protocol requested with ASN.1 type `ProtMgmtInstallRq`.

b) Update of an existing ITS-S protocol requested with ASN.1 type `ProtMgmtUpdateRq`. The existing ITS-S protocol is identified with ASN.1 type `ProtocolID` provided by the RSC in the acknowledgement of an initial installation.

c) Deletion of an existing ITS-S protocol requested with ASN.1 type `ProtMgmtDeleteRq`. The existing ITS-S protocol is identified with ASN.1 type `ProtocolID` provided by the RSC in the acknowledgement of an initial installation.

A maintenance request shall be acknowledged by an RMC with RSMP-Response(`protMaintenanceRs`) distinguishing three types of maintenance operation.

a) Acknowledgement of an installation of a new (instantiation of an) ITS-S application process with ASN.1 type `ProtMgmtInstallRs` providing the unique application Identifier of ASN.1 type `ProtocolID` of this instantiation of an ITS-S application process. In case of failure, `ProtocolID.ProtInstance` shall contain the value zero, otherwise the value contained in `ProtMgmtInstallRq`.

b) Acknowledgement of an update of an existing ITS-S protocol with ASN.1 type `ProtMgmtUpdateRs`. In case of failure, `ProtocolID.ProtInstance` contained in `ProtMgmtUpdateRs` shall contain the value zero, otherwise the value contained in `ProtMgmtUpdateRq`.

c) Acknowledgement of deletion of an existing ITS-S protocol with ASN.1 type `ProtMgmtDeleteRs`. In case of success, `ProtocolID.ProtInstance` contained in `ProtMgmtDeleteRs` shall contain the value zero, otherwise the value contained in `ProtMgmtDeleteRq`.

The return status of ASN.1 type `RSMPErrStatus` contained in the acknowledgement shall present the values as specified in Table 5.

Table 5 — Protocol maintenance result status

Status value in RSMPErrStatus	Installation	Update	Deletion
rsmpErrSuccess	Successful installation	Successful update	Successful deletion
rsmpErrRejected	Request rejected by RSC for unknown reasons		
rsmpErrAppUnknown	n.a.	Request rejected as referenced ITS-S protocol given in ASN.1 type ProtocolID is not known at the RSC.	
rsmpErrUnspecFailure	Request failed for unknown reasons		

9.5 Maintenance of ITS-S application processes

Maintenance of an ITS-S application process may be performed with RSMP-Request(appMaintenance) distinguishing three types of maintenance operation.

a) Installation of a new (instantiation of an) ITS-S application process requested with ASN.1 type AppMgmtInstallRq.

b) Update of an existing ITS-S application process requested with ASN.1 type AppMgmtUpdateRq. The existing ITS-S application process is identified with ASN.1 type ApplicationID provided by the RSC in the acknowledgement of an initial installation.

c) Deletion of an existing ITS-S application process requested with ASN.1 type AppMgmtDeleteRq. The existing ITS-S application process is identified with ASN.1 type ApplicationID provided by the RSC in the acknowledgement of an initial installation.

A maintenance request shall be acknowledged by an RMC with RSMP-Response(appMaintenanceRs) distinguishing three types of maintenance operation.

a) Acknowledgement of an installation of a new (instantiation of an) ITS-S application process with ASN.1 type AppMgmtInstallRs providing the unique application Identifier of ASN.1 type ApplicationID of this instantiation of an ITS-S application process. In case of failure, ApplicationID.AppInstance shall contain the value zero, otherwise the value contained in AppMgmtInstallRq.

b) Acknowledgement of an update of an existing ITS-S application process with ASN.1 type AppMgmtUpdateRs. In case of failure, ApplicationID.AppInstance contained in AppMgmtUpdateRs shall contain the value zero, otherwise the value contained in AppMgmtUpdateRq.

c) Acknowledgement of deletion of an existing ITS-S application process with ASN.1 type AppMgmtDeleteRs. In case of success, ApplicationID.AppInstance contained in AppMgmtDeleteRs shall contain the value zero, otherwise the value contained in AppMgmtDeleteRq.

The return status of ASN.1 type RSMPErrStatus contained in the acknowledgement shall present the values as specified in Table 6.

Table 6 — Application maintenance result status

Status value in RSMPErrStatus	Installation	Update	Deletion
rsmpErrSuccess	Successful installation	Successful update	Successful deletion
rsmpErrRejected	Request rejected by RSC for unknown reasons		
rsmpErrAppUnknown	n.a.	Request rejected as referenced instantiation of an ITS-S application process given in ASN.1 type ApplicationID is not known at the RSC.	
rsmpErrUnspecFailure	Request failed for unknown reasons		

9.6 Maintenance of configuration information

In order to read the value of one or several management parameters (M-Params specified in ISO 24102-1), an RMS shall send the RSMP-Request(getMparams) message. Upon reception of the RSMP-Request(getMparams) message, the RMC shall return the requested information in the RSMP-Response(getMparamsRs) message. In case a parameter value cannot be provided, it shall be omitted in RSMP-Response(getMparamsRs).

In order to write the value of one or several management parameters (M-Params specified in ISO 24102-1), an RMS shall send the RSMP-Request(setMparams) message. Upon reception of the RSMP-Request(setMparams), the RMC shall perform the requested parameter settings in case no access violation occurred and shall report success or failure in the RSMP-Response(setMparamsRs). In case all requested parameter settings could be performed, the global result status shall be set to "rsmpErrSuccess". Otherwise, the global result status shall be set to "rsmpErrSetErrorGeneral" and detailed result status shall be returned for every failure that occurred.

10 Usage of FSAP

10.1 General

The "Fast Service Advertisement Protocol" FSAP is specified in Reference [2]. This part of ISO 24102 specifies partly usage of FSAP for RSMP. Other parts may be subject to private specifications.

10.2 SAM

The ITS-AID presented in a "Service Advertisement Message" (SAM) shall be the registered value 134 of RMPE.

The serverPort field in a SAM shall show the registered well-known number 32763 of PORT_RSM (see Reference [3]).

The serviceData field in a SAM may contain the unique identifier of ASN.1 type ItsScuCmcID of the ITS-SCU-CMC specified in CEN/ISO/TS 17419 that offers remote station management. In case different ITS-SCU-CMCs are available, the serviceData field shall be empty.

10.3 CTX

The ITS-AID presented in a "Service Context Message" (CTX) shall be the registered value 134 of RMPE.

The clientPort field shall contain the dynamically assigned port number of the RMC as specified in References [2] and [3].

The contextData field shall contain the unique identifier of ASN.1 type ItsScuCmcID of the appropriate ITS-SCU-CMC specified in CEN/ISO/TS 17419.

Annex A
(normative)

ASN.1 modules

A.1 Overview

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

— ITSSremoteMgmt { ISO (1) standard (0) calm-management (24102) remote (2) asnm-1 (1)}.

Online updates of this ASN.1 module, especially amendments to the instantiations of types CLASS, will be published on the ISO maintenance web at http://standards.iso.org/iso/24102/-2.

A.2 Module ITSSremoteMgmt

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.

For updates of this ASN.1 module, see the ISO maintenance web at http://standards.iso.org/iso/24102/-2.

```
ITSSremoteMgmt { iso (1) standard (0) calm-management (24102) remote (2) asnm-1 (1)}

DEFINITIONS AUTOMATIC TAGS::=BEGIN

IMPORTS

NullType, Time48IAT, Lat, Lon FROM CALMllsap {iso(1) standard(0) calm-ll-sap(21218) asnm-1
(1)}

MPARAM, RefMPARAM, Param24102No, Param24102, ApplicationID FROM CALMmanagement { iso (1)
standard (0) calm-management (24102) local (1) asnm-1 (1)}

SFSAP-RR, RefSFSAP-RR, SFSAP-RC, RefSFSAP-RC, MFSAP-RC, RefMFSAP-RC, MFSAP-RR,
RefMFSAP-RR, MFSAP-CC, RefMFSAP-CC, MFSAP-CR, RefMFSAP-CR, MIcommand, MNcommand,
MFcommand, MAcommand, MScommand FROM CALMmsap { iso (1) standard (0) calm-management
(24102) msap (2) asnm-1 (1)}

ITSscuID, ItsScuCmcID, ITSaid, ITSSapPrPr, ITSSapdID, ITSSpPr, ITSaooID, ITSprotID,
ITSSpdID, ITSpoID, ProtocolID FROM CITSapplMgmtApplReg {iso(1) standard(0) cits-applMgmt
(17419) applRegistry (2)}

;

-- End of IMPORTS

-- Types

-- Security header and trailer --
SECRSMP::=CLASS{
       &ref    RefSECRSMP, -- security type identifier
       &SecRSMP
       }

RefSECRSMP::=INTEGER{
       c-noSecurity    (0),
       c-octString     (1)
       }(0..255)
```

```
SecRSMPs SECRSMP::={noSecurity | octString, ...}

noSecurity   SECRSMP::={&ref c-noSecurity, &SecRSMP NullType}
octString    SECRSMP::={&ref c-octString, &SecRSMP OctStringSec}

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

SecHeader::=SEQUENCE{
       secRef   SECRSMP.&ref({SecRSMPs}),
       secHead  SECRSMP.&SecRSMP({SecRSMPs}{@secRef})
       }

SecTrailer::=SEQUENCE{
       secRef    SECRSMP.&ref({SecRSMPs}),
       secTrail  SECRSMP.&SecRSMP({SecRSMPs}{@secRef})
       }

-- RSMP-Request and RSMP-Response common parts
-- RSMP PDU-ID
RsmpPDUcounter::=INTEGER(0..65535) -- cyclic counter

RsmpSessionID::=INTEGER(0..65535) -- cyclic counter

RSMPPDU::=CLASS{
       &ref   RefRSMPPDU, -- management request type identifier
       &RSMPpdu
       }

RefRSMPPDU::=INTEGER{
       c-requestPDU    (0),
       c-responsePDU   (1)
       } (0..255)

RSMPpdus   RSMPPDU::={requestPDU | responsePDU, ...}

requestPDU   RSMPPDU::={&ref c-requestPDU, &RSMPpdu RequestData}
responsePDU  RSMPPDU::={&ref c-responsePDU, &RSMPpdu ResponseData}

RSMPmessage::=SEQUENCE{
       sessionID   RsmpSessionID,
       pduCounter  RsmpPDUcounter,
       pduID  RSMPPDU.&ref({RSMPpdus}),
       pdu    RSMPPDU.&RSMPpdu({RSMPpdus}{@pduID})
       }

-- RSMP-Request

RSMPREQ::=CLASS{
       &ref  RefRSMPREQ, -- management request type identifier
       &RSMPrequest
       }

-- RqDataIDs
RefRSMPREQ::=INTEGER{
       c-pingRq              (0),
       c-rmcReqSession       (1),
       c-rmsInitSession      (2),
       c-rmsCloseSession     (3),
       c-getMparams          (4),
       c-setMparams          (5),
       c-firmwareUpdate      (6),
       c-protMaintenance     (7),
       c-appMaintenance      (8)
       } (0..255)

RSMPrequests RSMPREQ::={pingRq | rmcReqSession | rmsInitSession | rmsCloseSession |
getMparams | setMparams | firmwareUpdate | protMaintenance | appMaintenance, ...}

pingRq RSMPREQ::={&ref c-pingRq, &RSMPrequest RSMPping}
rmcReqSession RSMPREQ::={&ref c-rmcReqSession, &RSMPrequest RMCreqSession}
rmsInitSession      RSMPREQ::={&ref c-rmsInitSession, &RSMPrequest RMSinitSession}
```

```
rmsCloseSession     RSMPREQ::={&ref c-rmsCloseSession, &RSMPrequest RMScloseSession}
getMparams    RSMPREQ::={&ref c-getMparams, &RSMPrequest GetMparams}
setMparams    RSMPREQ::={&ref c-setMparams, &RSMPrequest SetMparams}
firmwareUpdate RSMPREQ::={&ref c-firmwareUpdate, &RSMPrequest FirmwareUpdate}
protMaintenance     RSMPREQ::={&ref c-protMaintenance, &RSMPrequest ProtMaintenance}
appMaintenance      RSMPREQ::={&ref c-appMaintenance, &RSMPrequest AppMaintenance}


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

-- RMC request for secure session initiation
-- Class to indicate reason for session initiation request by RMC
RMCREQREASON::=CLASS{
       &ref   RefRMCREQREASON, -- reason identifier
       &RMCreqReason
       }

-- IDs of reasons for session initiation request by RMC
RefRMCREQREASON::=INTEGER{
       c-rmcRqNoReason       (0), -- maybe RMS has news?
       c-rmcRqExceptLogFile  (1), -- exception occurred. RMS should read logfile.
       c-rmcRqRegulUpdate    (2) -- need regulatory info update.
       } (0..255)

RMCreqReasons RMCREQREASON::={rmcRqNoReason | rmcRqExceptLogFile | rmcRqRegulUpdate, ...}

rmcRqNoReason       RMCREQREASON::={&ref c-rmcRqNoReason, &RMCreqReason NullType}
rmcRqExceptLogFile  RMCREQREASON::={&ref c-rmcRqExceptLogFile, &RMCreqReason ExceptionID}
rmcRqRegulUpdate    RMCREQREASON::={&ref c-rmcRqRegulUpdate, &RMCreqReason RegulUpdateRq}

ExceptionID::=INTEGER{
       exceptUnknown  (0)
       }

RegulUpdateRq::=SEQUENCE{
       -- ID of regulatory issue
       lat   Lat, -- latitude of RMC position
       lon   Lon -- longitude of RMC position
       }

RMCreqReas::=SEQUENCE{
       rmcRqReasonRef   RMCREQREASON.&ref({RMCreqReasons}),
       reason RMCREQREASON.&RMCreqReason({RMCreqReasons}{@rmcRqReasonRef})
       }

RMCreqSession::=SEQUENCE{
       itsscuID  ITSscuID, -- ITS-SCU-ID
       itsScuCmcID   ItsScuCmcID, -- unique ID of intended RMS
       reqReason     RMCreqReas, -- reason for session request
       prevSessionID RsmpSessionID -- previous sessionID with intended RMS
       }

RMSinitSession::=SEQUENCE{
       itsScuCmcID   ItsScuCmcID, -- unique ID of RMS
       itsscuID   ITSscuID -- ITS-SCU-ID
       }

RMScloseSession::=SEQUENCE{
       itsScuCmcID   ItsScuCmcID, -- unique ID of intended RMS
       itsscuID   ITSscuID -- ITS-SCU-ID
       }

GetMparams::=SEQUENCE OF Param24102No

SetMparams::=SEQUENCE OF Param24102

FirmwareUpdate::=SEQUENCE{
       firmware   OCTET STRING -- specific to management centre
       }

PROTMNT::=CLASS{
```

```
       &ref   RefMNT,
       &PROTmntc
       }

ProtMaintenance::=SEQUENCE{
       protMntRef   PROTMNT.&ref({ProtMntRqs}),
       protMgmtTask   PROTMNT.&PROTmntc({ProtMntRqs}{@protMntRef})
       }

RefMNT::=INTEGER{
       c-MntInstall   (0),
       c-MntUpdate    (1),
       c-MntDelete    (2)
       }

ProtMntRqs PROTMNT::={installProtRq | updateProtRq | deleteProtRq, ...}

installProtRq PROTMNT::={&ref c-MntInstall, &PROTmntc ProtMgmtInstallRq}
updateProtRq PROTMNT::={&ref c-MntUpdate, &PROTmntc ProtMgmtUpdateRq}
deleteProtRq PROTMNT::={&ref c-MntDelete, &PROTmntc ProtMgmtDeleteRq}

ProtMgmtInstallRq::=SEQUENCE{
       protID    ITSprotID, -- ITS protocol identifier
       itspoID   ITSpoID,   -- ITS protocol owner
       itsspdID  ITSSpdID,  -- ITS-S protocol developer
       itsspPr   ITSSpPr,   -- ITS-S protocol provisioner
       protCode  ProtCode   -- software to be installed
       }

ProtCode::=OCTET STRING

ProtMgmtUpdateRq::=SEQUENCE{
       protI       ProtocolID, -- of ITS-S protocol to be updated
       itspoID     ITSpoID,    -- ITS protocol owner
       itsspdID    ITSSpdID,   -- ITS-S protocol developer
       itsspPr     ITSSpPr,    -- ITS-S protocol provisioner
       protPackage ProtCode    -- software to be installed
       }

ProtMgmtDeleteRq::=ProtocolID  -- of ITS-S protocol to be deleted

APPMNT::=CLASS{
       &ref   RefMNT,
       &APPmntc
       }

AppMaintenance::=SEQUENCE{
       appMntRef   APPMNT.&ref({AppMntRqs}),
       appMgmtTask APPMNT.&APPmntc({AppMntRqs}{@appMntRef})
       }

AppMntRqs APPMNT::={installAppRq | updateAppRq | deleteAppRq, ...}

installAppRq APPMNT::={&ref c-MntInstall, &APPmntc AppMgmtInstallRq}
updateAppRq APPMNT::={&ref c-MntUpdate, &APPmntc AppMgmtUpdateRq}
deleteAppRq APPMNT::={&ref c-MntDelete, &APPmntc AppMgmtDeleteRq}


AppMgmtInstallRq::=SEQUENCE{
       itsAID      ITSaid,      -- ITS application identifier
       itsaooID    ITSaooID,    -- ITS application object owner
       itssapdID   ITSSapdID,   -- ITS-S application process developer
       itssapPrPr  ITSSapPrPr,  -- ITS-S application process provisioner
       appPackage  OCTET STRING -- software to be installed
       }

AppMgmtUpdateRq::=SEQUENCE{
       appID  ApplicationID, -- of ITS-S application process to be updated
       itsaooID ITSaooID,    -- ITS application object owner
       itssapdID ITSSapdID,  -- ITS-S application process developer
       itssapPrPr ITSSapPrPr,-- ITS-S application process provisioner
```

```
        appCode AppCode        -- software to be installed
        }

AppCode::=OCTET STRING

AppMgmtDeleteRq::=ApplicationID -- of ITS-S application process to be deleted


RequestData::=SEQUENCE{
        rsmpRqRef    RSMPREQ.&ref({RSMPrequests}),
        request      RSMPREQ.&RSMPrequest({RSMPrequests}{@rsmpRqRef})
        }

-- RSMP-Response

RSMPRES::=CLASS{
        &ref  RefRSMPRES, -- management response type identifier
        &RSMPresponse
        }

-- RsDataIDs
RefRSMPRES::=INTEGER{
        c-pingRs              (0),
        c-nullResponse        (1),
        c-rmsInitSessionRs    (2),
        c-rmsCloseSessionRs   (3),
        c-getMparamsRs        (4),
        c-setMparamsRs        (5),
        c-firmwareUpdateRs    (6),
        c-protMaintenanceRs   (7),
        c-appMaintenanceRs    (8)
        }(0..255)

RSMPresponses RSMPRES::={pingRs | nullResponse | rmsInitSessionRs | rmsCloseSessionRs |
getMparamsRs | setMparamsRs | firmwareUpdateRs | protMaintenanceRs | appMaintenanceRs, ...}

pingRs  RSMPRES::={&ref c-pingRs, &RSMPresponse RSMPping}
nullResponse  RSMPRES::={&ref c-nullResponse, &RSMPresponse NullType}
rmsInitSessionRs  RSMPRES::={&ref c-rmsInitSessionRs, &RSMPresponse RMSinitSessionRs}
rmsCloseSessionRs  RSMPRES::={&ref c-rmsCloseSessionRs, &RSMPresponse RMScloseSessionRs}
getMparamsRs  RSMPRES::={&ref c-getMparamsRs, &RSMPresponse GetMparamsRs}
setMparamsRs  RSMPRES::={&ref c-setMparamsRs, &RSMPresponse SetMparamsRs}
firmwareUpdateRs  RSMPRES::={&ref c-firmwareUpdateRs, &RSMPresponse FirmwareUpdateRs}
protMaintenanceRs  RSMPRES::={&ref c-protMaintenanceRs, &RSMPresponse ProtMaintenanceRs}
appMaintenanceRs  RSMPRES::={&ref c-appMaintenanceRs, &RSMPresponse AppMaintenanceRs}


RMSinitSessionRs::=SEQUENCE{
        itsScuCmcID  ItsScuCmcID, -- unique ID of RMS
        itsscuID  ITSscuID, -- ITS-SCU-ID
        resultStatus  RSMPErrStatus -- (success / rejected)
        }

RMScloseSessionRs::=SEQUENCE{
        itsScuCmcID  ItsScuCmcID, -- unique ID of intended RMS
        itsscuID  ITSscuID, -- ITS-SCU-ID
        resultStatus  RSMPErrStatus -- (success / rejected)
        }

GetMparamsRs::=SEQUENCE OF Param24102

SetMparamsRs::=SEQUENCE{
        globalStat  RSMPErrStatus, -- success or setErrorGeneral
        detailStat  DetailStatusSetMparams -- present in case of errors
        }

DetailStatusSetMparams::=SEQUENCE (SIZE(0..255)) OF SetMparamStatus

SetMparamStatus::=SEQUENCE{
        paramNo    Param24102No,
        resultCode RSMPErrStatus
```