INTERNATIONAL UNION
OF RAILWAYS

FRMCS FFFIS

Form Fit Functional Interface Specification

Source:	**UIC**
Date:	**29th of April**
UIC Reference:	**FFFIS-7950**
Version:	**2.1.0**
No of pages:	**109**

ISBN 978-2-7461-3120-0

Document history

Version	Date	Details
0.0.1	24.02.2021	Creation of the document
0.0.2 to 0.0.8	08.07.2021	Interim internal versions
0.0.9	13.07.2021	First version reviewed internally
0.0.10	23.07.2021	Add description of API features
0.0.11	28.07.2021	Modifications after first review with industries
0.0.12	29.07.2021	Modifications after additional review with industries
0.1.0	02.08.2021	Draft for Review (S2R Consortium)
0.1.1	08.10.2021	Interim version including all comments received from S2R
0.1.2	16.11.2021	Interim version with consolidation of content
0.1.3	22.11.2021	Modifications after internal review
0.2.0	22.11.2021	Second Draft for Review (S2R Consortium)
0.2.1	08.12.2021	Modifications to reflect all S2R Consortium comments
0.2.2	16.12.2021	Modifications after internal review
0.3.0	17.12.2021	Stable FFFIS draft content mainly applicable to OB_{APP} for last consortium review
0.3.1	18.01.2022	Modifications after comments received from Kontron
0.4.0	21.01.2022	Final FFFIS draft with content mainly applicable to $OB_{APP.}$ For ERA EECT Review as official deliverable of SC3/SC4
0.4.1	29.03.2022	Update to take into account EECT comments
0.4.2	15/04/2022	Clarification of API parameters and update to reflect EECT comments (06/04/22)
0.5.0	06/05/2022	Consolidated FFFIS final draft to consider EECT review comments and API parameters evolutions
0.5.1	10/06/2022	Consolidation of IP negotiation parameters during Session start
0.6.0	30/06/2022	Consolidated FFFIS final draft to consider EECT review comments (round #3) and IP negotiation evolutions
0.6.1	2/08/2022	Update of API parameter structure and main comments from EECT
0.7.0	19/08/2022	Consolidated FFFIS with parameters and API messages encoded in ASN.1 format
0.7.1	23/09/2022	Consolidated FFFIS following open points resolutions work frame
0.8.0	27/09/2022	Update to take into account EECT review comments (09/09)
0.9.0	11/10/2022	Amendments from last EECT review round (EECT meeting on 7/10/2022)
0.10.0	18/10/2022	Amendments from last EECT review round (EECT meeting on 18/10/2022)
1.0.0	12/02/2023	Modifications proposed by ERA through “agency consistency check on FIS and FFFIS” document and new Annex added to present the "*Interoperability requirements in EU”* coming from “Agency proposal for categorisation annexes for RMR Baseline 0” document.

Version	Date	Details
1.1.0	29/03/2024	Release delivery.
1.2.0	10/05/2024	First delivery to EECT Review
1.2.1	27/09/2024	Second delivery to EECT Review + ASN1 syntax corrections
1.2.2	12/11/2024	Third delivery to EECT Review + editorials including ASN1 syntax corrections
1.2.3	26/11/2024	Forth delivery to EECT Review (alignment due to comments #104 and #105 within Subset-037-3 review sheet)
2.0.0	11/12/2024	Delivery of FRMCS v2, including the editorial changes requested by ERA.
2.0.1	24/03/2025	Addition of candidate MI tables in Annex D
2.1.0	29/04/2025	Final MI candidate after EECT review.

Table of Contents

1 List of abbreviations

3GPP	3rd Generation Partnership Project
API	Application Programming Interface
ATO	Automatic Train Operation
CP	Control Plane
ERTMS	European Rail Traffic Management System
ETCS	European Train Control System
FRMCS	Future Railway Mobile Communication System
GW	Gateway
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
MCX	3GPP Mission Critical Services
OB	On-Board
OB_{APP}	On-Board Application reference point/interface
QoS	Quality of service
SIP	Session Initiation Protocol
SSE	Server Sent Event
TCMS	Train Control and Management System
TLS	Transport Layer Security
TOBA	Telecom On-Board Architecture
TS	Trackside
TS_{APP}	Trackside Application reference point/interface
TSI	Technical Specification for Interoperability
TSI CCS	Control Command and Signalling TSI
UIC	Union Internationale des Chemins de Fer
UP	User Plane

2 List of definitions

Application

Provides a solution for a specific communication need that is necessary for railway operations. In the context of this document, an application interfaces with the On-Board FRMCS through the OB_{APP} reference point and with FRMCS Trackside Gateway through the TS_{APP} reference point.

Control Plane

The Control Plane (CP) carries signalling traffic between the network entities. Control plane and User Plane are to be considered independently of one another and can accordingly be managed separately between entities.

FRMCS Domain

A FRMCS Domain is an administrative domain which comprises a Service Domain and a Transport Domain under the control of an FRMCS Operator.

FRMCS System

Telecommunication system conforming to FRMCS specifications.

FRMCS Service client

Client that enables the use of the Communication Services and/or Complementary Services for the railway applications.

FRMCS Service server

Server that enables the use of the Communication Services and/or Complementary Services for the railway applications.

On-Board FRMCS

System enabling FRMCS communication to on-board applications. The On-Board FRMCS achieves a decoupling between On-Board Application(s) and transport service. For some applications, the decoupling is also achieved for the communication service.

FRMCS Trackside Gateway

System enabling FRMCS communication to trackside applications. The Trackside FRMCS achieves a decoupling between Trackside Application(s) and transport service. For some applications, the decoupling is also achieved for the communication service.

Interface

In this FFFIS, Interface and Reference Point describe the same notion, where Reference Point is used when discussing architecture, whereas Interface is the word used for the specification.

Low Layers

The term “low layers” corresponds to the OSI (Open Systems Interconnection) layers below the Application layer in the context of this FFFIS.

Lower Layers

The term “lower layers” originates from the UNIFE Working Group “FRMCS Lower Layers Requirements” and corresponds to the OSI layers 3 and below in the context of an on-board common bus.

Reference Point

Conceptual point applicable for interaction between functional services that enables authorised functions, e.g. in the network, to access their services. In this FFFIS, Interface and Reference Point describe the same notion, where Reference Point is used when discussing architecture, whereas Interface is the word used for the specification.

Transport service

It is a service that provides transport of user information and control signals between corresponding reference points considering the required QoS for the individual communication.

User Plane

The User Plane (UP) carries the user/application traffic. For the exchange of information between the communication partners (payload), the User Plane provides the necessary formats in order to provide the desired quality. Voice, video and data require different formats, for instance Codec to enable communication between partners. This is determined by the corresponding User Plane instance on the application side and controlled accordingly.

3 References

3.1 Applicability

3.1.1.1 References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.

3.1.1.2 For a specific reference, subsequent revisions do not apply.

3.1.1.3 For a non-specific reference, the latest version applies.

3.2 List of References

[FRMCS-FRS]	UIC, FRMCS, Functional Requirements Specification, FU-7120
[FRMCS-SRS]	UIC, FRMCS, System Requirements Specification, FW-AT-7800
[TOBA-FRS]	UIC**,** FRMCS, On-Board FRMCS – Functional Requirements Specification, TOBA-7510
[FRMCS-FIS]	UIC, FRMCS, Functional Interface Specification, FIS-7970
[SUBSET-147]	UNISIG ERTMS/ETCS and ATO over ETCS – FFFIS part: Communication Layers
[RFC 9113]	Hypertext Transfer Protocol Version 2 (HTTP/2) specifications.
[RFC 8259]	The JavaScript Object Notation (JSON) Data Interchange Format.
[RFC 4122]	A Universally Unique IDentifier (UUID) URN Namespace.
[RFC 1166]	Internet Numbers.
[RFC 5952]	A recommendation for IPv6 address text representation.
[RFC 3986]	Uniform Resource Identifier (URI): Generic Syntax.
[RFC 8446]	The Transport Layer Security (TLS) Protocol Version 1.3.
[3GPP TS 29.571]	5G System; Common Data Types for Service Based Interfaces; Stage 3.
[3GPP TS 29.500]	5G System; Technical Realization of Service Based Architecture; Stage 3.
[3GPP TS 23.032]	Universal Geographical Area Description (GAD).

4 Introduction

4.1 Purpose of this document

4.1.1 This Form Fit Functional Interface Specification (FFFIS) specifies the following interfaces: (I)

4.1.1.1 **OB$_{APP}$**, reference point between the On-Board Applications and the On-Board FRMCS, which is defined in **[FRMCS-SRS]**,

4.1.1.2 and **TS$_{APP}$**, reference point between the FRMCS Trackside Gateway and the Trackside Applications, which is defined in **[FRMCS-SRS]**.

4.1.2 Figure 1 below is a simplified FRMCS architecture. It depicts the main high-level functional blocks and indicates the location of the OB$_{APP}$ and TS$_{APP}$ interfaces. (I)

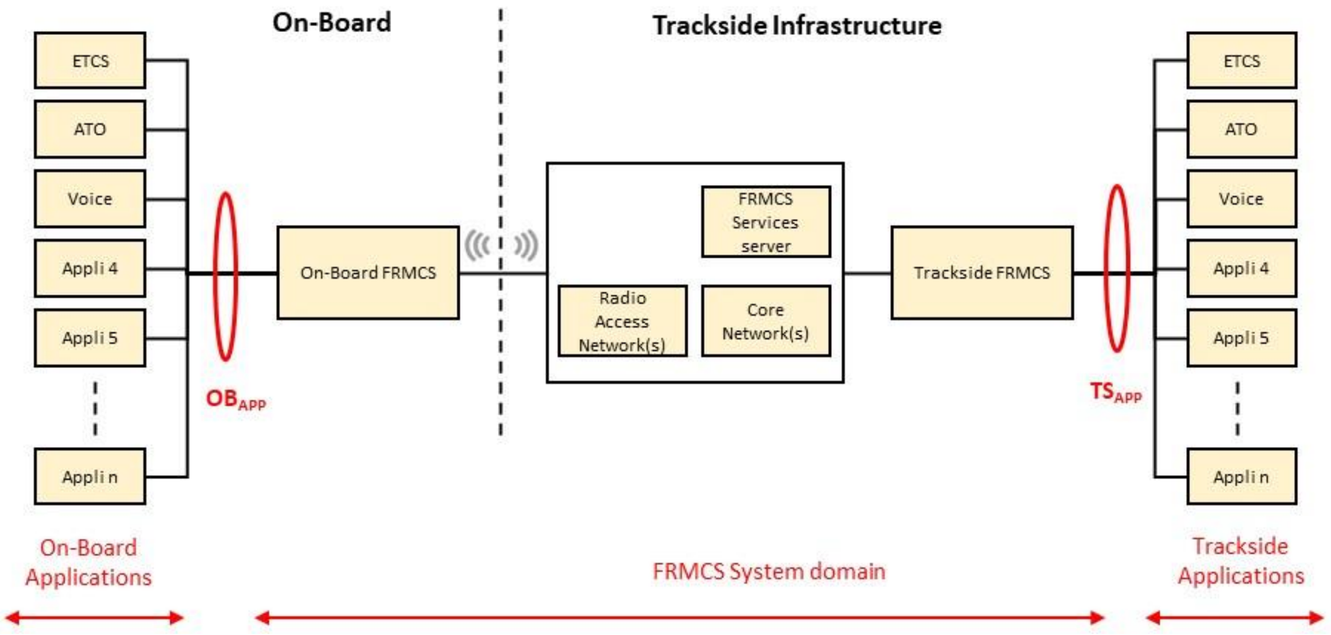


Figure 4-1: Positions of OB$_{APP}$ and TS$_{APP}$ interfaces

Note: the difference between Interface and Reference Point is given in chapter 2 (List of definitions).

4.2 Scope of this document

4.2.1 This FFFIS specifies the protocols, the messages and the format of the information exchanged over the OB$_{APP}$ and TS$_{APP}$ interfaces which enable interfacing between applications and the FRMCS System. (I)

4.2.2 This FFFIS cannot be used separately as the FRMCS specifications (**[FRMCS-FRS]**, **[FRMCS-SRS]**, **[FRMCS-FIS]** and **[TOBA-FRS]**) have to be considered as a whole. (I)

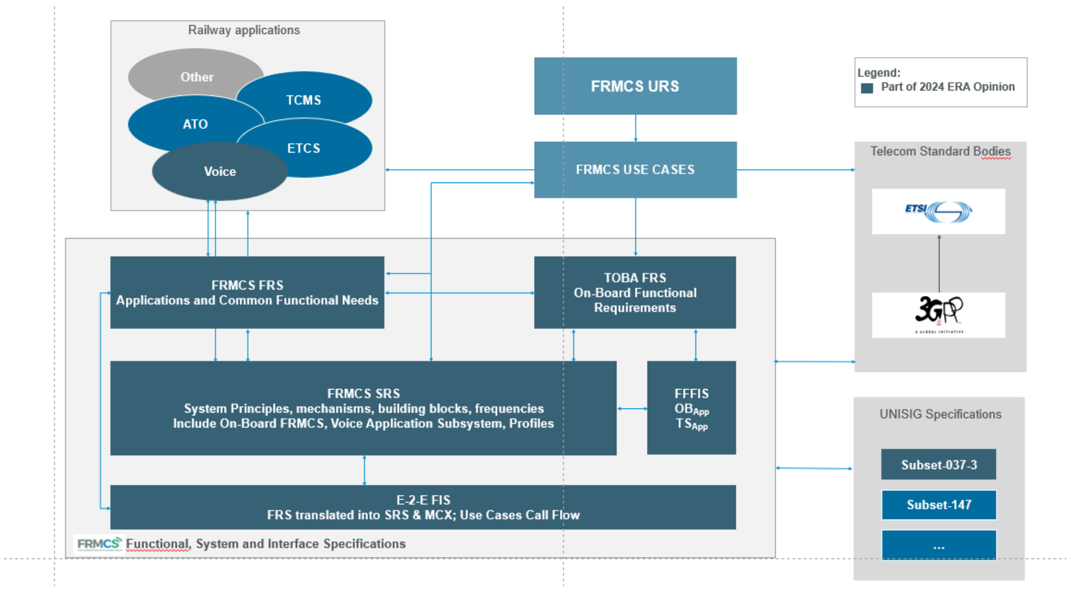

Railway applications
Other
TCMS
ATO
ETCS
Voice
FRMCS URS
FRMCS USE CASES
Legend:
Part of 2024 ERA Opinion
Telecom Standard Bodies
ETSI
3GPP
FRMCS FRS
Applications and Common Functional Needs
TOBA FRS
On-Board Functional Requirements
FRMCS SRS
System Principles, mechanisms, building blocks, frequencies
Include On-Board FRMCS, Voice Application Subsystem, Profiles
FFFIS
OBApp
TSApp
UNISIG Specifications
Subset-037-3
Subset-147
...
E-2-E FIS
FRS translated into SRS & MCX; Use Cases Call Flow
FRMCS Functional, System and Interface Specifications

Figure 4-2: FRMCS specifications

4.2.3 This FFFIS is part of the FRMCS specifications as depicted in Figure 4-2: (I)

4.2.4 The performance and security requirements applicable to OB_{APP} and TS_{APP} interfaces are defined in chapter 6. (I)

4.2.5 An On-Board Application interfacing On-Board FRMCS uses the low layers defined in chapter 7. This FFFIS does not assume a train common bus in all cases (named Ethernet Consist Network in TSI CCS), but only refers to **[SUBSET-147]** for the case there is a common bus or when some specific requirements in **[SUBSET-147]** to be used by this specification even in case of absence of common bus deployment. (I)

4.2.6 A Trackside Application interfacing the FRMCS Trackside Gateway uses the low layers defined in chapter 8 (I).

4.2.7 The On-Board FRMCS exposes the API defined in chapter 9 to On-Board applications. (I)

4.2.8 The FRMCS Trackside Gateway exposes the API defined in chapter 10 to Trackside applications. (I)

4.3 Categorization of requirements

4.3.1 The requirements are categorised as follows (I):

4.3.1.1 Mandatory for the System (indicated by '(M)' at the end of the clause). These requirements mean a condition set out in this specification that must be met without exception in order to deliver a system ensuring the fulfilment of essential functional and system needs, compliance to relevant standards and technical integration. The mandatory requirements are identified as sentences using the keyword "shall".

4.3.1.2 Optional for the system (indicated by '(O)' at the end of the clause). These requirements may be used based on the implementers' choice. When an option is selected, the related requirement(s) of this specification becomes mandatory for the system. The optional requirements are identified as sentences using the keyword "should".

4.3.1.3 Information (indicated by "(I)" at the end of the clause). These statements provide additional information to help the reader understanding a requirement.

4.3.2 The following marking is applied to denote the applicability of clauses: (I)

a) Indications (M), (O) and (I) are used for clauses within the scope of the V2 specification, which is the minimum set of requirements for validation;
b) Indications (M-V3), (O-V3) and (I-V3) are used for clauses within the scope of the V3 specification. The V3 series of specifications are the target version to be included in the TSI, to allow migration from the GSM-R system to the FRMCS system (FRMCS 1st edition). The V3 clauses are to be considered for information for V2;
c) Indications (M-Vx), (O-Vx) and (I-Vx) are used for clauses for a later version of the specification. These clauses are kept in the specification for readability and consistency purposes;
d) Indications (M-V3), (O-V3), (I-V3) and (M-Vx), (O-Vx), (I-Vx) may also be used for sub bullets within a clause to identify a different applicability. In this case each bullet will be indicated individually.

5 General principles

Note: this chapter is for information purpose only. It provides a description of the FRMCS messages going through the OB_{APP} and TS_{APP} leading to a better understanding of the different modes to be supported. The FRMCS end-to-end information is provided in the FRMCS Functional Interface Specifications **[FRMCS-FIS]**.

5.1 OB_{APP}: Interface between On-Board Applications(s) and On-Board FRMCS

5.1.1 The OB_{APP} corresponds to the interface between the On-Board Application(s) and the On-Board FRMCS. This interface ensures management of and access to the communication services allowing the authentication, authorisation, priority and quality of service profile management requested by those applications. (I)

Note: information regarding the authentication and authorisation mechanisms can be found in the section 6.3.

5.1.2 User Plane data from and to the application(s) is carried over the OB_{APP} interface. (I)

5.1.3 Control Plane data exchange between application and On-Board FRMCS is performed over the OB_{APP} interface. (I)

5.2 API Functions supported through the OB_{APP} interface

5.2.1 The OB_{APP} Control Plane exposes three main functions: (I)

5.2.1.1 Local Binding function: The Local Binding function provides functionalities to establish a secure link between an On-Board Application and the On-Board FRMCS, ensuring mutual authentication of both parties through the OB_{APP} as well as the integrity and confidentiality of the information exchanges related to the OB_{APP} Control Plane. The Local Binding function is spread over several mechanisms described in section 6.3 for the OB_{APP} Security requirements and in chapter 9 for the API services, namely, the local registration and opening the notification event stream;

5.2.1.2 Session function: The Session function provides functionalities to establish or terminate connectivity to or from a remote end point for applications operating in Loose Coupled mode. It is implemented through the API Service session features described in chapter 9;

5.2.1.3 Auxiliary/Notification function: This function enables the applications to subscribe / unsubscribe to one or more notification channel(s) (e.g., location reporting notifications, etc) exposed by the On-Board FRMCS.. The Notification function is implemented through the API notification services described in chapter 9.

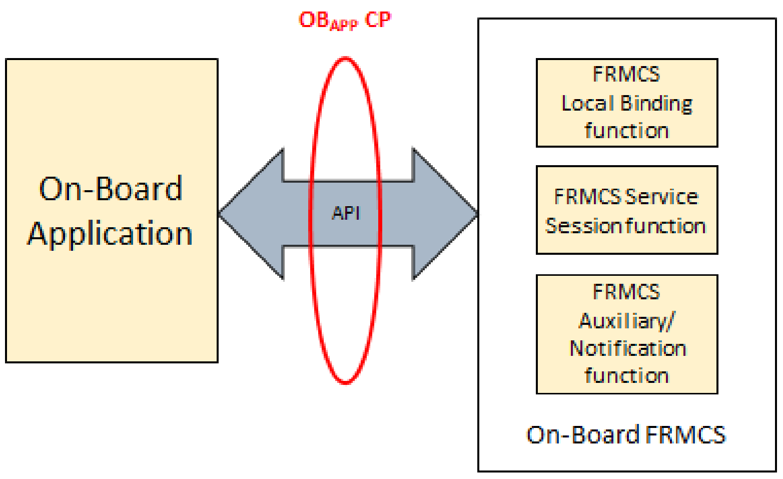

OB_APP CP
On-Board Application
API
FRMCS Local Binding function
FRMCS Service Session function
FRMCS Auxiliary/ Notification function
On-Board FRMCS

Figure 5-1: API features exposed by the OB$_{APP}$ Control Plane interface

5.3 TS$_{APP}$: Interface between Trackside Applications(s) and FRMCS Trackside Gateway

5.3.1 The TS$_{APP}$ corresponds to the interface between the Trackside Application(s) and the FRMCS Trackside Gateway. This interface ensures management of and access to the communication services allowing the authentication, authorisation, priority and quality of service profile management requested by those applications. (I)

5.3.2 User Plane data from and to the application(s) is carried over the TS$_{APP}$ interface. (I)

5.3.3 Control Plane data exchange between application and FRMCS Trackside Gateway is performed over the TS$_{APP}$ interface. (I)

5.4 API Functions supported through the TS$_{APP}$ interface

5.4.1 The TS$_{APP}$ Control Plane exposes three main functions: (I)

5.4.1.1 Local Binding function: The Local Binding function provides functionalities to establish a secure link between a Trackside Application and the FRMCS Trackside Gateway, ensuring mutual authentication of both parties through the TS$_{APP}$ as well as the integrity and confidentiality of the information exchanges related to the TS$_{APP}$ Control Plane. The Local Binding function is spread over several mechanisms described in section 6.6 for the TS$_{APP}$ Security requirements and in chapter 10 for the API services, namely, the local registration and opening the notification event stream;

5.4.1.2 Session function: The Session function provides functionalities to establish or terminate connectivity to or from a remote end point for applications operating in

Loose Coupled mode. It is implemented through the API Service session features described in chapter 10;

5.4.1.3 Auxiliary/Notification function: This function enables applications to subscribe / unsubscribe to one or more notification channel(s) exposed by the FRMCS Trackside Gateway. The API notification service is described in chapter 10.

5.5 <Intentionally Deleted>

5.6 FRMCS Service session in Tight Coupled mode

Note: The figure below depicts the Service session exchanges in Tight Coupled mode. The Local Binding function, Notification function and SIP Core are not shown in the figure.

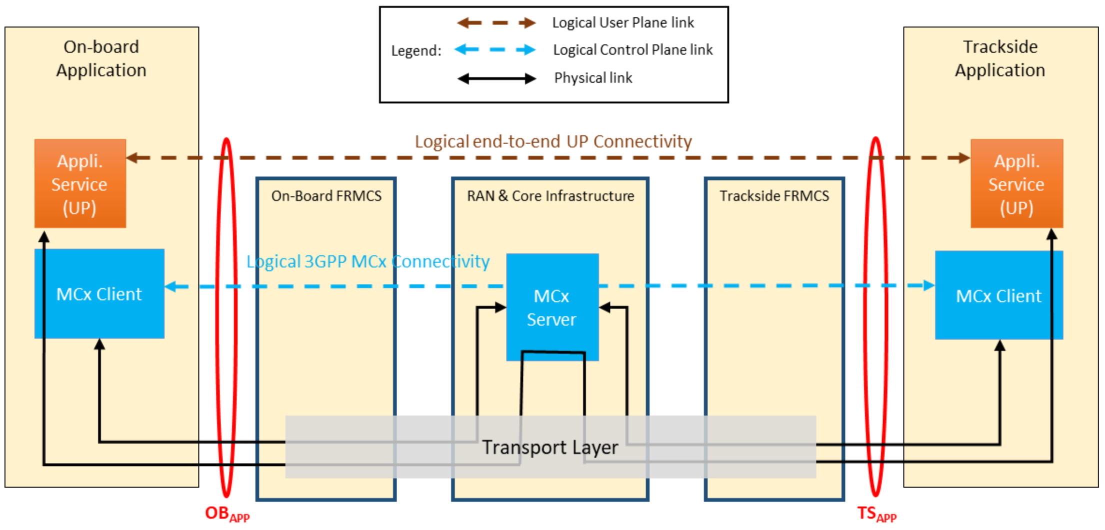


Figure 5-2: End-to-End Service session for Applications in Tight coupled mode

5.6.1 In Tight Coupled mode, after the Local Binding (see section 5.2.1) has been successfully performed, the embedded MCX client of the application performs the subsequent 3GPP MCX protocol exchanges over the IP interface of OB_{APP} / TS_{APP}. (I)

5.6.2 In Tight Coupled mode, the Application User Plane is also carried out over the IP interface of OB_{APP} / TS_{APP}. (I)

5.7 FRMCS Service session in Loose Coupled mode

Note: The figure below depicts the Service session exchanges in Loose Coupled mode The Local Binding function, Notification function and SIP Core are not shown in this figure.

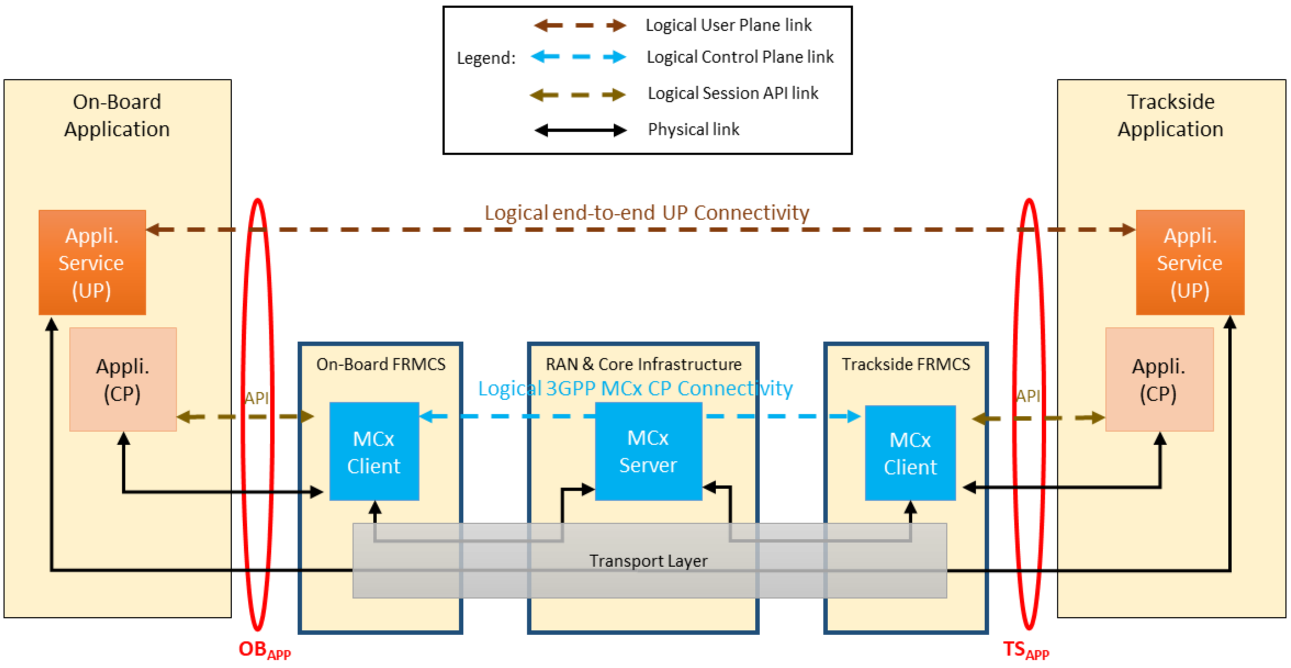


Figure 5-3: End-to-End Service session for Applications in Loose coupled mode

5.7.1 In Loose Coupled mode, after the Local Binding (see sections 5.2.1 and 5.4.1) has been successfully performed, the Application requests the FRMCS Domain to establish a logical Application Control Plane based on 3GPP MCX on its behalf. It does so by calling a dedicated application interface (API) exposed by the On-Board FRMCS or by the FRMCS Trackside Gateway. The features supported by this API are described in the API Services chapters (refer to chapter 9 and section 10). The On-Board FRMCS and FRMCS Trackside Gateway are in charge to translate these API calls into the relevant of 3GPP MCX procedures with the necessary information. (I)

5.7.2 In Loose Coupled mode, the Application User Plane is carried out through the OB_{APP} and TS_{APP} over IP. (I)

6 Performance and Security

This chapter provides the requirements in terms of performance and security for both OB_{APP} and TS_{APP}.

6.1 OB_{APP} Performance requirements

6.1.1 The physical layer of the OB_{APP} interface at On-Board FRMCS side supports the minimum gross data rate defined for layer 1 of Ethernet Consist Network (CCS) in **[SUBSET-147]** (see clause 7.2.2). (I)

6.2 <Intentionally Deleted>

6.3 OB_{APP} Security requirements

6.3.1 If an FRMCS On-Board is connected to an Ethernet Consist Network compliant with **[SUBSET-147]**, the interface shall comply with the authentication mechanisms specified in **[SUBSET-147]**. (M)

6.3.2 On the OB_{APP} Control Plane, a mutual authentication based on client and server certificates shall be performed between the application and the On-Board FRMCS using the Transport Layer Security (TLS) protocol. During the TLS handshake, client (application) and server (On-Board FRMCS) send their certificate and authenticate themselves. (M)

6.3.3 The integrity and confidentiality protection of the OB_{APP} Control Plane implemented through the API features shall rely on the Transport Layer Security (TLS) protocol. (M)

6.3.4 The TLS end points shall support TLS 1.3. (**[RFC 8446]**). (M)

6.4 TS_{APP} Performance requirements

6.4.1 The data rate on TS_{APP} interface depends essentially on 1) the size of the operated railway infrastructure and the traffic volume, 2) whether the load is distributed over multiple FRMCS Trackside Gateways. This is fully dependant on implementation choice of the Railway infrastructure manager and is outside the scope of this FFFIS. (I)

6.5 <Intentionally Deleted>

6.6 TS_{APP} Security requirements

6.6.1 On the TS_{APP} Control Plane, a mutual authentication based on client and server certificates shall be performed between the application and the FRMCS Trackside Gateway using the Transport Layer Security (TLS) protocol. During the TLS handshake, client (application) and server (FRMCS Trackside Gateway) send their certificate and authenticate themselves. (M)

6.6.2 The integrity and confidentiality protection of the TS_{APP} Control Plane implemented through the API features shall rely on the Transport Layer Security (TLS) protocol. (M)

6.6.3 The TLS end points shall support TLS 1.3 (**[RFC 8446]**). (M)

6.7 TLS requirements

6.7.1 The OB_{APP} shall satisfy the TLS requirements in this clause. (M-V3)

6.7.2 The TS_{APP} shall satisfy the TLS requirements in this clause. (M-V3)

7 OB$_{APP}$ Low layers specifications and protocol stacks

7.1 <Intentionally Deleted>

7.2 OB$_{APP}$ Physical interface

7.2.1 The physical interface of the OB$_{APP}$ at On-Board FRMCS side is made of common off-the-shelf technologies based on Ethernet (IEEE 802.3). (I)

7.2.2 The physical interface of the OB$_{APP}$ at On-Board FRMCS side shall comply with layers 1 and 2 requirements of Ethernet Consist Network (CCS) in **[SUBSET-147]**. (M)

7.3 OB$_{APP}$ Internet Protocol versions

7.3.1 <intentionally deleted>

7.3.2 <intentionally deleted>

7.3.2i The support of IP versions exposed by On-Board FRMCS on OBapp shall comply with [FRMCS-SRS] requirements in section 6.5.1. (M)

7.4 OB$_{APP}$ local IP allocation scheme

7.4.1 At the OB$_{APP}$ interface side, the On-Board FRMCS is seen as a host in the train network and hence it shall be configured in accordance with the IP plan of the train network. (M)

7.4.2 The On-Board FRMCS shall expose on OB$_{APP}$ an IP interface with IP address(es) that can be used by the On-Board Application to send/receive OB$_{APP}$ User Plane and Control Plane data. (M)

7.5 <Intentionally Deleted>

8 TS$_{APP}$ Low layers specifications and protocol stacks

8.1 TS$_{APP}$ Connectivity

8.1.1 The Trackside Applications need to have connectivity to use the FRMCS Trackside Gateway. This connectivity can be established according to different technical choices depending on which device/entity the application is installed, e.g. commercial off-the-shelf (COTS) computer, proprietary fixed equipment. It depends also on the location of the physical Application entities and Trackside FRMCS. (I)

8.1.2 The communication network architecture and distance between the Trackside Application and the FRMCS Trackside Gateway are fully dependant on implementation choice of the Railway infrastructure manager. This is outside the scope of this FFFIS. (I)

8.1.3 In case the application does not support TS$_{APP}$ requirements (physical and/or logical), an agent supporting TS$_{APP}$ is used in between to connect to the FRMCS Trackside Gateway. The physical and logical interface specifications between the application and agent are outside the scope of the FRMCS specifications. (I)

8.2 TS$_{APP}$ Physical interface

8.2.1 The physical interface of the TS$_{APP}$ at FRMCS Trackside Gateway side is made of common off-the-shelf technologies based on Ethernet (IEEE 802.3). (I)

8.2.2 The TS$_{APP}$ interface supports the following physical interface requirements: (I)

- links over copper twisted-pair cable or over fiber-optical cable
- standardized physical connectors, for instance RJ45 or M12 in case of twisted-pair cable or 10GBASE-SR or LR connector in case of fiber-optical cable.

8.3 TS$_{APP}$ Internet Protocol versions

8.3.1 <intentionally deleted>

8.3.2 <intentionally deleted>

8.3.2i The support of IP versions exposed by FRMCS Trackside Gateway on TSapp shall comply with [FRMCS-SRS] requirements in section 6.5.1. (M)

8.4 TS$_{APP}$ local IP allocation scheme

8.4.1 The FRMCS Trackside Gateway shall expose on TS$_{APP}$ an IP interface with an IP gateway address that can be used by the Trackside Applications to send/receive TS$_{APP}$ User Plane and Control Plane data. (M)

8.5 <Intentionally Deleted>

9 OB_{APP} API Services

9.1 Overview of OB_{APP} API features

OB_{APP} enables the following services between an application and the On-Board FRMCS:

9.1.1 **API version**: This OB_{APP} service is used by On-Board Application to obtain the list of API version(s) supported by the On-Board FRMCS. (I)

9.1.2 **Local registration**: This OB_{APP} service is used to perform the Local registration between an On-Board Application and the On-Board FRMCS. (I)

9.1.3 **Local deregistration**: This OB_{APP} service is used to request a local de-registration of the On-Board Application from the On-Board FRMCS. (I)

9.1.4 **Session opening**: This OB_{APP} service is used to establish a session between an On-Board Application and a remote (Trackside or On-Board) application at the initiative of the On-Board Application. (I)

9.1.5 **Incoming Session acceptance**: This OB_{APP} service is used as a part of the establishment of a session between an On-Board Application and a remote (Trackside or On-Board) application at the initiative of the remote application. (I)

9.1.6 **Session closure**: This OB_{APP} service is used to close a session between an On-Board Application and a remote application. (I)

9.1.7 **Session status**: This OB_{APP} service is used to provide the status of a session involving the On-Board Application (I)

9.1.8 **Subscription to notification event stream:** This feature is used to request the opening of an event stream enabling On-Board FRMCS to send notifications to the On-Board Application after the local registration. This is done during the local binding. (I)

9.1.9 **General notification:** upon On-Board Application’s subscription to notification event stream, the On-Board Application receive a set of general notifications which are linked to the following events: (I)

- Incoming session request: The On-Board FRMCS notifies the On-Board Application of the reception of an incoming (On-Board terminated) session request.
- Final answer of a session initiation: the On-Board FRMCS notifies the On-Board Application of whether the establishment of the E2E communication session at the initiative of the On-Board Application was successful / failed / declined.
- Availability of FRMCS Transport Domain (FTD): the On-Board FRMCS notifies the On-Board Application of the availability of FRMCS Transport Domain.
- Availability of FRMCS Service Domain (FSD): the On-Board FRMCS notifies the On-Board Application of the availability of FRMCS Service Domain.
- Session closure notification: The On-Board FRMCS notifies the On-Board Application of the closure of an open session over OB_{APP}.

- Upcoming deregistration notification: The On-Board FRMCS notifies the On-Board Application of an imminent deregistration of On-Board FRMCS (e.g., as a preparation for a train turnoff).

Note: These notifications do not require an explicit subscription from application and are implicitly included in the subscription to the notification event stream.

9.1.10 **Subscription/Unsubscription to a notification channel**: the On-Board FRMCS exposes notification channel(s) to which the On-Board Application can subscribe / unsubscribe. Upon a subscription, the On-Board Application receives the notifications corresponding to that specific channel on the notification event stream which is opened during the local binding. (I)

9.1.11 **Location reporting notification:** this notification channel on OB_{APP} is used by On-Board Application to subscribe to the location change notifications in one or several of the following manner: (I)

- Periodic location reporting with a given time interval.
- Location reporting at the occurrence of a cell change.
- Location reporting at each interval of travelled distance.

Note: in the context of the API, the term application refers to the application instance, which is a concrete running software occurrence of an application of a specific type.

9.1.12 The completion of Local Binding shall imply the successful execution of the following steps: (M)

(i) The first step in which an application and the On-Board FRMCS shall mutually authenticate using TLS, which is not part of the API. See section 6.3 for more details;

(ii) And the second step in which an application, through API local registration service, request a registration to the On-Board FRMCS. The success response from On-Board FRMCS completes this step;

(iii) And the third step in which the On-Board application subscribes to the notification event stream. The success response from On-Board FRMCS completes this step.

9.1.13 The invocation of any API services beside API versions, local registration, and opening of notification event stream is conditioned on the successful execution of the Local Binding steps. (I)

9.1.14 Mandatory OB_{APP} API services for different types of applications are covered in section 9.15. (I)

9.2 <Intentionally Deleted>

9.3 <Intentionally Deleted>

9.4 Definition of the parameters used in the API services

9.4.1 A comprehensive description of attributes and some basic data types used for OB_{APP} API services are provided in informative tables, Table 9-1 and Table 9-2, respectively. The data types are formally defined in ASN.1 format in the normative Annex A. (I)

#	Attribute name	Description
1	appCategory	Provides the category of the Application. The value is taken out of an enumerated list of application categories which includes at least standardised categories (e.g., etcs, ato, vas, tcms) defined in this document. This enumerated list can be extended with other (non-standardised) application categories per railway infrastructure manager's or railway undertaking's use, This parameter is a local attribute which can be used by FRMCS to decide how an application can be served..
2	staticId	Unique identifier of the Application within the scope of an On-Board FRMCS.
3	dynamicId	Identifier of the application instance dynamically assigned at the On-Board FRMCS, unique in the scope of the On-Board FRMCS. The format is a Universally Unique Identifier (UUID) version 4, as described in IETF **[RFC 4122]**.
4	apiVersion	The API version of OB_{APP} used by On-Board Application, as a part of API URI. The API version is in format v{MAJOR}.{MINOR}, as defined in clause 9.6. The default value is set to "v1.0".
5	supportedVersionsList	List of supported API versions of OB_{APP} by On-Board FRMCS. The API version is in format v{MAJOR}.{MINOR}, as defined in clause 9.6.
6	couplingMode	The application coupling mode (i.e. Loose coupled, Tight coupled)..
7	uriResource	The resource field in the http error response.
8	cause	The machine-readable failure cause in the http failure response.
9	detail	The human-readable failure cause in the http failure response.
10	recipient	The remote recipient of a session originated by On-Board Application. This parameter is constituted of the address of the remote recipient.
11	sessionsList	List of session IDs of the sessions which are open for a given dynamicId.
12	sessionId	Identifier of the session, unique in the scope of the On-Board FRMCS per dynamicId. The format shall be a Universally Unique Identifier (UUID) version 4, as described in IETF **[RFC 4122]**.
13	remoteId	Remote identifier of an application in the scope of session exchange messages.
14	sessionStatus	The status of a session indicating one of the following three cases: a) the E2E session is succeeded, b) the E2E session is failed, c) the E2E session is declined.
15	nextHopIPAddress	Local On-Board FRMCS IP address to be used by the On-Board Application as local next hop IP address for the User Plane data in case of successful session establishment.
16	destApplicationIPAddr ess	The FRMCS IP address of destination application endpoint to be used by the On-Board Application as destination address for the User Plane data in case of successful session establishment.
17	communicationCatego ry	This parameter reflects the different categories of communication (session) that can be established over the FRMCS for a given application. This parameter is used by the On-Board FRMCS to initiate an end-to-end session. Based on this parameter, the On-Board FRMCS assigns the right communication profile including the QoS level to the session.
18	localAppIPAddress	Local Application IP address to be used by the On-Board FRMCS as destination address for the User Plane data in case of successful session establishment.
19	sessionOriginator	The origin of a session A session on OB_{APP} is either originated by the On-Board Application or is an incoming session which is terminated at On-Board Application.
20	channel	A notification channel which is exposed by On-Board FRMCS e.g., for location update notifications.
21	period	Requested update period for the location reporting notification in seconds, defaulting at infinity, if not present. This element specifies the minimum wait period between consecutive location reports.
22	distance	Strictly positive integer number of meters and defaulting at infinity, if not present. This element is used in the decision of sending a location report based on travelled distance, and specifies the minimum required distance, between the current location and the location of the most recent sending of a location report.
23	subscriptionId	The identity of the notification subscription, which is allocated by On-Board FRMCS, unique in the scope of the On-Board FRMCS per dynamicId.
24	locReportType	Location reporting type which is one of the followings: a) periodic, b)distance travelled, c) cell change.
25	ftdAVL	FALSE if FRMCS Transport Domain (FTD) is not available, TRUE if FTD is available.
26	fsdAVL	FALSE if FRMCS Service Domain (FSD) is not available, TRUE if FSD is available.
27	nwTransiiton	TRUE if the ftdAVL or fsdAVL event is due is network transition, FALSE otherwise.

28	incomingSessionAppResponse	Application response to an incoming session which is one the followings: a) accepted, b) rejected.
29	frmcsDomain	The Target FRMCS Transport Domain of the network transition. This is a string representing the PlmnId in the format "{mcc}-{mnc}" as defined in **[3GPP TS 29.571]** Table 5.4.2-1.
30	servingCellId	The serving Cell ID within FRMCS Transport Domain. This is a string in the format "{PlmnId}.{NrCellID}", where PlmnId and NrCellID are as defined in **[3GPP TS 29.571]** Table 5.4.2-1.
31	longitude	The longitude as a constituent of Train Geographic 2D Position. The geographical longitude as defined in **[3GPP TS 23.032]** clause 6.1.
32	latitude	The latitude as a constituent of Train Geographic 2D Position. The geographical longitude as defined in **[3GPP TS 23.032]** clause 6.1.
33	horizontalAccuracy	The Accuracy of the Train Geographic 2D Position (horizontal accuracy). The geographical longitude as defined in **[3GPP TS 23.032]** clause 6.2b.
34	gnssInformation	This attribute regroups the following information elements of the GNSS coordinate: longitude, latitude, horizontalAccuracy, speed, speedAccuracy, and direction.
35	timeStamp	The time stamp of location report. A string with format DateTime as defined in **[3GPP TS 29.571]** Table 5.2.2-1.
36	speed	It represents the train speed. The speed as defined in **[3GPP TS 23.032]** clause 8.7. This is an integer between 0 and 2^{16}-1.
37	speedAccuracy	It represents the train speed accuracy. The speed as defined in **[3GPP TS 23.032]** clause 8.11. This is an integer between 0 and 2^{8}-1.
38	direction	It represents the train bearing. The direction as defined in **[3GPP TS 23.032]** clause 8.8. This is an integer between 0 and 359 (degree).
39	timeToDeregistration	This allows to indicate to the application with which delay (in seconds) after the reception of the upcomingDeregistrationNotif the On-Board FRMCS will turn off.

Table 9-1: Definition of the parameter types that are used in the API request / response

Type Name	**Description**
Uuid	The format shall be a Universally Unique Identifier (UUID) version 4, as described in IETF **[RFC 4122]**.
Ipv4Addr	String identifying a IPv4 address formatted in the "dotted decimal" notation as defined in IETF **[RFC 1166]**. Pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])$'
Ipv6Addr	String identifying an IPv6 address formatted according to clause 4 of IETF **[RFC 5952]**. The mixed IPv4 IPv6 notation according to clause 5 of IETF **[RFC 5952]** shall not be used. Pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))$' And Pattern: '^((([^:]+:){7}([^:]+))\|((([^:]+:)*[^:]+)?::(([^:]+:)*[^:]+)?))$'
Uri	String providing an URI formatted according to IETF **[RFC 3986]**. If the URI fields intended to convey generic data (e.g., in the value part of a query parameter, or in the URI path segments) contain reserved characters, these reserved characters shall be percent-encoded as defined in clause 5.2.10.2 of **[3GPP TS 29.500]**.

Table 9-2: Basic Data Types

9.4.2 The OB_{APP} interface to the On-Board FRMCS will have different versions as new features will be introduced. Supported versions are communicated over the OB_{APP}. For each interface version, a change log is maintained, and changes are categorised into Major and Minor categories. (I)

9.4.3 The OB_{APP} API versioning is defined in clause 9.6. (I)

9.4.3i For an OB_{APP} API implemented according to the present FFFIS, the API version shall be set to v0.1. (M)

9.4.4 The dynamicId, sessionId, and subscriptionId shall be random cryptographic identities generated by On-Board FRMCS at runtime. (M)

9.4.5 The appCategory, as defined in Annex B, allows a list of both harmonized and non-harmonized applications. (I)

9.4.6 The field name for non-harmonized applications within appCategory shall include a prefix “ext.”, indicating non-harmonized extension of the application list. (M)

9.4.7 The static identifier of the application shall be unique in the scope of all FRMCS application instances within an On-Board FRMCS. The structure of FRMCS System identities that are used to set up the relevant FRMCS services and communication link(s) with other FRMCS users shall fulfil the requirements as specified in the **[FRMCS-SRS]**. (M)

9.4.8 The remote address of an application in the scope of OB_{APP} session exchange messages shall fulfil the requirements as specified in the **[FRMCS-SRS]** section 11.6.5. (M)

9.5 API URI

9.5.1 The API URI of the OB_{APP} APIs shall be:
{apiRoot}/<apiName>/<apiVersion>/<ResourceName>, with the following components:

- The {apiRoot} shall be set as “https://{localIdApiRoot}”. (M)
- The <apiName> shall be "obapp". (M)
- The <apiVersion> shall be set to “{apiVersion}” (see details in clause 9.6). (M)
- The <ResourceName> shall be set as described in clause 9.8.2. (M)

9.5.2 The localIdApiRoot is of string type with value being deployment-specific, such as the IP address of the On-Board FRMCS within the train IP network or a locally resolvable FQDN (if the train is equipped with a DNS server). (I)

9.6 API version

9.6.1 API version (represented by ApiVersion data type) shall be a string with format “v{MAJOR}.{MINOR}”. (M)

9.6.2 The 1st Field (MAJOR) and the 2nd Field (MINOR) shall contain unsigned integer numbers, and they shall not contain leading zeroes. (M)

9.6.3 Given the format of API version, the version increments follow the rules defined in the following clauses. (I)

9.6.4 The 1st Field (MAJOR) shall be incremented only if the applied change is backward incompatible relative to the earlier, i.e. frozen version of the API. (M)

9.6.5 For a non-frozen API, the first backwards incompatible change(s) relative to the latest frozen version triggers incrementing the 1st Field (MAJOR), while subsequent backwards incompatible changes do not increment the value, until the API stays non-frozen. When the (MAJOR) field is incremented the (MINOR) field will be reset to 0. (I)

9.6.6 The 2nd Field (MINOR) shall be incremented only if the applied change is a backward compatible new feature relative to the earlier, i.e. frozen version of the API. (M)

9.6.7 For a non-frozen API, the first backwards compatible change(s) relative to the latest frozen version triggers incrementing the 2nd Field (MINOR), while subsequent backwards compatible changes do not increment the value, until the API stays non-frozen. (I)

9.6.8 An On-Board Application which wants to communicate with On-Board FRMCS will priorly request the supported API version(s) by On-Board FRMCS as defined in clause 9.9. The On-Board Application will then use its selected API version among the list communicated by On-Board FRMCS as the apiVersion in the URI path of the subsequent requests. (I)

9.7 Http and SSE usage

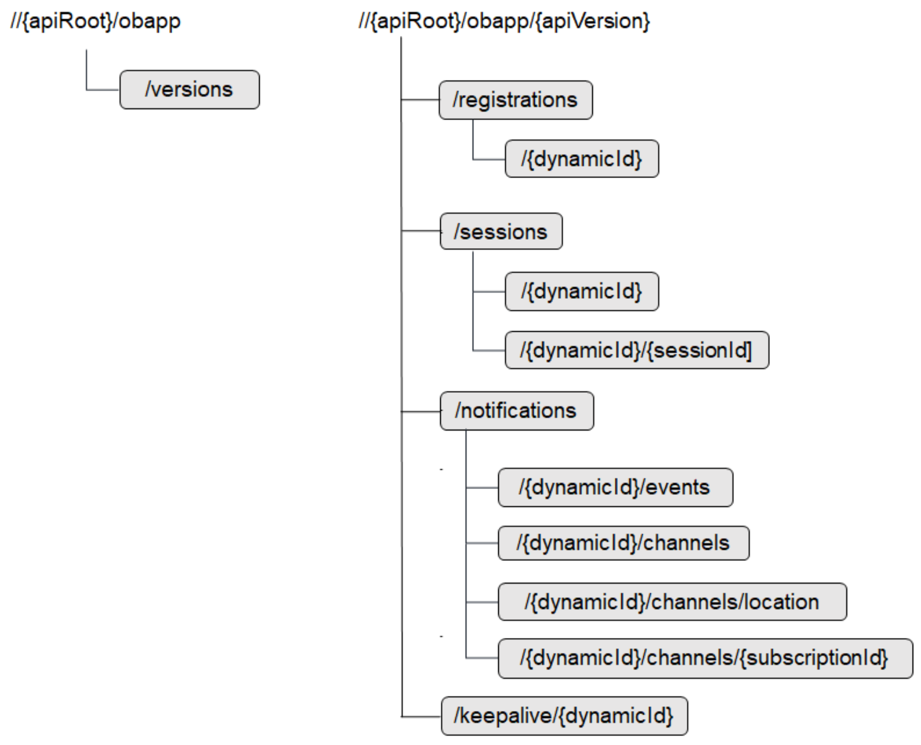
9.7.1 HTTP/2, as defined in **[RFC 9113]**, shall be used. (M)

9.7.2 The data contained in the body of HTTP request, HTTP response, and in the Data filed of SSE message shall be encoded in JSON as specified in **[RFC 8259]**. (M)

9.7.3 The use of the JSON format shall be signalled by the content type "application/json". (M)

9.8 Resource names and HTTP methods

9.8.1 Figure below describes the resource URI structure of the OB$_{APP}$ API. (I)



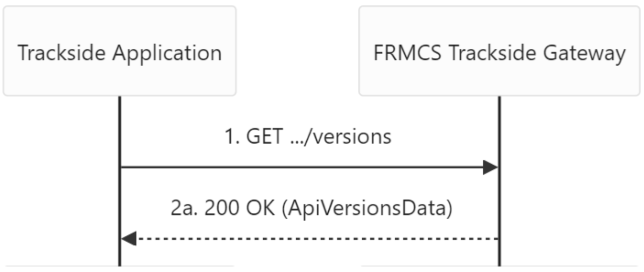
9.8.2 Table below provides an overview of the resources' names and applicable HTTP methods. (I)

Endpoint	**Method**	**Purpose**
/versions	GET	Obtain supported API versions by the On-Board FRMCS
/registrations	POST	Register an application
/registrations/{dynamicId}	DELETE	De-register an application
/sessions/{dynamicId}	GET	List of sessions for an application
	POST	Create a session for an application
/sessions/{dynamicId}/{sessionId}	GET	Get information on a session of an application
	PUT	Accept an incoming session for an application
	DELETE	Terminate a session for an application
/notifications/{dynamicId}/events	GET	Subscribe to the event stream to receive notifications
/notifications/{dynamicId}/channels	GET	Obtain list of notifications to which application has subscriptions
	DELETE	Unsubscribe all the notification channels (except the default notifications linked to the event stream)
/notifications/{dynamicId}/channels/location	POST	Subscribe to the location reporting channel
	DELETE	Unsubscribe from a specific channel for an application

/notifications/{dynamicId}/channels/{subscriptionId}	DELETE	Unsubscribe from a specific notification subscription
/keepalive/{dynamicId}	GET	Request a life signal from On-Board FRMCS

9.9 API version service

9.9.1 This API service allows an On-Board Application to obtain the supported version(s) of API by On-Board FRMCS and can be invoked without local registration. (I)



9.9.2 The On-Board Application shall send a GET request to the {apiRoot}/obapp/versions endpoint. (M)

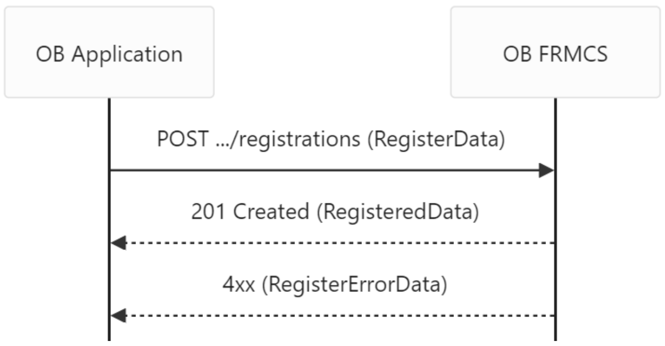
9.9.3 On success, 200” (OK) shall be returned with ApiVersionsData content as defined in Annex A. (M)

9.9.4 The On-Board Application shall utilise one of the API versions among the list of supported versions by On-Board FRMCS as the apiVersion in the API URI (see clause 9.5.1) of its subsequent requests. (M)

9.10 Local registration services

9.10.1 Register an On-Board Application

9.10.1.1 This API service allows an On-Board Application to register to the On-Board FRMCS and to obtain a unique identity (i.e., dynamicId) to be used in the API URI path of the subsequent requests. (I)


OB Application
OB FRMCS
POST .../registrations (RegisterData)
201 Created (RegisteredData)
4xx (RegisterErrorData)

9.10.1.2 The On-Board Application shall send a POST request to the /registrations endpoint. (M)

9.10.1.3 The On-Board Application shall send RegisterData content as defined in Annex A in the POST request.(M)

9.10.1.4 On success, 201 (Created) shall be returned, with Location header set to the URI of the registered application instance. (M)

9.10.1.5 The 201 (Created) response shall contain RegisteredData structure as defined in Annex A. (M)

9.10.1.6 On failure, one of the HTTP status codes listed in Table 9-3 shall be returned. (M)

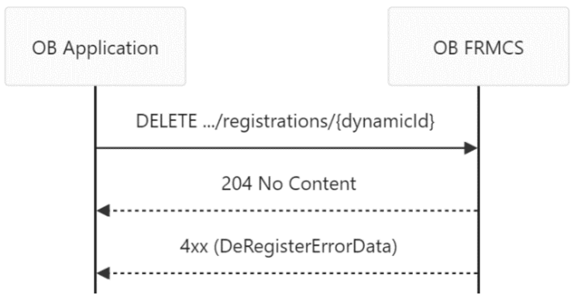
9.10.1.7 For a 4xx, the message body shall contain a RegisterErrorData structure as defined in Annex A. (M)

9.10.1.8 In the RegisterErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-3. (M)

Data type	P	Response codes	Description
RegisteredData	M	201 Created	Successful.
RegisterErrorData	M	400 Bad Request	The “cause” attribute shall be set to the following: - ILL_FORMED_REQUEST The “detail” attribute can provide more details in a human-readable format.
RegisterErrorData	M	403 Forbidden	The “cause” attribute shall be set to the following**Error! Reference source not found.**: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format.

Table 9-3.Data structures supported by the POST Response Body

9.10.2 De-Register an On-Board Application


OB Application
OB FRMCS
DELETE .../registrations/{dynamicId}
204 No Content
4xx (DeRegisterErrorData)

9.10.2.1 The On-Board Application shall send a DELETE request to the /registrations/{dynamicId} endpoint. (M)

9.10.2.2 On success, 204 (No Content) shall be returned. (M)

9.10.2.3 On failure, one of the HTTP status code listed in Table 9-4 shall be returned. (M)

9.10.2.4 For a 4xx, the message body shall contain a DeRegisterErrorData structure as defined in Annex A. (M)

9.10.2.5 In the DeRegisterErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-4. (M)

Data type	P	Response codes	Description
N/A	M	204 No Content	Successful.
DeRegisterErrorData	M	401 Unauthorized	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
DeRegisterErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - NOT_FOUND The “detail” attribute can provide more details in a human-readable format**Error! Reference source not found.**.

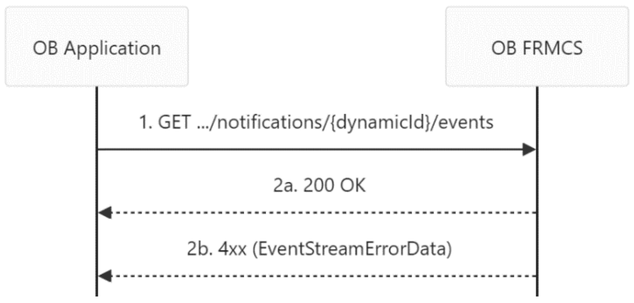
Table 9-4. Data structures supported by the DELETE Response Body

9.11 Notification services

9.11.1 Opening the notification event stream for an application

9.11.1.1 As a part of local binding an application subscribes to a notification event stream as defined in this clause. This notification event stream receives general notifications as well as the notifications from the notification channels to which the application is explicitly subscribed (e.g., clause 9.11.3). (I)

9.11.1.2 The subscription to a notification event stream implicitly includes the subscription to a set of general notifications of the following types: OpenSessionFinalAnswerNotif (see clause 9.11.1.9), IncomingSessionNotif (see clause 9.11.1.10), FtdAvlNotif (see clause 9.11.1.11), FsdAvlNotif (see clause 9.11.1.12), SessionClosureNotif (see clause 9.11.1.13) and UpcomingDeregistrationNotif (see clause 9.11.1.14). (I)


OB Application
OB FRMCS
1. GET .../notifications/{dynamicId}/events
2a. 200 OK
2b. 4xx (EventStreamErrorData)

9.11.1.3 The On-Board Application shall send a GET request to the /notifications/{dynamicId}/events endpoint. The following headers shall be set: (M)

- accept: text/event-stream

- cache-control: no-cache

9.11.1.4 On success, 200 (OK) shall be returned, the following headers shall be set: (M)

- content-type: text/event-stream

9.11.1.5 On failure, one of the HTTP status code listed in Table 9-5 shall be returned. (M)

9.11.1.6 For a 4xx, the message body shall contain a EventStreamErrorData structure as defined in Annex A. (M)

9.11.1.7 In the EventStreamErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-5. (M)

Data type	P	Response codes	Description
N/A	M	200 OK	Successful.
EventStreamErrorData	M	401 Unauthorised	The “cause” attribute shall be set tothe followings: **Error! Reference source not found.** - UNREGISTERED The “detail” attribute can provide more details in a human-readable format .
EventStreamErrorData	M	403 Forbidden	The “cause” attribute shall be set to the following: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format**Error! Reference source not found.**.

Table 9-5. Data structures supported by the GET Response Body

9.11.1.8 Following a successful subscription of an application to the notification event stream, the On-Board FRMCS shall send the SSE messages with the following field: (M)

- data: A JSON object of the type ObEventType as defined in Annex A.

9.11.1.9 General event type “openSessionFinalAnswerNotif”

9.11.1.9.1 The openSessionFinalAnswerNotif notifies to the application one of the 3 following statuses of E2E session: successful or failed or declined. (I)

9.11.1.9.2 If the notification concerns a successful status, the SSE message for openSessionFinalAnswerNotif event shall contain a OpenSessionFinalAnswerNotifSuccessData structure as defined in Annex A.(M)

9.11.1.9.3 If the notification concerns a declined status, The SSE message for openSessionFinalAnswerNotif event shall contain a OpenSessionFinalAnswerNotifDeclinedData structure as defined in Annex A and with cause set to one of the values in the corresponding row in Table 9-6. (M)

9.11.1.9.4 If the notification concerns a failed status, The SSE message for openSessionFinalAnswerNotif event shall contain a OpenSessionFinalAnswerNotifFailedData structure as defined in Annex A and with cause set to one of the values in the corresponding row in Table 9-6. (M)

Data type	P	Description
OpenSessionFinalAnswerNotif SuccessData	M	The E2E session is successful.
OpenSessionFinalAnswerNotif FailedData	M	The E2E session is failed.. The “cause” attribute shall be set to one of the followings: - MCX_ENDPOINT_NOT_REACHABLE - TERMINATING_APPLICATION_ENDPOINT_NOT_REACHABLE - TERMINATING_APPLICATION_NOT_ALLOWED The “detail” attribute can provide more details in a human-readable format.
OpenSessionFinalAnswerNotif DeclinedData	M	The E2E session is declined. The “cause” attribute shall be set to the following: - REMOTE_ENDPOINT_DECLINED The “detail” attribute can provide more details in a human-readable format.

Table 9-6. Data structures for the SSE message OpenSessionFinalAnswerNotif

9.11.1.10 General event type “incomingSessionNotif”

9.11.1.10.1 The SSE message for incomingSessionNotif event shall contain a IncomingSessionNotifData structure as defined in Annex A. (M)

9.11.1.11 General event type “ftdAvlNotif”

9.11.1.11.1 The SSE message for ftdAvlNotif event shall contain a FtdAvlNotifData structure as defined in Annex A. (M)

9.11.1.12 General event type “fsdAvlNotif”

9.11.1.12.1 The SSE message for fsdAvlNotif event shall contain a FsdAvlNotifData structure as defined in Annex A. (M)

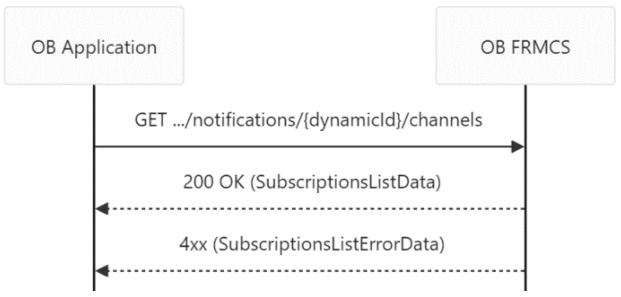
9.11.1.13 General event type “sessionClosureNotif”

9.11.1.13.1 The SSE message for sessionClosureNotif event shall contain a SessionClosureNotifData structure as defined in Annex A. (M)

9.11.1.14 General event type “upcomingDeregistrationNotif”

9.11.1.14.1 The SSE message for upcomingDeregistrationNotif event shall contain a UpcomingDeregistrationNotifData structure as defined in Annex A. (M)

9.11.2 Get information on subscriptions to notification for an application


OB Application
OB FRMCS
GET .../notifications/{dynamicId}/channels
200 OK (SubscriptionsListData)
4xx (SubscriptionsListErrorData)

9.11.2.1 The On-Board Application shall send a GET request to the /notifications/{dynamicId}/channels endpoint. (M)

9.11.2.2 On success, 200 (OK) shall be returned containing the SubscriptionsListData as defined in Annex A. (M)

9.11.2.3 On failure, one of the HTTP status codes listed in Table 9-7 shall be returned. (M)

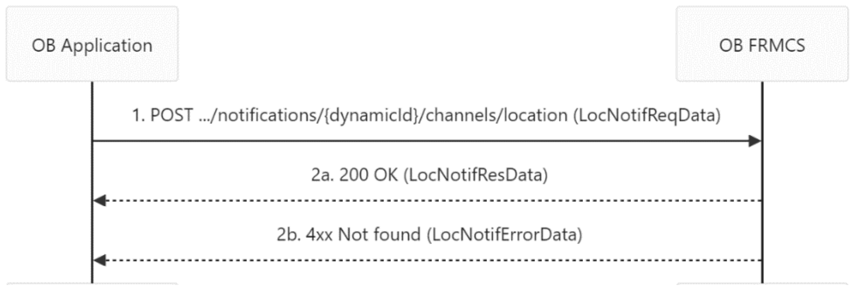
9.11.2.4 For a 4xx, the message body shall contain a SubscriptionsListErrorData structure, as defined in Annex A. (M)

9.11.2.5 In the SubscriptionsListErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-7. (M)

Data type	P	Response codes	Description
SubscriptionsListData	M	200 OK	Successful.
SubscriptionsListErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
SubscriptionsListErrorData	M	403 Forbidden	The “cause” attribute shall be set to the following: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format.
SubscriptionsListErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - NOT_FOUND The “detail” attribute can provide more details in a human-readable format.

Table 9-7. Data structures supported by the GET Response Body

9.11.3 Subscription to location reporting channel


OB Application
OB FRMCS
1. POST .../notifications/{dynamicId}/channels/location (LocNotifReqData)
2a. 200 OK (LocNotifResData)
2b. 4xx Not found (LocNotifErrorData)

9.11.3.1 The On-Board Application shall send a POST request to the /notifications/{dynamicId}/channels/location endpoint. (M)

9.11.3.2 The POST request shall contain the LocNotifReqData structure as defined on Annex A. (M):

9.11.3.3 On success, 200 (OK) shall be returned. (M)

9.11.3.4 The “200 OK” response shall contain the LocNotifResData structure as defined in Annex A. (M)

9.11.3.5 On failure, one of the HTTP status codes listed in Table 9-8 shall be returned.(M)

9.11.3.6 For a 4xx, the message body shall contain a LocNotifErrorData structure, as defined in Annex A. (M)

9.11.3.7 In the LocNotifErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-8. (M)

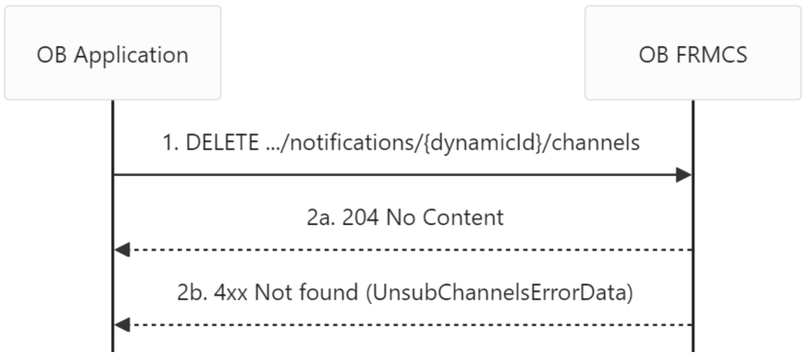
Data type	P	Response codes	Description
LocNotifResData	M	200 OK	Successful.
LocNotifErrorData	M	400 Bad Request	The “cause” attribute shall be set to the following: - ILL_FORMED_REQUEST The “detail” attribute can provide more details in a human-readable format.
LocNotifErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
LocNotifErrorData	M	403 Forbidden	The “cause” attribute shall be set to the following: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format.
LocNotifErrorData	M	501 Not Implemented	The “cause” attribute shall be set to the following: - "DISTANCE_BASED_LOC_REPORT_NOT_SUPPORTED" The “detail” attribute can provide more details in a human-readable format.

Table 9-8.Data structures supported by the POST Response Body

9.11.3.8 <intentionally deleted>

9.11.3.9 The SSE message for any of the locReportTypes shall contain a LocReportNotifData structure as defined in Annex A. (M)

9.11.4 Unsubscription from notification channels


OB Application
OB FRMCS
1. DELETE .../notifications/{dynamicId}/channels
2a. 204 No Content
2b. 4xx Not found (UnsubChannelsErrorData)

9.11.4.1 The On-Board Application shall send a DELETE request to the /notifications/{dynamicId}/channels endpoint. (M)

9.11.4.2 On success, 204 (No Content) shall be returned. (M)

9.11.4.3 On failure, one of the HTTP status code listed in Table 9-9 shall be returned. (M)

9.11.4.4 For a 4xx, the message body shall contain a UnsubChannelsErrorData structure, as defined in Annex A. (M)

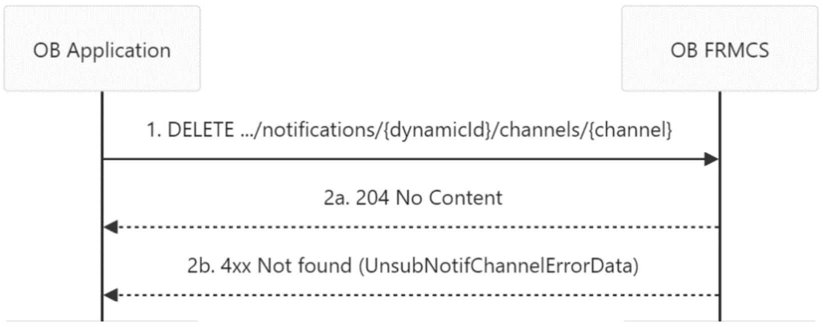
9.11.4.5 In the UnsubChannelsErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-9. (M)

Data type	P	Response codes	Description
N/A	M	204 No Content	Successful.
UnsubChannelsErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
UnsubChannelsErrorData	M	403 Not Found	The “cause” attribute shall be set to the following: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format.
UnsubChannelsErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - NOT_FOUND The “detail” attribute can provide more details in a human-readable format.

Table 9-9.Data structures supported by the DELETE Response Body

9.11.5 Unsubscription from a specific notification channel

9.11.5.0 This clause specifies how an application can unsubcribe from a notification channel. The list of notification channels from which the application can unsubscribe is provided by Notifchannel data type in Annex A. (I)


OB Application
OB FRMCS
1. DELETE .../notifications/{dynamicId}/channels/{channel}
2a. 204 No Content
2b. 4xx Not found (UnsubNotifChannelErrorData)

9.11.5.1 The On-Board Application shall send a DELETE request to the /notifications/{dynamicId}/channels/{channel} endpoint. (M)

9.11.5.2 On success, 204 (No Content) shall be returned. (M)

9.11.5.3 On failure, one of the HTTP status code listed in Table 9-10 shall be returned. (M)

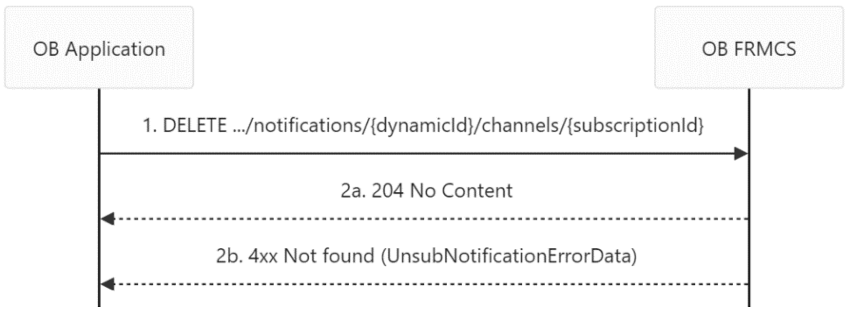
9.11.5.4 For a 4xx, the message body shall contain a UnsubNotifChannelErrorData structure, as defined in Annex A. (M)

9.11.5.5 In the UnsubNotifChannelErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-10. (M)

Data type	P	Response codes	Description
N/A	M	204 No Content	Successful.
UnsubNotifChannel ErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
UnsubNotifChannel ErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - UNKNOWN_NOTIF_CHANNEL The “detail” attribute can provide more details in a human-readable format.

Table 9-10. Data structures supported by the DELETE Response Body

9.11.6 Unsubscription from a specific notification using subscription identity


OB Application
OB FRMCS
1. DELETE .../notifications/{dynamicId}/channels/{subscriptionId}
2a. 204 No Content
2b. 4xx Not found (UnsubNotificationErrorData)

9.11.6.1 The On-Board Application shall send a DELETE request to the /notifications/{dynamicId}/channels/{subscriptionId} endpoint. (M)

9.11.6.2 On success, 204 (No Content) shall be returned. (M)

9.11.6.3 On failure, one of the HTTP status code listed in Table 9-11 shall be returned. (M)

9.11.6.4 For a 4xx, the message body shall contain a UnsubNotificationErrorData structure as defined in Annex A. (M)

9.11.6.5 In the UnsubNotificationErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-11. (M)

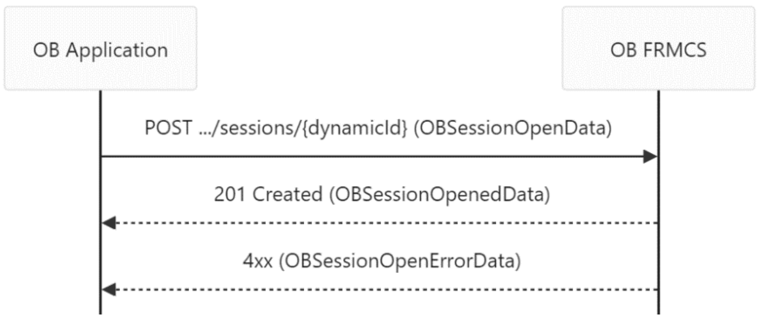
Data type	**P**	**Response codes**	**Description**
N/A	M	204 No Content	Successful.
UnsubNotificationErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
UnsubNotificationErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - UNKNOWN_SUBSCRIPTION_ID The “detail” attribute can provide more details in a human-readable format.

Table 9-11. Data structures supported by the DELETE Response Body

9.12 Session services

9.12.1 Opening a session for an application

9.12.1.1 This section describes how to initiate a session for an application on OBapp. (I)


OB Application
OB FRMCS
POST .../sessions/{dynamicId} (OBSessionOpenData)
201 Created (OBSessionOpenedData)
4xx (OBSessionOpenErrorData)

9.12.1.2 The On-Board Application shall send a POST request to the /sessions/{dynamicId} endpoint. (M)

9.12.1.3 The On-Board Application shall send OBSessionOpenData content as defined in Annex A in the POST request. (M)

Editor's Note: The combination of values provided by the Application for two attributes “appCategory” and “communicationCategory” is mapped to the corresponding FRMCS Railway On-Board Profile (FROP) as defined in [FRMCS SRS] section 19. For applications in the scope of [FRMCS FIS], this mapping between API attributes and FROP will be specified in [FRMCS FIS].

9.12.1.4 On success, 201 (Session Created) shall be returned. (M)

9.12.1.5 The 201 (Session Created) response shall contain OBSessionOpenedData structure as defined in Annex A. (M)

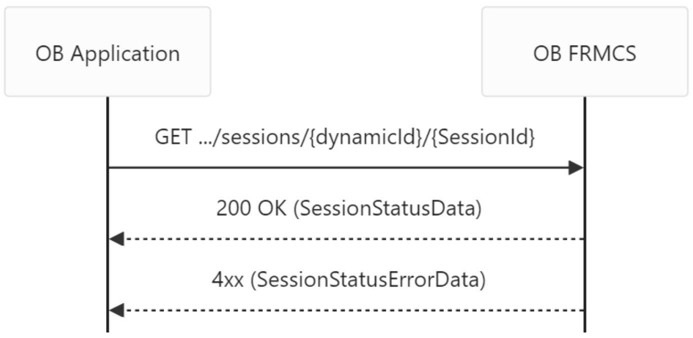
9.12.1.6 For a 4xx, the message body shall contain a OBSessionOpenErrorData structure as defined in Annex A. (M)

9.12.1.7 In the OBSessionOpenErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-12. (M)

Data type	P	Response codes	Description
OBSessionOpenedData	M	201 Session Created	Successful.
OBSessionOpenErrorData	M	400 Bad Request	The “cause” attribute shall be set to the following: - ILL_FORMED_REQUEST The “detail” attribute can provide more details in a human-readable format.
OBSessionOpenErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
OBSessionOpenErrorData	M	403 Forbidden	The “cause” attribute shall be set to the following: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format.

Table 9-12.Data structures supported by the POST Response Body

9.12.2 Get a session status


OB Application
OB FRMCS
GET .../sessions/{dynamicId}/{SessionId}
200 OK (SessionStatusData)
4xx (SessionStatusErrorData)

9.12.2.1 The On-Board Application shall send a Get request to the /sessions/{dynamicId}/{sessionId} endpoint. (M)

9.12.2.2 On success, 200 (OK) shall be returned containing SessionStatusData structure as defined in Annex A. (M)

9.12.2.3 On failure, one of the HTTP status codes listed in Table 9-13 shall be returned. (M)

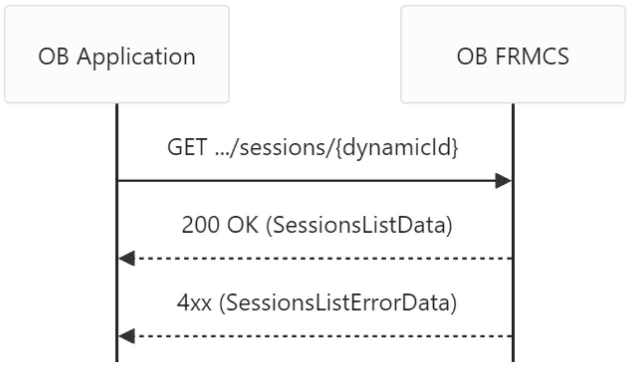
9.12.2.4 For a 4xx, the message body shall contain a SessionStatusErrorData structure, as defined in Annex A. (M)

9.12.2.5 In the SessionStatusErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-13. (M)

Data type	P	Response codes	Description
SessionStatusData	M	200 OK	Successful.
SessionStatusErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
SessionStatusErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - UNKNOWN_SESSION The “detail” attribute can provide more details in a human-readable format.

Table 9-13.Data structures supported by the GET Response Body

9.12.3 Get list of sessions for an application


OB Application
OB FRMCS
GET .../sessions/{dynamicId}
200 OK (SessionsListData)
4xx (SessionsListErrorData)

9.12.3.1 The On-Board Application shall send a Get request to the /sessions/{dynamicId} endpoint. (M)

9.12.3.2 On success, 200 (OK) shall be returned containing the SessionsListData as defined in Annex A. (M)

9.12.3.3 On failure, one of the HTTP status codes listed in Table 9-14 shall be returned. (M)

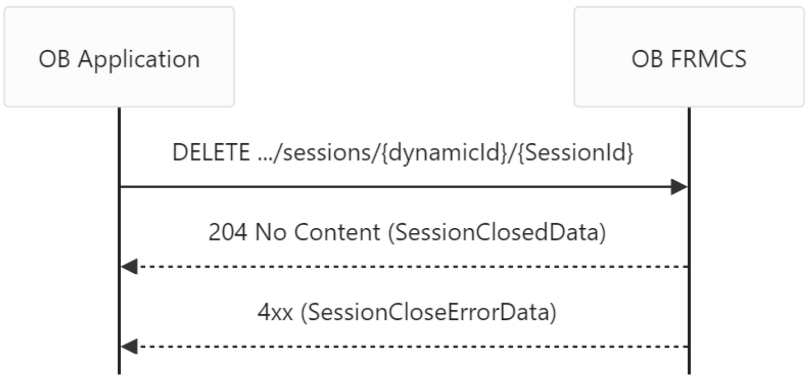
9.12.3.4 For a 4xx, the message body shall contain a SessionsListErrorData structure, as defined in Annex A. (M)

9.12.3.5 In the SessionsListErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-14. (M)

Data type	P	Response codes	Description
SessionsListData	M	200 OK	Successful.
SessionsListErrorData	M	401 Unauthorised	The "cause" attribute shall be set to the following: - UNREGISTERED The "detail" attribute can provide more details in a human-readable format.
SessionsListErrorData	M	404 Not Found	The "cause" attribute shall be set to the following: - NOT_FOUND The "detail" attribute can provide more details in a human-readable format.

Table 9-14. Data structures supported by the GET Response Body

9.12.4 Closures of a session


OB Application
OB FRMCS
DELETE .../sessions/{dynamicId}/{SessionId}
204 No Content (SessionClosedData)
4xx (SessionCloseErrorData)

9.12.4.1 The On-Board Application shall send a DELETE request to the /sessions/{dynamicId}/{sessionId} endpoint. (M)

9.12.4.2 On success, 204 (No Content) shall be returned with the SessionClosedData structure as defined in Annex A. (M)

9.12.4.3 On failure, one of the HTTP status codes listed in Table 9-15 shall be returned. (M)

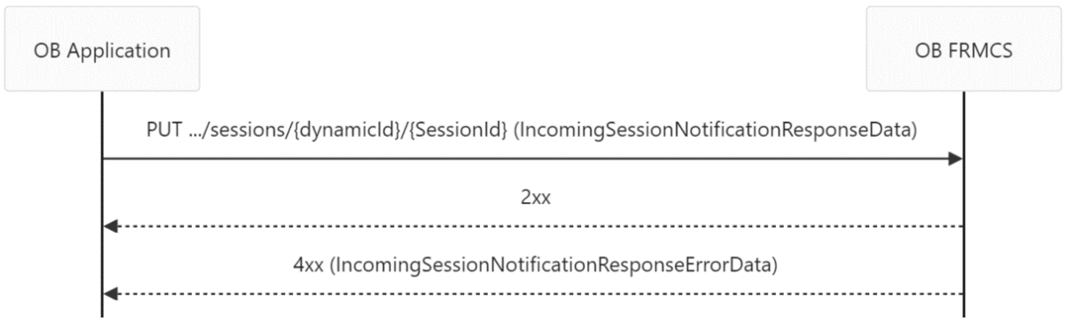
9.12.4.4 For a 4xx, the message body shall contain a SessionCloseErrorData structure, as defined in Annex A. (M)

9.12.4.5 In the SessionCloseErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-15. (M)

Data type	P	Response codes	Description
SessionCloseData	M	204 No Content	Successful.
SessionCloseErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
SessionCloseErrorData	M	403 Forbidden	The “cause” attribute shall be set to the following: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format.
SessionCloseErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - UNKNOWN_SESSION The “detail” attribute can provide more details in a human-readable format.

Table 9-15.Data structures supported by the DELETE Response Body

9.12.5 Accept an incoming session


OB Application
OB FRMCS
PUT .../sessions/{dynamicId}/{SessionId} (IncomingSessionNotificationResponseData)
2xx
4xx (IncomingSessionNotificationResponseErrorData)

9.12.5.1 The On-Board Application shall send a PUT request to the /sessions/{dynamicId}/{sessionId} endpoint containing IncomingSessionNotificationResponseData structure as defined in Annex A. (M)

9.12.5.2 On success, one of the status codes 2xx listed in Table 9-16 shall be returned. (M)

9.12.5.3 <Intentionally Deleted>

9.12.5.4 On failure, one of the HTTP status codes listed in Table 9-16 shall be returned.

9.12.5.5 For a 4xx, the message body shall contain a IncomingSessionNotificationResponseErrorData structure, as defined in Annex A. (M)

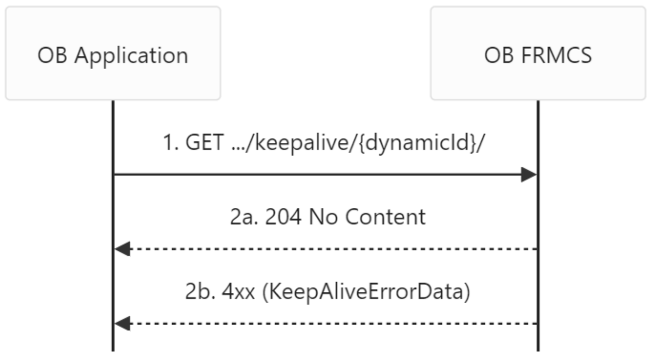
9.12.5.6 In the IncomingSessionNotificationResponseErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-16. (M)

Data type	P	Response codes	Description
N/A	M	201 Created	Sent if the application has accepted the incoming session request.
N/A	M	204 No Content	Acknowledgement of the application having declined the incoming session request.
IncomingSessionNotificationResponseErrorData	M	400 Bad Request	The "cause" attribute shall be set to the following: - ILL_FORMED_REQUEST The "detail" attribute shall be set to the corresponding description defined in.
IncomingSessionNotificationResponseErrorData	M	401 Unauthorised	The "cause" attribute shall be set to the following: - UNREGISTERED The "detail" attribute can provide more details in a human-readable format.
IncomingSessionNotificationResponseErrorData	M	404 Not Found	The "cause" attribute shall be set to the following: - UNKNOWN_SESSION The "detail" attribute can provide more details in a human-readable format.

Table 9-16.Data structures and response codes supported by the PUT Response Body

9.13 Keep alive service

9.13.1 This API service allows an On-Board Application to get a life signal from On-Board FRMCS. (I)


OB Application
OB FRMCS
1. GET .../keepalive/{dynamicId}/
2a. 204 No Content
2b. 4xx (KeepAliveErrorData)

9.13.2 The On-Board Application shall send a GET request to the /keepalive/{dynamicId}/ endpoint. (M)

9.13.3 On success, 204 (No Content) shall be returned.

9.13.4 On failure, one of the HTTP status codes listed in Table 9-17 shall be returned. (M)

9.13.5 For a 4xx, the message body shall contain a KeepAliveErrorData structure, as defined in Annex A. (M)

9.13.6 In the KeepAliveErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 9-17. (M)

Data type	P	Response codes	Description
NA	M	204	No Content.
KeepAliveErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute shall be set to the corresponding description defined in.

Table 9-17. Data structures supported by the GET Response Body

9.14 <Intentionally Deleted>

9.15 API support by On-Board FRMCS and On-Board Applications

9.15.1 [FRMCS SRS] section 6.1.3.1 defines for which application the OBapp is applied. (I)

9.15.2 The On-Board FRMCS shall implement and expose all API services provided for OBapp. (M)

9.15.3 The following API services shall be implemented by all On-Board Applications: (M)

- GET /versions
- POST /registrations
- GET /notifications/{dynamicId}/events

9.15.4 The following API services shall be implemented by all On-Board Loose Coupled (LC) applications requiring OB-originated communications: (M)

- POST /sessions/{dynamicId}
- DELETE /sessions/{dynamicId}/{sessionId}

9.15.5 The following API services shall be implemented by all On-Board Loose Coupled (LC) applications requiring OB-terminated communications: (M)

- PUT /sessions/{dynamicId}}/{sessionId}
- DELETE /sessions/{dynamicId}/{sessionId}

9.15.6 The *Table 9-18* represents the mandatory part of API services for applications by “x” (I):

Endpoint	**Method**	**Purpose**	**All**	**LC (OB-originated)**	**LC (OB-terminated)**
/versions	GET	Obtain supported API versions by the On-Board FRMCS	X		
/registrations	POST	Register an application	X		
/registrations/{dynamicId}	DELETE	De-register an application			
/sessions/{dynamicId}	GET	List of sessions for an application			
	POST	Create a session for an application		X	
/sessions/{dynamicId}/{sessionId}	GET	Get information on a session of an application			
	PUT	Accept an incoming session for an application			X
	DELETE	Terminate a session for an application		X	X
/notifications/{dynamicId}/events	GET	Subscribe to the event stream to receive notifications	X		
/ notifications/{dynamicId}/channels	GET	Obtain list of notifications to which application has subscriptions			
	DELETE	Unsubscribe all the notification channels (except general notifications linked to the event stream)			

/notifications/{dynamicId}/channels/location	POST	Subscribe to the location reporting channel			
	DELETE	Unsubscribe from a specific channel for an application			
/notifications/{dynamicId}/channels/{subscriptionId}	DELETE	Unsubscribe from a specific notification subscription			
/keepalive/{dynamicId}/	GET	Request a life signal from On-Board FRMCS			

Table 9-18.Mandatory part of API services for different categories of application

9.16 <Intentionally Deleted>

9.17 <Intentionally Deleted>

10 TS_{APP} API Services

10.1 Overview of TS_{APP} API features

TS_{APP} enables the following services between an application and the FRMCS Trackside Gateway:

10.1.1 **API version**: This TS_{APP} service is used by Trackside Application to obtain the list of API version(s) supported by the FRMCS Trackside Gateway. (I)

10.1.2 **Local registration service**: This TS_{APP} service is used to perform the Local registration between a Trackside Application and the FRMCS Trackside Gateway. (I)

10.1.3 **Local deregistration service**: This TS_{APP} service is used to request a local de-registration of the Trackside Application from the FRMCS Trackside Gateway. (I)

10.1.4 **Session opening service**: This TS_{APP} service is used to establish a session between a Trackside Application and a remote (Trackside or On-Board) application at the initiative of the Trackside Application. (I)

10.1.5 **Incoming session acceptance service**: This TS_{APP} service is used as a part of the establishment of a session between a Trackside Application and a remote (Trackside or On-Board) application at the initiative of the remote application. (I)

10.1.6 **Session closure service**: This TS_{APP} service is used to close a session between a Trackside Application and a remote application. (I)

10.1.7 **Session status service**: This TS_{APP} service is used to provide the status of a session involving the Trackside Application (I)

10.1.8 **Subscription to notification event stream service:** This feature is used to request the opening of an event stream enabling FRMCS Trackside Gateway to send notifications to the Trackside Application after the local registration. This is done during the local binding. (I)

10.1.9 **General notification service:** upon Trackside Application's subscription to notification event stream, the Trackside Application receive a set of general notifications which are linked to the following events: (I)

- Incoming session request: The FRMCS Trackside Gateway notifies the Trackside Application of the reception of an incoming (Trackside terminated) session request.
- Final answer of a session initiation: the FRMCS Trackside Gateway notifies the Trackside application of whether the establishment of the E2E communication session at the initiative of the Trackside Application was successful / failed / declined.
- Availability of FRMCS Service Domain (FSD): the FRMCS Trackside Gateway notifies the Trackside Application of the availability of FRMCS Service Domain.
- Session closure notification: The FRMCS Trackside Gateway notifies the Trackside Application of the closure of an open session over TS_{APP}.

- Upcoming deregistration notification: The FRMCS Trackside Gateway notifies the Trackside Application of an imminent deregistration of FRMCS Trackside Gateway (e.g., as a preparation for a train turnoff).

Note: These notifications do not require an explicit subscription from application and are implicitly included in the subscription to the notification event stream.

10.1.10 **<Intentionally Deleted>**

Note: in the context of the API, the term application refers to the application instance, which is a concrete running software occurrence of an application of a specific type.

10.1.11 The completion of Local Binding shall imply the successful execution of the following steps: (M)

i. The first step in which an application and the FRMCS Trackside Gateway shall mutually authenticate using TLS, which is not part of the API. See section 6.6 for more details;

ii. And the second step in which an application, through API local registration service, requests a registration to the FRMCS Trackside Gateway. The success response from FRMCS Trackside Gateway completes this step;

iii. And the third step in which the Trackside Application subscribes to the notification event stream. The success response from FRMCS Trackside Gateway completes this step.

10.1.12 The invocation of any API services beside API version, local registration, and subscription to the notification event stream is conditioned on the successful execution of the Local Binding steps. (I)

10.1.13 Mandatory TS_{APP} API services for different types of applications is covered in section 10.13. (I)

10.2 Definition of the parameters used in the API services

10.2.1 A comprehensive description of attributes and some basic data types used for TS_{APP} API services are provided in informative tables, Table 10-1 and Table 10-2, respectively. The data types are formally defined in ASN.1 format in the normative Annex B. (I)

#	Attribute name	Description
1	appCategory	Provides the category of the Application. The value is taken out of an enumerated list of application categories which includes at least standardised categories (e.g., etcs, ato, vas, tcms) defined in this document. This enumerated list can be extended with other (non-standardised) application categories per railway infrastructure manager's or railway undertaking's use, This parameter is a local attribute which can be used by FRMCS to decide how an application can be served.
2	staticId	Unique identifier of the Application within the scope of an FRMCS Trackside Gateway.
3	dynamicId	Identifier of the application instance dynamically assigned at the FRMCS Trackside Gateway, unique in the scope of the FRMCS Trackside Gateway. The format shall be a Universally Unique Identifier (UUID) version 4, as described in IETF **[RFC 4122]**.
4	apiVersion	The API version of TS_{APP} used by Trackside Application, as a part of API URI. The API version is in format v{MAJOR}.{MINOR}, as defined in clause 10.4. The default value is set to "v1.0".

5	supportedVersionsList	List of supported API versions of TS$_{APP}$ by FRMCS Trackside Gateway. The API version is in format v{MAJOR}.{MINOR}, as defined in clause 10.4.
6	couplingMode	The application coupling mode (i.e. Loose coupled, Tight coupled)..
7	uriResource	The resource field in the http error response.
8	cause	The machine-readable failure cause in the http failure response.
9	detail	The human-readable failure cause in the http failure response.
10	recipient	The remote recipient of a session originated by Trackside Application. This parameter is constituted of the address of the remote recipient.
11	sessionId	Identifier of the session, unique in the scope of the FRMCS Trackside Gateway per dynamicId. The format shall be a Universally Unique Identifier (UUID) version 4, as described in IETF **[RFC 4122]**.
12	remoteId	Remote identifier of an application in the scope of session exchange messages.
13	sessionStatus	The status of a session indicating one of the following three cases: a) the E2E session is succeeded, b) the E2E session is failed, c) the E2E session is declined.
14	nextHopIPAddress	Local FRMCS Trackside Gateway IP address to be used by the Trackside Application as local next hop IP address for the User Plane data in case of successful session establishment.
15	destApplicationIPAddress	The FRMCS IP address of destination application endpoint to be used by the Trackside Application as destination address for the User Plane data in case of successful session establishment.
16	communicationCategory	This parameter reflects the different categories of communication (session) that can be established over the FRMCS for a given application. This parameter is used by the FRMCS Trackside Gateway to initiate an end-to-end session. Based on this parameter, the FRMCS Trackside Gateway assigns the right communication profile including the QoS level to the session.
17	localAppIPAddress	Local Application IP address to be used by the FRMCS Trackside Gateway as destination address for the User Plane data in case of successful session establishment.
18	sessionOriginator	The origin of a session A session on TS$_{APP}$ is either originated by the Trackside Application or is an incoming session which is terminated at Trackside Application.
19	incomingSessionAppResponse	Application response to an incoming session which is one the followings: a) accepted, b) rejected.

Table 10-1: Definition of the parameter types that are used in the API request / response

Type Name	Description
Uuid	The format shall be a Universally Unique Identifier (UUID) version 4, as described in IETF **[RFC 4122]**.
Ipv4Addr	String identifying a IPv4 address formatted in the "dotted decimal" notation as defined in IETF **[RFC 1166]**. Pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])$'
Ipv6Addr	String identifying an IPv6 address formatted according to clause 4 of IETF **[RFC 5952]**. The mixed IPv4 IPv6 notation according to clause 5 of IETF **[RFC 5952]** shall not be used. Pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))$' And Pattern: '^((([^:]+:){7}([^:]+))\|((([^:]+:)*[^:]+)?::(([^:]+:)*[^:]+)?))$'
Uri	String providing an URI formatted according to IETF **[RFC 3986]**. If the URI fields intended to convey generic data (e.g., in the value part of a query parameter, or in the URI path segments) contain reserved characters, these reserved characters shall be percent-encoded as defined in clause 5.2.10.2 of **[3GPP TS 29.500]**.
ApiVersion	String in format "v{MAJOR}.{MINOR}".

Table 10-2: Basic Data Types

10.2.2 The TS$_{APP}$ interface to the FRMCS Trackside Gateway will have different versions as new features will be introduced. Supported versions are communicated over the

TS$_{APP}$. For each interface version, a change log is maintained, and changes are categorised into Major and Minor categories. (I)

10.2.3 The TS$_{APP}$ API versioning is defined in clause 10.4. (I)

10.2.3i For a TS$_{APP}$ API implemented according to the present FFFIS, the API version shall be set to v0.1. (M)

10.2.4 The dynamicId and sessionId shall be random cryptographic identities generated by FRMCS Trackside Gateway at runtime. (M)

10.2.5 The appCategory, as defined in Annex B, allows a list of both harmonized and non-harmonized applications. (I)

10.2.6 The field name for non-harmonized applications within appCategory shall include a prefix “ext.”, indicating non-harmonized extension of the application list. (M)

10.2.7 The static identifier of the application shall be unique in the scope of all FRMCS application instances within an FRMCS Trackside Gateway. The structure of FRMCS System identities that are used to set up the relevant FRMCS services and communication link(s) with other FRMCS users shall fulfil the requirements as specified in the **[FRMCS-SRS]**. (M)

10.2.8 The remote address of an application in the scope of TS$_{APP}$ session exchange messages shall fulfil the requirements as specified in the **[FRMCS-SRS]** section 11.6.5. (M)

10.3 API URI

10.3.1 The API URI of the TS$_{APP}$ APIs shall be:
{apiRoot}/<apiName>/<apiVersion>/<ResourceName>, with the following components:

- The {apiRoot} shall be set as “https://{localIdApiRoot}”. (M)
- The <apiName> shall be "tsapp". (M)
- The <apiVersion> shall be set to “{apiVersion}” (see details in clause 10.4). (M)
- The <ResourceName> shall be set as described in clause 10.6.2. (M)

10.3.2 The localIdApiRoot is of string type with value being deployment-specific, such as the IP address of the FRMCS Trackside Gateway within the train IP network or a locally resolvable FQDN (if the train is equipped with a DNS server). (I)

10.4 API version

10.4.1 API version (represented by ApiVersion data type) shall be a string with format “v{MAJOR}.{MINOR}”. (M)

10.4.2 The 1st Field (MAJOR) and the 2nd Field (MINOR) shall contain unsigned integer numbers, and they shall not contain leading zeroes. (M)

10.4.3 Given the format of API version, the version increments follow the rules defined in the following clauses. (I)

10.4.4 The 1st Field (MAJOR) shall be incremented only if the applied change is backward incompatible relative to the earlier, i.e. frozen version of the API. (M)

10.4.5 For a non-frozen API, the first backwards incompatible change(s) relative to the latest frozen version triggers incrementing the 1st Field (MAJOR), while subsequent backwards incompatible changes do not increment the value, until the API stays non-frozen. When the (MAJOR) field is incremented the (MINOR) field will be reset to 0. (I)

10.4.6 The 2nd Field (MINOR) shall be incremented only if the applied change is a backward compatible new feature relative to the earlier, i.e. frozen version of the API. (M)

10.4.7 For a non-frozen API, the first backwards compatible change(s) relative to the latest frozen version triggers incrementing the 2nd Field (MINOR), while subsequent backwards compatible changes do not increment the value, until the API stays non-frozen. (I)

10.4.8 An Trackside Application which wants to communicate with FRMCS Trackside Gateway will priorly request the supported API version(s) by FRMCS Trackside Gateway as defined in clause 10.7. The Trackside Application will then use its selected API version among the list communicated by FRMCS Trackside Gateway as the apiVersion in the URI path of the subsequent requests. (I)

10.5 Http and SSE usage

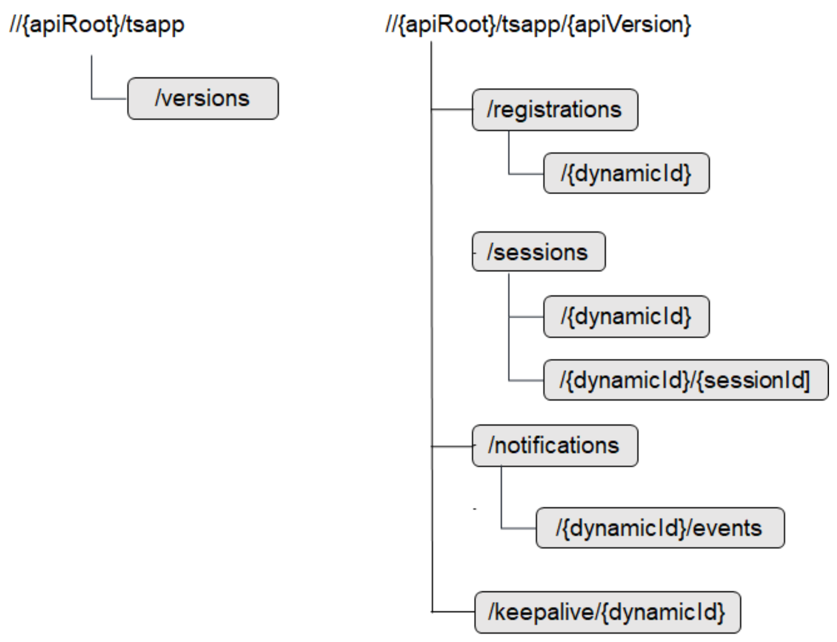
10.5.1 HTTP/2, as defined in **[RFC 9113]**, shall be used. (M)

10.5.2 The data contained in the body of HTTP request, HTTP response, and in the Data filed of SSE message shall be encoded in JSON as specified in **[RFC 8259]**. (M)

10.5.3 The use of the JSON format shall be signalled by the content type "application/json". (M)

10.6 Resource names and HTTP methods

10.6.1 Figure below describes the resource URI structure of the TS$_{APP}$ API. (I)

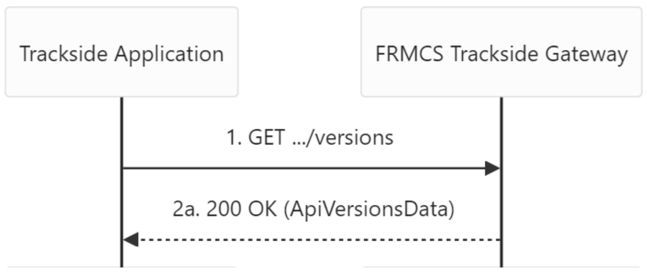


10.6.2 Table below provides an overview of the resources' names and applicable HTTP methods. (I)

Endpoint	**Method**	**Purpose**
/versions	GET	Obtain supported API versions by the FRMCS Trackside Gateway
/registrations	POST	Register an application
/registrations/{dynamicId}	DELETE	De-register an application
/sessions/{dynamicId}	GET	List of sessions for an application
	POST	Create a session for an application
/sessions/{dynamicId}/{sessionId}	GET	Get information on a session of an application
	PUT	Accept an incoming session for an application
	DELETE	Terminate a session for an application
/notifications/{dynamicId}/events	GET	Subscribe to the event stream to receive notifications
/keepalive/{dynamicId}/	GET	Request a life signal from FRMCS Trackside Gateway

10.7 API version service

10.7.1 This API service allows an Trackside Application to obtain the supported version(s) of API by FRMCS Trackside Gateway and can be invoked without local registration. (I)



10.7.2 The Trackside Application shall send a GET request to the {apiRoot}/tsapp/versions endpoint. (M)

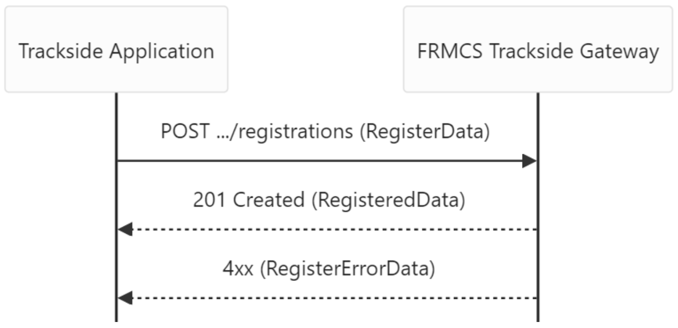
10.7.3 On success, 200 (OK) shall be returned with ApiVersionsData content as defined in Annex B. (M)

10.7.4 The Trackside Application shall utilise one of the API versions among the list of supported versions by FRMCS Trackside Gateway as the apiVersion in the API URI (see clause 10.3) of its subsequent requests. (M)

10.8 Local registration services

10.8.1 Register a Trackside Application

10.8.1.1 This API service allows an Trackside Application to register to the FRMCS Trackside Gateway and to obtain a unique identity (i.e., dynamicId) to be used in the API URI path of the subsequent requests. (I)



10.8.1.2 The Trackside Application shall send a POST request to the /registrations endpoint. (M)

10.8.1.3 The Trackside Application shall send RegisterData content as defined in Annex B in the POST request.(M)

10.8.1.4 On success, 201 (Created) shall be returned, with Location header set to the URI of the registered application instance. (M)

10.8.1.5 The 201 (Created) response shall contain RegisteredData structure as defined in Annex B. (M)

10.8.1.6 On failure, one of the HTTP status codes listed in Table 10-3 shall be returned. (M)

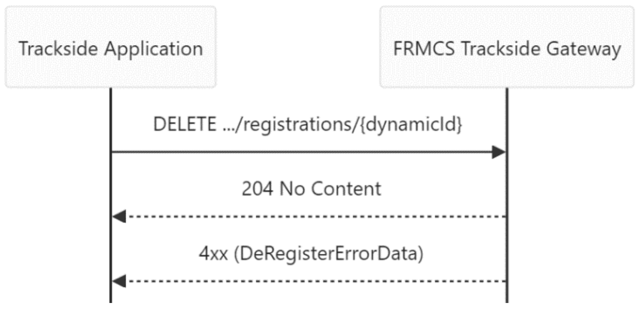
10.8.1.7 For a 4xx, the message body shall contain a RegisterErrorData structure as defined in Annex B. (M)

10.8.1.8 In the RegisterErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-3. (M)

Data type	P	Response codes	Description
RegisteredData	M	201 Created	Successful.
RegisterErrorData	M	400 Bad Request	The “cause” attribute shall be set to the following: - ILL_FORMED_REQUEST The “detail” attribute can provide more details in a human-readable format..
RegisterErrorData	M	403 Forbidden	The “cause” attribute shall be set to the following: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format.

Table 10-3. Data structures supported by the POST Response Body

10.8.2 De-Register an Trackside Application


Trackside Application
FRMCS Trackside Gateway
DELETE .../registrations/{dynamicId}
204 No Content
4xx (DeRegisterErrorData)

10.8.2.1 The Trackside Application shall send a DELETE request to the /registration/{dynamicId} endpoint. (M)

10.8.2.2 On success, 204 (No Content) shall be returned. (M)

10.8.2.3 On failure, one of the HTTP status code listed in Table 10-4 shall be returned. (M)

10.8.2.4 For a 4xx, the message body shall contain a DeRegisterErrorData structure as defined in Annex B. (M)

10.8.2.5 In the DeRegisterErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-4. (M)

Data type	P	Response codes	Description
N/A	M	204 No Content	Successful.
DeRegisterErrorData	M	401 Unauthorized	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
DeRegisterErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - NOT_FOUND The “detail” attribute can provide more details in a human-readable format.

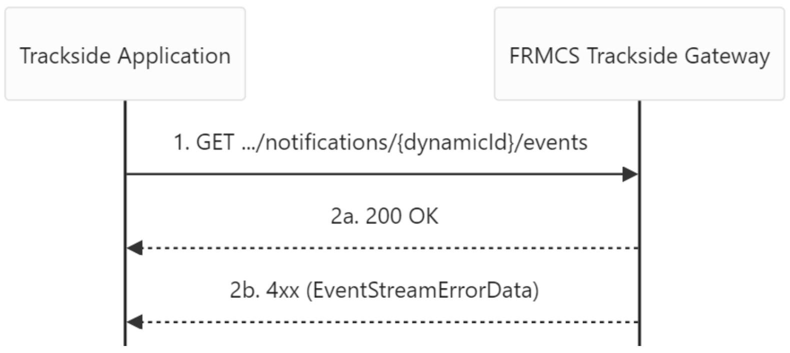
Table 10-4. Data structures supported by the DELETE Response Body

10.9 Notification services

10.9.1 Opening the notification event stream for an application

10.9.1.1 As a part of local binding an application subscribes to a notification event stream as defined in this clause. This notification event stream receives general notifications as well as the notifications from the notification channels to which the application is explicitly subscribed. (I)

10.9.1.2 The subscription to a notification event stream implicitly includes the subscription to a set of general notifications of the following types: OpenSessionFinalAnswerNotif (see clause 10.9.1.9), IncomingSessionNotif (see clause 10.9.1.10), FsdAvlNotif (see clause 10.9.1.12), SessionClosureNotif (see clause 10.9.1.13) and UpcomingDeregistrationNotif (see clause 10.9.1.14). (I)


Trackside Application
FRMCS Trackside Gateway
1. GET .../notifications/{dynamicId}/events
2a. 200 OK
2b. 4xx (EventStreamErrorData)

10.9.1.3 The Trackside Application shall send a GET request to the /notifications/{dynamicId}/events endpoint. The following headers shall be set: (M)

- accept: text/event-stream

- ccache-control: no-cache

10.9.1.4 On success, 200 (OK) shall be returned, the following headers shall be set: (M)

- content-type: text/event-stream

10.9.1.5 On failure, one of the HTTP status codes listed in Table 10-5 shall be returned. (M)

10.9.1.6 For a 4xx, the message body shall contain a EventStreamErrorData structure as defined in Annex B. (M)

10.9.1.7 In the EventStreamErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-5. (M)

Data type	P	Response codes	Description
N/A	M	200 OK	Successful.
EventStreamErrorData	M	401 Unauthorised	The "cause" attribute shall be set to the following: - UNREGISTERED The "detail" attribute shall be set to the corresponding description defined in.
EventStreamErrorData	M	403 Forbidden	The "cause" attribute shall be set to the following: - UNAUTHORIZED The "detail" attribute can provide more details in a human-readable format.

Table 10-5. Data structures supported by the GET Response Body

10.9.1.8 Following a successful subscription of an application to the notification event stream, the FRMCS Trackside Gateway shall send the SSE messages with the following field: (M)

- data: A JSON object of the type TsEventType as defined in Annex B.

10.9.1.9 General event type "openSessionFinalAnswerNotif"

10.9.1.9.1 The openSessionFinalAnswerNotif notifies to the application one of the 3 following statuses of E2E session: successful or failed or declined. (I)

10.9.1.9.2 If the notification concerns a successful status, the SSE message for openSessionFinalAnswerNotif event shall contain a OpenSessionFinalAnswerNotifSuccessData structure as defined in Annex B.(M)

10.9.1.9.3 If the notification concerns a declined status, The SSE message for openSessionFinalAnswerNotif event shall contain a OpenSessionFinalAnswerNotifDeclinedData structure as defined in Annex B and with cause set to one of the values in the corresponding row in Table 10-6. (M)

10.9.1.9.4 If the notification concerns a failed status, The SSE message for openSessionFinalAnswerNotif event shall contain a OpenSessionFinalAnswerNotifFailedData structure as defined in Annex B and with cause set to one of the values in the corresponding row in Table 10-6. (M)

Data type	P	Description
OpenSessionFinalAnswerNotif SuccessData	M	The E2E session is successful.
OpenSessionFinalAnswerNotif FailedData	M	The E2E session is failed.. The “cause” attribute shall be set to one of the followings: - MCX_ENDPOINT_NOT_REACHABLE - TERMINATING_APPLICATION_ENDPOINT_NOT_REACHABLE - TERMINATING_APPLICATION_NOT_ALLOWED The “detail” attribute can provide more details in a human-readable format.
OpenSessionFinalAnswerNotif DeclinedData	M	The E2E session is declined. The “cause” attribute shall be set to the following: - REMOTE_ENDPOINT_DECLINED The “detail” attribute can provide more details in a human-readable format.

Table 10-6. Data structures for the SSE message OpenSessionFinalAnswerNotif

10.9.1.10 General event type “incomingSessionNotif”

10.9.1.10.1 The SSE message for incomingSessionNotif event shall contain a IncomingSessionNotifData structure as defined in Annex B. (M)

10.9.1.11 <intentionally deleted>

10.9.1.12 General event type “fsdAvlNotif”

10.9.1.12.1 The SSE message for fsdAvlNotif event shall contain a FsdAvlNotifData structure as defined in Annex B. (M)

10.9.1.13 General event type “sessionClosureNotif”

10.9.1.13.1 The SSE message for sessionClosureNotif event shall contain a SessionClosureNotifData structure as defined in Annex B. (M)

10.9.1.14 General event type “upcomingDeregistrationNotif”

10.9.1.14.1 The SSE message for upcomingDeregistrationNotif event shall contain a UpcomingDeregistrationNotifData structure as defined in Annex B. (M)

10.9.2 <Intentionally Deleted>

10.9.3 <Intentionally Deleted>

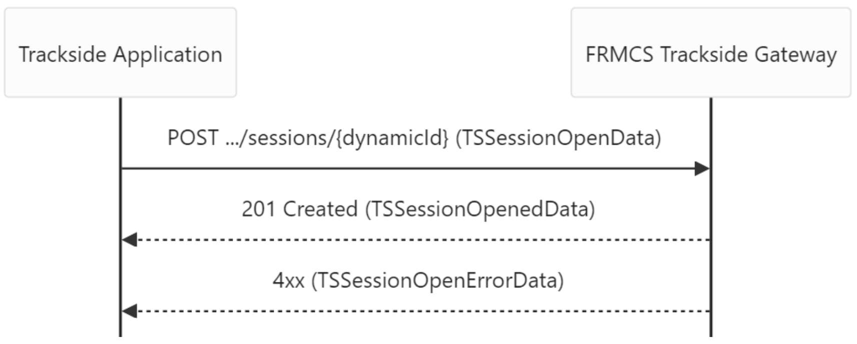
10.9.4 <Intentionally Deleted>

10.9.5 <Intentionally Deleted>

10.10 Session services

10.10.1 Opening a session for an application

10.10.1.1 This section describes how to initiate a session for an application on TS_{APP}. (I)


Trackside Application
FRMCS Trackside Gateway
POST .../sessions/{dynamicId} (TSSessionOpenData)
201 Created (TSSessionOpenedData)
4xx (TSSessionOpenErrorData)

10.10.1.2 The Trackside Application shall send a POST request to the /sessions/{dynamicId} endpoint. (M)

10.10.1.3 The Trackside Application shall send TSSessionOpenData content as defined in Annex B in the POST request. (M)

10.10.1.4 On success, 201 (Session Created) shall be returned. (M)

10.10.1.5 The 201 (Session Created) response shall contain TSSessionOpenedData structure as defined in Annex B. (M)

10.10.1.6 On failure, one of the HTTP status codes listed in Table 10-7 shall be returned. (M)

10.10.1.7 For a 4xx, the message body shall contain a TSSessionOpenErrorData structure as defined in Annex B. (M)

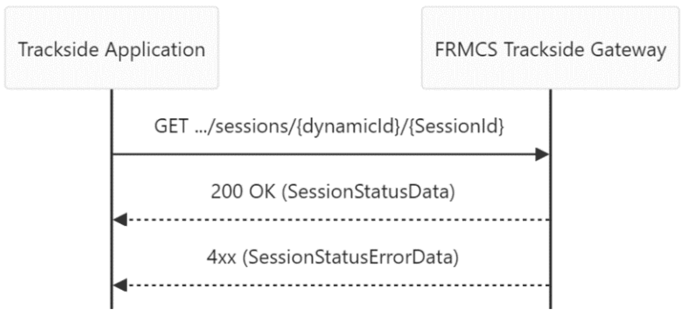
10.10.1.8 In the TSSessionOpenErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-7. (M)

Data type	P	Response codes	Description
TSSessionOpenedData	M	201 Session Created	Successful.
TSSessionOpenErrorData	M	400 Bad Request	The “cause” attribute shall be set to the following: - ILL_FORMED_REQUEST

			The “detail” attribute can provide more details in a human-readable format.
TSSessionOpenErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
TSSessionOpenErrorData	M	403 Forbidden	The “cause” attribute shall be set to the following: - UNAUTHORIZED The “detail” attribute can provide more details in a human-readable format.

Table 10-7.Data structures supported by the POST Response Body

10.10.2 Get a session status



Trackside Application
FRMCS Trackside Gateway
GET .../sessions/{dynamicId}/{SessionId}
200 OK (SessionStatusData)
4xx (SessionStatusErrorData)

10.10.2.1 The Trackside Application shall send a Get request to the /sessions/{dynamicId}/{sessionId} endpoint. (M)

10.10.2.2 On success, 200 (OK) shall be returned containing SessionStatusData structure as defined in Annex B. (M)

10.10.2.3 On failure, one of the HTTP status codes listed in Table 10-8 shall be returned. (M)

10.10.2.4 For a 4xx, the message body shall contain a SessionStatusErrorData structure, as defined in Annex B. (M)

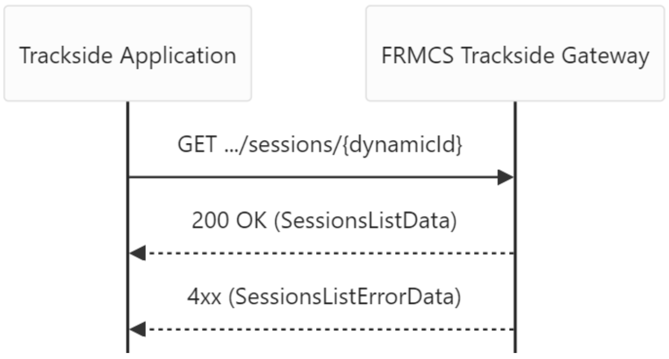
10.10.2.5 In the SessionStatusErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-8. (M)

Data type	P	Response codes	Description
SessionStatusData	M	200 OK	Successful.
SessionStatusErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.

SessionStatusError Data	M	404 Not Found	The “cause” attribute shall be set to the following: - UNKNOWN_SESSION The “detail” attribute can provide more details in a human-readable format.

Table 10-8.Data structures supported by the GET Response Body

10.10.3 Get list of sessions for an application


Trackside Application
FRMCS Trackside Gateway
GET .../sessions/{dynamicId}
200 OK (SessionsListData)
4xx (SessionsListErrorData)

10.10.3.1 The Trackside Application shall send a Get request to the /sessions/{dynamicId} endpoint. (M)

10.10.3.2 On success, 200 (OK) shall be returned containing the SessionsListData as defined in Annex B. (M)

10.10.3.3 On failure, one of the HTTP status codes listed in Table 10-9 shall be returned. (M)

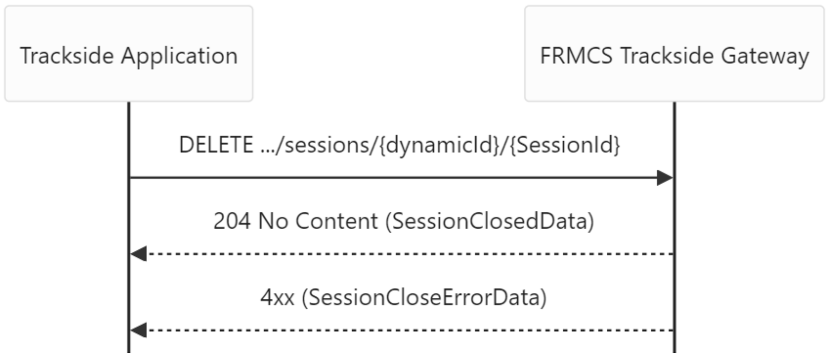
10.10.3.4 For a 4xx, the message body shall contain a SessionsListErrorData structure, as defined in Annex B. (M)

10.10.3.5 In the SessionsListErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-9. (M)

Data type	P	Response codes	Description
SessionsListData	M	200 OK	Successful.
SessionsListErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
SessionsListErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - NOT_FOUND The “detail” attribute can provide more details in a human-readable format.

Table 10-9. Data structures supported by the GET Response Body

10.10.4 Closures of a session


Trackside Application
FRMCS Trackside Gateway
DELETE .../sessions/{dynamicId}/{SessionId}
204 No Content (SessionClosedData)
4xx (SessionCloseErrorData)

10.10.4.1 The Trackside Application shall send a DELETE request to the /sessions/{dynamicId}/{sessionId} endpoint. (M)

10.10.4.2 On success, 204 (No Content) shall be returned with the SessionClosedData structure as defined in Annex B. (M)

10.10.4.3 On failure, one of the HTTP status codes listed in Table 10-10 shall be returned. (M)

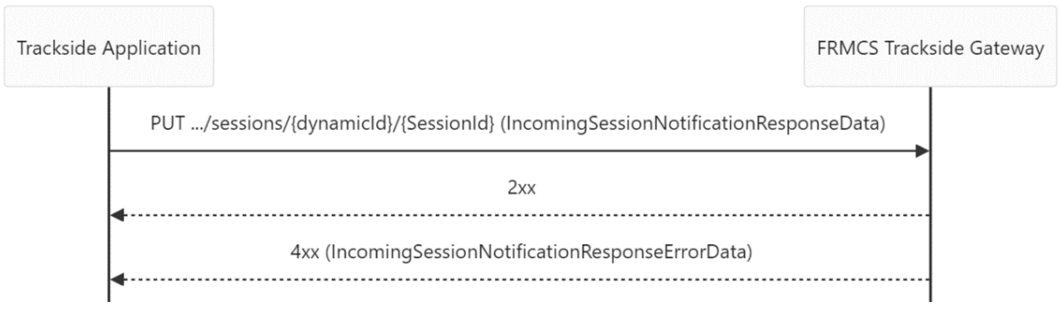
10.10.4.4 For a 4xx, the message body shall contain a SessionCloseErrorData structure, as defined in Annex B. (M)

10.10.4.5 In the SessionCloseErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-10. (M)

Data type	P	Response codes	Description
SessionCloseData	M	204 No Content	Successful.
SessionCloseErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
SessionCloseErrorData	M	404 Not Found	The “cause” attribute shall be set to the following: - UNKNOWN_SESSION The “detail” attribute can provide more details in a human-readable format.

Table 10-10.Data structures supported by the DELETE Response Body

10.10.5 Accept an incoming session


Trackside Application
FRMCS Trackside Gateway
PUT .../sessions/{dynamicId}/{SessionId} (IncomingSessionNotificationResponseData)
2xx
4xx (IncomingSessionNotificationResponseErrorData)

10.10.5.1 The Trackside Application shall send a PUT request to the /sessions/{dynamicId}/{sessionId} endpoint containing IncomingSessionNotificationResponseData structure as defined in Annex B. (M)

10.10.5.2 On success, one of the status codes 2xx listed in Table 10-11 shall be returned. (M)

10.10.5.3 <Intentionally Deleted>

10.10.5.4 On failure, one of the HTTP status codes listed in Table 10-11 shall be returned.

10.10.5.5 For a 4xx, the message body shall contain a IncomingSessionNotificationResponseErrorData structure, as defined in Annex B. (M)

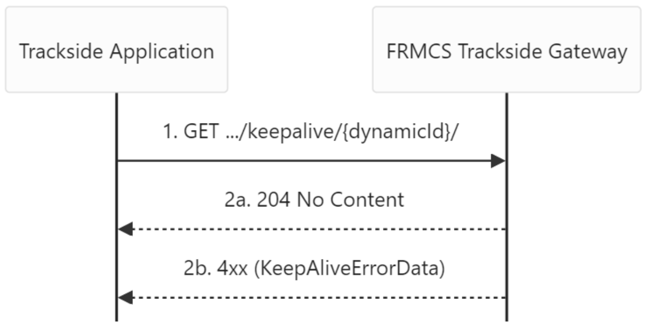
10.10.5.6 In the IncomingSessionNotificationResponseErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-11. (M)

Data type	P	Response codes	Description
N/A	M	201 Created	Sent if the application has accepted the incoming session request.
N/A	M	204 No Content	Acknowledgement of the application having declined the incoming session request.
IncomingSessionNotificationResponseErrorData	M	400 Bad Request	The “cause” attribute shall be set to the following: - ILL_FORMED_REQUEST The “detail” attribute can provide more details in a human-readable format.
IncomingSessionNotificationResponseErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.
IncomingSessionNotificationResponseErrorData		404 Not Found	The “cause” attribute shall be set to the following: - UNKNOWN_SESSION The “detail” attribute can provide more details in a human-readable format.

Table 10-11.Data structures and response codes supported by the PUT Response Body

10.11 Keep alive service

10.11.1 This API service allows an Trackside Application to get a life signal from FRMCS Trackside Gateway. (I)



10.11.2 The Trackside Application shall send a GET request to the /keepalive/{dynamicId}/ endpoint. (M)

10.11.3 On success, 204 (No Content) shall be returned.

10.11.4 On failure, one of the HTTP status codes listed in Table 10-12 shall be returned. (M)

10.11.5 For a 4xx, the message body shall contain a KeepAliveErrorData structure, as defined in Annex B. (M)

10.11.6 In the KeepAliveErrorData of HTTP failure response, the uriResource shall be set to the revoked URI resource, and cause shall be set to the values in one of the rows of Table 10-12. (M)

Data type	P	Response codes	Description
NA	M	204 OK	No Content.
KeepAliveErrorData	M	401 Unauthorised	The “cause” attribute shall be set to the following: - UNREGISTERED The “detail” attribute can provide more details in a human-readable format.

Table 10-12. Data structures supported by the GET Response Body

10.12 <Intentionally Deleted>

10.13 API support by FRMCS Trackside Gateway and Trackside Applications

10.13.1 [FRMCS SRS] section 6.1.3 defines for which application the TS$_{APP}$ is applied. (I)

10.13.2 The FRMCS TRACKSIDE GATEWAY shall implement and expose all API services provided for TS$_{APP}$. (M)

10.13.3 The following API services shall be implemented by all Trackside Applications: (M)

- GET /versions
- POST /registrations
- GET /notifications/{dynamicId}/events

10.13.4 The following API services shall be implemented by all Trackside Loose Coupled (LC) applications requiring TS-originated communications: (M)

- POST /sessions/{dynamicId}
- DELETE /sessions/{dynamicId}/{sessionId}

10.13.5 The following API services shall be implemented by all Trackside Loose Coupled (LC) applications requiring TS-terminated communications: (M)

- PUT /sessions/{dynamicId}}/{sessionId}
- DELETE /sessions/{dynamicId}/{sessionId}

10.13.6 The *Table 10-13* represents the mandatory part of API services for applications by "x" (I):

Endpoint	**Method**	**Purpose**	**All**	**LC (TS-originated)**	**LC (TS-terminated)**
/versions	GET	Obtain supported API versions by the FRMCS Trackside Gateway	X		
/registrations	POST	Register an application	X		
/registrations/{dynamicId}	DELETE	De-register an application			
/sessions/{dynamicId}	GET	List of sessions for an application			
	POST	Create a session for an application		X	
/sessions/{dynamicId}/{sessionId}	GET	Get information on a session of an application			
	PUT	Accept an incoming session for an application			X
	DELETE	Terminate a session for an application		X	X
/notifications/{dynamicId}/events	GET	Subscribe to the event stream to receive notifications	X		
/keepalive/{dynamicId}/	GET	Request a life signal from FRMCS Trackside Gateway			

Table 10-13.Mandatory part of API services for different categories of application

10.14 <Intentionally Deleted>

10.15 <Intentionally Deleted>

11 <Intentionally Deleted>

Annex A. (Normative) ASN.1 notations of OBAPP parameters

A.1 Basic Data Types

```
Uuid ::= UTF8String(PATTERN "[0-9A-F]#8-[0-9A-F]#4-[4][0-9A-F]#3-[89AB][0-9A-F]#3-[0-9A-F]#12")

ApiVersion ::= UTF8String(PATTERN "|v[0-9]#(1,2)\.[0-9]#(1,2)")

DomainName ::= UTF8String(PATTERN "[0-9]#3-[0-9]#(2,3)")

Uri ::= UTF8String(SIZE(3..256))

Ipv4Address ::= UTF8String(PATTERN "(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.)#3([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])")

Ipv6Address ::= UTF8String(PATTERN "((:|(0?|([1-9a-f][0-9a-f]#(0,3)))):)((0?|([1-9a-f][0-9a-f]#(0,3))):)#(0,6)(:|(0?|([1-9a-f][0-9a-f]#(0,3))))")


IpAddress ::= CHOICE {
  v4 Ipv4Address,
  v6 Ipv6Address
}
-- DATE-TIME is a built-in data type of ASN1 in format UTF8String(PATTERN "[0-9]#4-(0[0-9]|1[0-2])-(0[1-9]|[1-2][0-9]|3[12])T([01][0-9]|2[0-3]):[0-5][0-9]:([0-5][0-9]|60)(\.[0-9]+)?(Z|[+-]([01][0-9]|2[0-3]):[0-5][0-9])?") --
```

A.2 OBapp parameters

```
-- In later versions this any-character format after ext. can be limited to some character class, if needed.
AppCategory ::= UTF8String(PATTERN "etcs|ato|vas|tcms|ext\..*")


StaticId ::= UTF8String(SIZE(3..256))

DynamicId ::= Uuid

TimerValue ::= INTEGER(0..300)

ApiVersionList ::= SET OF ApiVersion

CouplingMode ::= ENUMERATED {
  tight,
  loose
}
```

```
-- The machine-readable failure cause in the http failure response.  --
ErrorCause ::= UTF8String(
  "ILL_FORMED_REQUEST" |
  "UNAUTH_UNKNOWN_APP_CATEGORY" |
  "UNREGISTERED" |
  "UNAUTHORIZED" |
  "NOT_FOUND" |
  "UNKNOWN_NOTIF_CHANNEL" |
  "UNKNOWN_SUBSCRIPTION_ID" |
  "MCX_ENDPOINT_NOT_REACHABLE" |
  "TERMINATING_APPLICATION_ENDPOINT_NOT_REACHABLE" |
  "TERMINATING_APPLICATION_NOT_ALLOWED" |
  "UNKNOWN_SESSION" |
  "REMOTE_ENDPOINT_DECLINED" |
  "DISTANCE_BASED_LOC_REPORT_NOT_SUPPORTED")

-- The detailed failure case description in the http failure response. --
ErrorDetail ::= UTF8String

ErrorUriResource ::= Uri

SessionId ::= Uuid


-- Remote address of an application in the scope of session exchange messages. --
RemoteId ::= Uri

SessionStatus ::= ENUMERATED {
  succeeded,
  failed,
  declined
}

NextHopIPAddress ::= IpAddress

DestApplicationIPAddress ::= IpAddress

CommunicationCategory ::= UTF8String

LocalAppIPAddress ::= IpAddress

SessionOriginator ::= ENUMERATED {
  -- On-Board application originated --
  localApplication,
  -- On-Board application incoming session --
  remoteApplication
}

-- defaulting at infinity if not present--
Period ::= INTEGER
```

```
-- defaulting at infinity if not present--
Distance ::= INTEGER

LocReportType ::= ENUMERATED {
  periodicLocRep,
  travelledDistanceLocRep,
  cellChangeLocRep
}

SubscriptionId ::= Uuid



ObEventType ::= CHOICE {
  -- The On-Board FRMCS can use this event type for notifying the application on the
successful establishment of the E2E session. --
  openSessionFinalAnswerNotif OpenSessionFinalAnswerNotifData,
  -- The On-Board FRMCS can use this event type for notifying the application on an
incoming session. --
  incomingSessionNotif IncomingSessionNotifData,
  -- The On-Board FRMCS can use this event type for notifying the availability of FRMCS
Transport Domain. --
  ftdAvlNotif FtdAvlNotifData,
  -- The On-Board FRMCS can use this event type for notifying the availability of FRMCS
Service Domain. --
  fsdAvlNotif FsdAvlNotifData,
-- The On-Board FRMCS can use this event type for reporting subscribed location updates. --
  locReportNotif LocReportNotifData,
-- The On-Board FRMCS can use this event type for requesting the application entity the
closure of an OBapp session (incoming or outgoing). --
  sessionClosureNotif SessionClosureNotifData,
-- The On-Board FRMCS can use this event type for informing the application of an
upcoming deregistration/turnoff of On-Board FRMCS. --
  upcomingDeregistrationNotif UpcomingDeregistrationNotifData
}


-- 0 if FTD is not available, 1 if FTD is available. --
FtdAVL ::= BOOLEAN

-- 0 if FSD is not available, 1 if FSD is available. --
FsdAVL ::= BOOLEAN

-- 1 if the event is due is network transition, 0 otherwise. --
NWTransition ::= BOOLEAN

FrmcsDomain ::= DomainName


Recipient ::= SET {
  remoteId RemoteId
}


IncomingSessionAppResponse ::= ENUMERATED {
  -- Incoming session is accepted by application. --
  accepted,
  -- Incoming session is rejected by application. --
  rejected
}

-- The notification channels to which an application can subscribe. --
NotifChannel ::= ENUMERATED {
    -- Location information update notification channel. --
  location
}

ServingCellId ::= UTF8String(PATTERN "^[0-9]#3-[0-9]#(2,3)\.[A-Fa-f0-9]#9")

-- The longitude as a constituent of Train Geographic 2D Position. --
Longitude ::= INTEGER(-8388608..8388607)

-- The latitude as a constituent of Train Geographic 2D Position. --
Latitude ::= INTEGER(-8388608..8388607)

-- The Accuracy of the Train Geographic 2D Position (horizontal accuracy). --
HorizontalAccuracy ::= INTEGER(0..255)

-- The speed of the train. --
Speed ::= INTEGER(0..65535)

-- The direction of the train. --
Direction ::= INTEGER(0..359)

-- The Accuracy of the speed. –
SpeedAccuracy ::= INTEGER(0..255)


GnssInformation ::= SET {
  longitude Longitude,
  latitude Latitude,
  horizontalAccuracy HorizontalAccuracy,
  speed Speed,
  direction Direction,
  speedAccuracy SpeedAccuracy
}


-- The time stamp of location report. --
TimeStamp ::= DATE-TIME
```

```
ErrorData ::= SET {
  uriResource ErrorUriResource,
  cause ErrorCause,
  detail ErrorDetail
}
```

A.3 Data structures within OBapp message body text

```
ApiVersionsData ::= SET {
  supportedVersionsList ApiVersionList
}

RegisterData ::= SET {
  appCategory AppCategory,
  staticId StaticId,
  couplingMode CouplingMode DEFAULT loose
}


RegisteredData ::= SET {
  dynamicId DynamicId
}

RegisterErrorData ::= ErrorData

DeRegisterErrorData ::= ErrorData

EventStreamErrorData::= ErrorData



--The value SubscriptionInfo is channel-specific. For the moment we only have location
channel. --
SubscriptionData::= SET {
subscriptionId SubscriptionId,
channel SubscriptionInfoData
}

SubscriptionInfoData ::= CHOICE {
  location LocNotifReqData
}

SubscriptionsListData ::= SET OF SubscriptionData

SubscriptionsListErrorData ::= ErrorData

OpenSessionFinalAnswerNotifData ::= CHOICE {
  success OpenSessionFinalAnswerNotifSuccessData,
  declined OpenSessionFinalAnswerNotifDeclinedData,
  failed OpenSessionFinalAnswerNotifFailedData
}

OpenSessionFinalAnswerNotifSuccessData ::= SET {
  sessionId SessionId,
  nextHopIPAddress IpAddress,
  destApplicationIPAddress IpAddress
}

OpenSessionFinalAnswerNotifDeclinedData ::= SET {
  sessionId SessionId,
  cause ErrorCause,
  detail ErrorDetail
}

OpenSessionFinalAnswerNotifFailedData ::= SET {
  cause ErrorCause,
  detail ErrorDetail
}

IncomingSessionNotifData ::= SET {
  remoteId RemoteId,
  communicationCategory CommunicationCategory,
  sessionId SessionId
}

FtdAvlNotifData ::= SET {
  ftdAVL FtdAVL,
  nwTransition NWTransition,
  frmcsDomain FrmcsDomain OPTIONAL -- if ftdAVL and nwTransition is TRUE --
} (WITH COMPONENTS {
  ftdAVL(FALSE),
  nwTransition(FALSE),
  frmcsDomain ABSENT
} | WITH COMPONENTS {
  ftdAVL(FALSE),
  nwTransition(TRUE),
  frmcsDomain ABSENT
} | WITH COMPONENTS {
  ftdAVL(TRUE),
  nwTransition(FALSE),
  frmcsDomain ABSENT
} | WITH COMPONENTS {
  ftdAVL(TRUE),
  nwTransition(TRUE),
  frmcsDomain PRESENT
})

FsdAvlNotifData ::= SET {
  fsdAVL FsdAVL,
  nwTransition NWTransition
```

```
}

SessionClosureNotifData ::= SET {  sessionId SessionId,
  sessionOriginator SessionOriginator
}

UpcomingDeregistrationNotifData ::= SET {
  timeToDeregistration TimerValue
}


LocNotifReqData ::= SET {
  locReportType LocReportType,
  period Period OPTIONAL , -- if locReportType is periodicLocRep --
  distance Distance OPTIONAL -- if locReportType is travelledDistanceLocRep --
} (WITH COMPONENTS {
  locReportType(periodicLocRep),
  period PRESENT,
  distance ABSENT
} | WITH COMPONENTS {
  locReportType(travelledDistanceLocRep),
  period ABSENT,
  distance PRESENT
} | WITH COMPONENTS {
  locReportType(cellChangeLocRep),
  period ABSENT,
  distance ABSENT
})


LocNotifResData ::= SET {
  locReportId SubscriptionId
}


LocNotifErrorData ::= ErrorData


LocReportNotifData ::= SET {
  subscriptionId SubscriptionId,
  servingCellId ServingCellId OPTIONAL,
  gnssInformation GnssInformation OPTIONAL,
  timeStamp TimeStamp
} (WITH COMPONENTS {
  subscriptionId,
  servingCellId PRESENT,
  gnssInformation ABSENT,
  timeStamp
} | WITH COMPONENTS {
  subscriptionId,
  servingCellId ABSENT,
  gnssInformation PRESENT,
  timeStamp
} | WITH COMPONENTS {
  subscriptionId,
  servingCellId PRESENT,
  gnssInformation PRESENT,
  timeStamp
})

UnsubNotifChannelErrorData ::= ErrorData

UnsubChannelsErrorData ::= ErrorData

UnsubNotificationErrorData::= ErrorData


OBSessionOpenData ::= SET {
  localAppIPAddress IpAddress,
  communicationCategory CommunicationCategory,
  recipient Recipient
}


OBSessionOpenedData ::= SET {
  sessionId SessionId
}


OBSessionOpenErrorData ::= ErrorData


SessionStatusData ::= SET {
  sessionOriginator SessionOriginator,
  communicationCategory CommunicationCategory,
  remoteId RemoteId,
  nextHopIPAddress IpAddress OPTIONAL, -- if sessionOriginator is localApplication--
  destApplicationIPAddress IpAddress OPTIONAL, --if sessionOriginator is
localApplication--
  localAppIPAddress IpAddress OPTIONAL -- if sessionOriginator is remoteApplication--
 } (WITH COMPONENTS {
  sessionOriginator(localApplication),
  communicationCategory,
  remoteId,
  nextHopIPAddress PRESENT,
  destApplicationIPAddress PRESENT,
  localAppIPAddress ABSENT
  } | WITH COMPONENTS {
  sessionOriginator(remoteApplication),
  communicationCategory,
  remoteId,
  nextHopIPAddress ABSENT,
  destApplicationIPAddress ABSENT,
  localAppIPAddress PRESENT
})

SessionStatusErrorData ::= ErrorData

StatusPerSessionId ::= SET {
  sessionId SessionId,
  sessionStatusData SessionStatusData
}

SessionsListData ::= SET OF StatusPerSessionId


SessionsListErrorData ::= ErrorData


SessionClosedData ::= SET {
  sessionOriginator SessionOriginator
}


SessionCloseErrorData ::= ErrorData


IncomingSessionNotificationResponseData ::= SET {
  incomingSessionAppResponse IncomingSessionAppResponse,
  localAppIPAddress IpAddress OPTIONAL -- if incomingSessionAppResponse is accepted
--
} (WITH COMPONENTS {
  incomingSessionAppResponse(accepted),
  localAppIPAddress PRESENT
} | WITH COMPONENTS {
  incomingSessionAppResponse(rejected),
  localAppIPAddress ABSENT
})


IncomingSessionNotificationResponseErrorData ::= ErrorData

KeepAliveErrorData ::= ErrorData
```

Annex B. (Normative) ASN.1 notations of TS_{APP} parameters

B.1 Basic Data Types

```
Uuid ::= UTF8String(PATTERN "[0-9A-F]#8-[0-9A-F]#4-[4][0-9A-F]#3-[89AB][0-9A-F]#3-[0-9A-F]#12")

ApiVersion ::= UTF8String(PATTERN "|v[0-9]#(1,2)\.[0-9]#(1,2)")

DomainName ::= UTF8String(PATTERN "[0-9]#3-[0-9]#(2,3)")

Uri ::= UTF8String(SIZE(3..256))

Ipv4Address ::= UTF8String(PATTERN "(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.)#3([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])")

Ipv6Address ::= UTF8String(PATTERN "((:|(0?|([1-9a-f][0-9a-f]#(0,3)))):)((0?|([1-9a-f][0-9a-f]#(0,3))):)#(0,6)(:|(0?|([1-9a-f][0-9a-f]#(0,3))))")


IpAddress ::= CHOICE {
  v4 Ipv4Address,
  v6 Ipv6Address
}
```

B.2 TS_{APP} parameters

```
--in later versions this any-character format after ext. can be limited to some character class, if needed. --
AppCategory ::= UTF8String(PATTERN "etcs|ato|vas|tcms|ext\..*")


StaticId ::= UTF8String(SIZE(3..256))

DynamicId ::= Uuid

-- Time value between 0 and 300 seconds --
TimerValue ::= INTEGER(0..300)

ApiVersionList ::= SET OF ApiVersion

CouplingMode ::= ENUMERATED {
  tight,
  loose
}

-- The machine-readable failure cause in the http failure response.  --
```

```
ErrorCause ::= UTF8String(
  "ILL_FORMED_REQUEST" |
  "UNAUTH_UNKNOWN_APP_CATEGORY" |
  "UNREGISTERED" |
  "UNAUTHORIZED" |
  "NOT_FOUND" |
  "MCX_ENDPOINT_NOT_REACHABLE" |
  "TERMINATING_APPLICATION_ENDPOINT_NOT_REACHABLE" |
  "TERMINATING_APPLICATION_NOT_ALLOWED" |
  "UNKNOWN_SESSION" |
  "REMOTE_ENDPOINT_DECLINED"
)


-- The detailed failure case description in the http failure response. --
ErrorDetail ::= UTF8String

ErrorUriResource ::= Uri

SessionId ::= Uuid


-- Remote address of an application in the scope of session exchange messages. --
RemoteId ::= Uri

SessionStatus ::= ENUMERATED {
  succeeded,
  failed,
  declined
}

NextHopIPAddress ::= IpAddress

DestApplicationIPAddress ::= IpAddress

CommunicationCategory ::= UTF8String

LocalAppIPAddress ::= IpAddress

SessionOriginator ::= ENUMERATED {
  -- Trackside application originated --
  localApplication,
  -- Trackside application incoming session --
  remoteApplication
}



TsEventType ::= CHOICE {
  -- The FRMCS Trackside Gateway can use this event type for notifying the application on
the successful establishment of the E2E session. --
```

```
  openSessionFinalAnswerNotif OpenSessionFinalAnswerNotifData,
  -- The FRMCS Trackside Gateway can use this event type for notifying the application on
an incoming session. --
  incomingSessionNotif IncomingSessionNotifData,
-- The FRMCS Trackside Gateway can use this event type for notifying the availability of
FRMCS Service Domain. --
  fsdAvlNotif FsdAvlNotifData,
-- The FRMCS Trackside Gateway can use this event type for requesting the application
entity the closure of a TSapp session (incoming or outgoing). --
  sessionClosureNotif SessionClosureNotifData,
-- The FRMCS Trackside Gateway can use this event type for informing the application of an
upcoming deregistration/turnoff of FRMCS Trackside Gateway. --
  upcomingDeregistrationNotif UpcomingDeregistrationNotifData
}

-- 0 if FSD is not available, 1 if FSD is available. --
FsdAVL ::= BOOLEAN

Recipient ::= SET {
  remoteId RemoteId

}


IncomingSessionAppResponse ::= ENUMERATED {
  -- Incoming session is accepted by application. --
  accepted,
  -- Incoming session is rejected by application. --
  rejected
}



ErrorData ::= SET {
  uriResource ErrorUriResource,
  cause ErrorCause,
  detail ErrorDetail
}
```

B.3 Data structures within TS$_{APP}$ message body text

```
ApiVersionsData ::= SET {
  supportedVersionsList ApiVersionList
}

RegisterData ::= SET {
  appCategory AppCategory,
  staticId StaticId,
  couplingMode CouplingMode DEFAULT loose
```

```
}


RegisteredData ::= SET {
  dynamicId DynamicId
}

RegisterErrorData ::= ErrorData

DeRegisterErrorData ::= ErrorData

EventStreamErrorData::= ErrorData



OpenSessionFinalAnswerNotifData ::= CHOICE {
  success OpenSessionFinalAnswerNotifSuccessData,
  declined OpenSessionFinalAnswerNotifDeclinedData,
  failed OpenSessionFinalAnswerNotifFailedData
}

OpenSessionFinalAnswerNotifSuccessData ::= SET {
  sessionId SessionId,
  nextHopIPAddress IpAddress,
  destApplicationIPAddress IpAddress
}

OpenSessionFinalAnswerNotifDeclinedData ::= SET {
  sessionId SessionId,
  cause ErrorCause,
  detail ErrorDetail
}

OpenSessionFinalAnswerNotifFailedData ::= SET {
  sessionId SessionId,
  cause ErrorCause,
  detail ErrorDetail
}

IncomingSessionNotifData ::= SET {
  remoteId RemoteId,
  communicationCategory CommunicationCategory,
  sessionId SessionId
}

FsdAvlNotifData ::= SET {
  fsdAVL FsdAVL
}


SessionClosureNotifData ::= SET {
  sessionId SessionId,
  sessionOriginator SessionOriginator
}

UpcomingDeregistrationNotifData ::= SET {
  timeToDeregistration TimerValue
}

TSSessionOpenData ::= SET {
  localAppIPAddress IpAddress,
  communicationCategory CommunicationCategory,
  recipient Recipient
}


TSSessionOpenedData ::= SET {
  sessionId SessionId
}


TSSessionOpenErrorData ::= ErrorData


SessionStatusData ::= SET {
  sessionOriginator SessionOriginator,
  communicationCategory CommunicationCategory,
  remoteId RemoteId,
  nextHopIPAddress IpAddress OPTIONAL, -- if sessionOriginator is localApplication--
  destApplicationIPAddress IpAddress OPTIONAL, --if sessionOriginator is
localApplication--
  localAppIPAddress IpAddress OPTIONAL -- if sessionOriginator is remoteApplication--
 } (WITH COMPONENTS {
  sessionOriginator(localApplication),
  communicationCategory,
  remoteId,
  nextHopIPAddress PRESENT,
  destApplicationIPAddress PRESENT,
  localAppIPAddress ABSENT
 } | WITH COMPONENTS {
  sessionOriginator(remoteApplication),
  communicationCategory,
  remoteId,
  nextHopIPAddress ABSENT,
  destApplicationIPAddress ABSENT,
  localAppIPAddress PRESENT
})

SessionStatusErrorData ::= ErrorData

StatusPerSessionId ::= SET {
  sessionId SessionId,
  sessionStatusData SessionStatusData
}

SessionsListData ::= SET OF StatusPerSessionId


SessionsListErrorData ::= ErrorData


SessionClosedData ::= SET {
  sessionOriginator SessionOriginator
}


SessionCloseErrorData ::= ErrorData


IncomingSessionNotificationResponseData ::= SET {
  incomingSessionAppResponse IncomingSessionAppResponse,
  localAppIPAddress IpAddress OPTIONAL -- if incomingSessionAppResponse is accepted
--
} (WITH COMPONENTS {
  incomingSessionAppResponse(accepted),
  localAppIPAddress PRESENT
} | WITH COMPONENTS {
  incomingSessionAppResponse(rejected),
  localAppIPAddress ABSENT
})


IncomingSessionNotificationResponseErrorData ::= ErrorData

KeepAliveErrorData ::= ErrorData
```

Annex C. (Informative) Yaml codes of OBAPP

Editor's note: the contents of this annex might not be totally in line with the updates of present FFFIS. The intention is to update this when all normative texts of FRMCS FFFIS v2 are finalized. In v3 this might be replaced by the GitHub link.

---------------------------- Start: Yaml code --

```
openapi: 3.1.0
info:
  version: 1.0
  title: OBapp
  description: |
    OBapp reference point.
    © International Union of Railways (UIC) – Paris, 2024
externalDocs:
  description: OBapp reference point
  url: https://uic.org/rail-system/telecoms-signalling/frmcs
servers:
  - url: '{apiRoot}/obapp/{apiVersion}'
    variables:
      apiRoot:
        default: https://obapp.uic.org
        description: apiRoot as defined in clause 9.5 of UIC FFFIS
      apiVersion:
        default: v1.0
        description: version of the API (see clause 9.6 of UIC FFFIS)
tags:
  - name: version management
    description: Management of interface version
  - name: registration management
    description: Management of application registration
  - name: session management
    description: Management of application sessions
  - name: notification management
    description: Management of application notifications
  - name: keepalive management
```

```
    description: Management of keepalive endpoint for application
paths:
  /versions:
    servers:
      - url: '{apiRoot}/obapp'
        variables:
          apiRoot:
            default: https://obapp.uic.org
            description: apiRoot as defined in clause 9.5 of UIC FFFIS
    get:
      summary: List of OBapp versions supported by the Onboard FRMCS
      description: |
        Operation used to list the OBapp versions supported by the Onboard FRMCS.
        Can be invoked without local registration
      operationId: listObAppVersion
      tags:
        - version management
      responses:
        '200':
          description: Successful operation - list major and minor versions of OBapp supported
by the Onboard FRMCS
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/ApiVersionsData"
              examples:
                apiVersionsData:
                  value: ["v1.0", "v1.1", "v2.0", "v2.1", "v3.0"]
  /registrations:
    post:
      summary: Register an application
      operationId: registerApplication
      tags:
        - registration management
      requestBody:
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/RegisterData'
            examples:
              registerData:
                value:
                  appCategory: etcs
                  staticId: etcs-ob.etcs
                  couplingMode: loose
      responses:
        '201':
          description: Successful registration
          headers:
            Location:
              description: 'URI of the registered application instance'
              required: true
              schema:
                $ref: '#/components/schemas/Uri'
              examples:
                location:
                  value: "https://192.168.1.254/obapp/v1.0/registrations/4210f20b-23e6-4354-bb1a-
be4e0bc56f57"
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/RegisteredData'
              examples:
                registeredData:
                  summary: registration data example
                  value:
                    dynamicId: 4210f20b-23e6-4354-bb1a-be4e0bc56f57
        '400':
          description: Bad request
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/RegisterErrorData'
          examples:
            registerErrorData:
              summary: example error response for registration bad request
              value:
                resource: "https://192.168.1.254/obapp/v1.0/registrations"
                cause: "ILL_FORMED_REQUEST"
                detail: "the field couplingMode must be loose or tight"
      '403':
        description: Forbidden (unrecognized application, ...)
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/RegisterErrorData'
            examples:
              registerErrorData:
                summary: example error response for registration Forbidden
                value:
                  resource: "https://192.168.1.254/obapp/v1.0/registrations"
                  cause: "UNAUTHORIZED"
                  detail: "the voice application cannot use the 'couplingMode' parameter with value
'loose'"
  /registrations/{DynamicId}:
    parameters:
      - name: DynamicId
        in: path
        required: true
        schema:
          $ref: '#/components/schemas/DynamicId'
        examples:
          dynamicId:
            summary: UUID associated to a successful registration
            value: 4210f20b-23e6-4354-bb1a-be4e0bc56f57
    delete:
      summary: De-register an application
      operationId: deregisterApplication
      tags:
        - registration management
      responses:
        '204':
        description: No content (application deregistered)
        '401':
        description: Unauthorized
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/DeRegisterErrorData'
            examples:
              deRegisterErrorData:
                summary: example error response for unauthorized deregistration
                value:
                  resource: "https://192.168.1.254/obapp/v1.0/registrations/4210f20b-23e6-4354-bb1a-be4e0bc56f57"
                  cause: UNREGISTERED
                  details: "Deregistration unauthorized; local binding required"
        '404':
        description: Application with {DynamicId} not found
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/DeRegisterErrorData'
            examples:
              deRegisterErrorData:
                summary: example error response for deregistration of a registration not found
                value:
                  resource: "https://192.168.1.254/obapp/v1.0/registrations/4210f20b-23e6-4354-bb1a-be4e0bc56f57"
                  cause: "NOT_FOUND"
                  details: "Application with {dynamicId} 4210f20b-23e6-4354-bb1a-be4e0bc56f57 not found for deregistration"
  /sessions/{dynamicId}:
    parameters:
        - name: dynamicId
        in: path
        required: true
        schema:
```

```
        $ref: '#/components/schemas/DynamicId'
    get:
      summary: List of sessions for an application
      operationId: listApplicationSessions
      tags:
        - session management
      responses:
        '200':
          description: OK
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/SessionsListData'
              examples:
                sessionsListData:
                    value: ["ca7b8255-447b-416e-97ba-ee0cbd0a1652", "dc9b42be-6945-47ff-9158-
7cce4c9e1580", "eaf35690-17df-4c3d-8f59-51dc1cc1525b" ]
        '401':
          description: Unauthorized, local binding required
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/SessionsListErrorData'
              examples:
                sessionsListErrorData:
                  summary: example error response for unauthorized sessions listing
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57"
                    cause: UNREGISTERED
                    details: "Sessions listing unauthorized; local binding required"
        '404':
          description: Application with {dynamicId} not found
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/SessionsListErrorData'
              examples:
```

```
          sessionsListErrorData:
            summary: example error response for sessions listing for a registration not found
            value:
              resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57"
              cause: "NOT_FOUND"
              details: "Application with {dynamicId} 4210f20b-23e6-4354-bb1a-be4e0bc56f57
not found for sessions listing"
  post:
    summary: Create a session for an application
    operationId: createApplicationSession
    tags:
      - session management
    requestBody:
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/OBSessionOpenData'
          examples:
            obSessionOpenDataIpV4:
              summary: example of open session data ipv4
              value:
                  localAppIPAddress:
                    ipv4Addr: "198.168.100.50"
                  recipientsList:
                    - remoteId: etcs-ts.etcs
                      communicationCategory: basic
            obSessionOpenDataIpV6:
              summary: example of open session data ipv6
              value:
                  localAppIPAddress:
                    ipv6Addr: "2001:db8:85a3::8a2e:370:7334"
                  recipientsList:
                    - remoteId: ato-ob.ato
                      communicationCategory: basic
    responses:
      '201':
        description: 'Session created'
        headers:


        content:
          application/json:
            schema:
              $ref: '#/components/schemas/OBSessionOpenedData'
            examples:
              obSessionOpenedData:
            summary: example of successful response for session opening
            value:
              sessionId: ca7b8255-447b-416e-97ba-ee0cbd0a1652
              sessionStatus: inProgress
      '400':
        description: Bad request
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/OBSessionOpenErrorData'
            examples:
              obSessionOpenErrorData:
            summary: example error response for session opening bad request
            value:
              resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57"
              cause: "ILL_FORMED_REQUEST"
              detail: "Missing parameter"
      '401':
        description: Unauthorized, local binding required
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/OBSessionOpenErrorData'
            examples:
              obSessionsListErrorData:
            summary: example error response for unauthorized session opening
            value:
              resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57"
              cause: UNREGISTERED
              details: "Session opening unauthorized; local binding required"
      '403':
        description: Forbidden, session initiation not authorized
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/OBSessionOpenErrorData'
            examples:
              obSessionsListErrorData:
            summary: example error response for unauthorized session opening
            value:
              resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57"
              cause: UNAUTHORIZED
              details: "Session opening unauthorized"
delete:
summary: Terminate all sessions for an application
operationId: terminateApplicationSessions
tags:
      - session management
responses:
      '204':
        description: No content (sessions terminated)
      '401':
        description: Unauthorized
      '404':
        description: Not found
/sessions/{dynamicId}/{sessionId}:
parameters:
      - name: dynamicId
        in: path
        required: true
        schema:
          $ref: '#/components/schemas/DynamicId'
        examples:
```

```yaml
        dynamicId:
          summary: UUID associated to a successful registration
          value: 4210f20b-23e6-4354-bb1a-be4e0bc56f57
    - name: sessionId
      in: path
      required: true
      schema:
        $ref: '#/components/schemas/SessionId'
      examples:
        sessionId:
          summary: UUID associated to a session
          value: ca7b8255-447b-416e-97ba-ee0cbd0a1652
get:
  summary: Get information on a session of an application
  operationId: listApplicationSessionStatus
  tags:
    - session management
  responses:
    '200':
      description: Successful operation get information about a session
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/SessionStatusData'
          examples:
            sessionStatusData:
              value:
                sessionOriginator: localApplication
                communicationCategory: basic
                nextHopIPAddress:
                  ipv4Addr: 192.168.1.221
                destApplicationIPAddress:
                  ipv4Addr: 172.16.5.1
                remoteId: etcs-ts.etcs
    '401':
      description: Unauthorized, local binding required
      content:
            application/json:
              schema:
                $ref: '#/components/schemas/SessionStatusErrorData'
              examples:
                sessionStatusErrorData:
                  summary: example error response for unauthorized session status listing
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57/ca7b8255-447b-416e-97ba-ee0cbd0a1652"
                    cause: UNREGISTERED
                    details: "Session status listing unauthorized; local binding required"
        '404':
          description: Session {sessionId} not found for application {dynamicId}
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/SessionStatusErrorData'
              examples:
                sessionStatusErrorData:
                  summary: example error response for session status listing for a session not found
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57/ca7b8255-447b-416e-97ba-ee0cbd0a1652"
                    cause: "NOT_FOUND"
                    details: "Session with {sessionId} ca7b8255-447b-416e-97ba-ee0cbd0a1652 not
found for sessions listing"
    put:
      summary: Respond to an incoming session request notification for an application
      operationId: answerApplicationIncomingSessionRequest
      tags:
        - session management
      requestBody:
        description: Request body to respond to an incoming session request
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/IncomingSessionNotificationResponseData'
            examples:
              incomingSessionNotificationResponseData:
                value:
                  appResponse: accepted
                  localAppIPAddress:
                    ipv4Addr: 198.50.200.1
      responses:
        '200':
          description: OK, sent if the application has accepted the incoming session request
          headers:


        '204':
          description: No content (acknowledgement of the application having declined the incoming session request)
        '400':
          description: Bad request
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/IncomingSessionNotificationResponseErrorData'
              examples:
                incomingSessionNotificationResponseErrorData:
                  summary: example error response for incoming session response bad request
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-be4e0bc56f57/ca7b8255-447b-416e-97ba-ee0cbd0a1652"
                    cause: "ILL_FORMED_REQUEST"
                    details: "Incoming session response; missing parameter 'appResponse'"
        '401':
          description: Unauthorized, local binding required
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/IncomingSessionNotificationResponseErrorData'
              examples:
                incomingSessionNotificationResponseErrorData:
                  summary: example error response for unauthorized incoming session response
                  value:
                  resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
  be4e0bc56f57/ca7b8255-447b-416e-97ba-ee0cbd0a1652"
                    cause: UNREGISTERED
                    details: "Incoming session response unauthorized; local binding required"
        '404':
          description: Not found
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/IncomingSessionNotificationResponseErrorData'
              examples:
                incomingSessionNotificationResponseErrorData:
                    summary: example error response for incoming session response for a session not
  found
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
  be4e0bc56f57/ca7b8255-447b-416e-97ba-ee0cbd0a1652"
                    cause: "NOT_FOUND"
                    details: "Session with {SessionId} ca7b8255-447b-416e-97ba-ee0cbd0a1652 not
  found"
    delete:
      summary: Terminate a session for an application
      operationId: terminateApplicationSession
      tags:
        - session management
      responses:
        '204':
          description: No content (session terminated)
        # TODO: consider possible rejection by the server of a session closure?
        '401':
          description: Unauthorized
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/SessionCloseErrorData'
              examples:
                sessionTerminateErrorData:
                  summary: example error response for unauthorized session termination
```

```yaml
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-be4e0bc56f57/ca7b8255-447b-416e-97ba-ee0cbd0a1652"
                    cause: UNREGISTERED
                    details: "Session termination unauthorized; local binding required"
        '403':
          description: Forbidden
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/SessionCloseErrorData'
              examples:
                sessionTerminateErrorData:
                  summary: example error response for unauthorized session termination
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-be4e0bc56f57"
                    cause: UNAUTHORIZED
                    details: "Session termination unauthorized"
          # TODO '404':
  /notifications/{dynamicId}/events:
    parameters:
        - name: dynamicId
          in: path
          required: true
          schema:
            $ref: '#/components/schemas/DynamicId'
          examples:
            dynamicId:
              summary: UUID associated to a successful registration
              value: 4210f20b-23e6-4354-bb1a-be4e0bc56f57
    get:
      summary: Subscribe to notification event stream for an application (SSE)
      operationId: subscribeApplicationNotificationEventStream
      tags:
        - notification management
      responses:
        '200':
          description: Successful
        '401':
          description: Unauthorized
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/EventStreamErrorData'
        '403':
          description: Forbidden
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/EventStreamErrorData'
        '404':
          description: Not found
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/EventStreamErrorData'
              examples:
                eventStreamErrorData:
                  summary: example error response for event stream subscription for a {dynamicId}
not found
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57/events"
                    cause: "NOT_FOUND"
                    details: "Application with {dynamicId} 4210f20b-23e6-4354-bb1a-be4e0bc56f57
not found"
  /notifications/{dynamicId}/channels:
    parameters:
        - name: dynamicId
          in: path
          required: true
          schema:
            $ref: '#/components/schemas/DynamicId'
          examples:
            dynamicId:
```

```
          summary: UUID associated to a successful registration
          value: 4210f20b-23e6-4354-bb1a-be4e0bc56f57
  get:
    summary: Get information on subscriptions to notifications for an application
    operationId: listApplicationSubscriptions
    tags:
      - notification management
    responses:
      '200':
        description: Successful operation list subscriptions
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/SubscriptionsListData'
      '401':
        description: Unauthorized
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/SubscriptionsListErrorData'
            examples:
              subscriptionsListErrorData:
                summary: example error response for unauthorized listing of subscriptions
                value:
                  resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57/channels"
                  cause: UNREGISTERED
                  details: "Listing of subscriptions unauthorized; local binding required"
      '403':
        description: Forbidden
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/SubscriptionsListErrorData'
            examples:
              subscriptionsListErrorData:
                summary: example error response for unauthorized listing of subscriptions
                value:
                  resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-be4e0bc56f57/channels"
                  cause: UNAUTHORIZED
                  details: "Listing of subscriptions unauthorized"
        '404':
          description: Not found
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/SubscriptionsListErrorData'
              examples:
                subscriptionsListErrorData:
                  summary: example error response for listing subscriptions for a {dynamicId} not found
                value:
                  resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-be4e0bc56f57/channels"
                  cause: "NOT_FOUND"
                  details: "Application with {dynamicId} 4210f20b-23e6-4354-bb1a-be4e0bc56f57 not found"
    delete:
      summary: Unsubscribe to all subscriptions to notifications for an application except for the general channel
      operationId: deleteApplicationSubscriptions
      tags:
        - notification management
      responses:
        '204':
          description: No content (subscriptions removal successful)
        '401':
          description: Unauthorized
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/UnsubChannelsErrorData'
              examples:
                subscriptionsRemovalErrorData:
                  summary: example error response for unauthorized removal of subscriptions
                value:
                  resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57/channels"
                  cause: UNREGISTERED
                  details: "Removal of subscriptions unauthorized; local binding required"
        '403':
          description: Forbidden
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/UnsubChannelsErrorData'
              examples:
                subscriptionsRemovalErrorData:
                  summary: example error response for unauthorized removal of subscriptions
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57/channels"
                    cause: UNAUTHORIZED
                    details: "Removal of subscriptions unauthorized"
        '404':
          description: Not found
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/UnsubChannelsErrorData'
              examples:
                subscriptionsRemovalErrorData:
                  summary: example error response for bulk unsubscriptions for a {dynamicId} not
found
                  value:
                    resource: "https://192.168.1.254/obapp/v1.0/sessions/4210f20b-23e6-4354-bb1a-
be4e0bc56f57/channels"
                    cause: "NOT_FOUND"
                    details: "Application with {dynamicId} 4210f20b-23e6-4354-bb1a-be4e0bc56f57
not found"
  /notifications/{dynamicId}/channels/{channel}:
    parameters:
        - name: dynamicId
          in: path
          required: true
          schema:
            $ref: '#/components/schemas/DynamicId'
        - name: channel
          in: path
          required: true
          schema:
            $ref: '#/components/schemas/NotifChannel'
    delete:
      summary: Unsubscribe to a specific channel for an application
      operationId: unsubscribeApplicationNotificationChannel
      tags:
        - notification management
      responses:
        '204':
          description: No content (channel unsubscribed)
  /keepalive/{dynamicId}:
    parameters:
        - name: dynamicId
          in: path
          required: true
          schema:
            $ref: '#/components/schemas/DynamicId'
    get:
      summary: Check the server is alive at the HTTP level
      operationId: checkKeepalive
      tags:
        - keepalive management
      responses:
        '204':
          description: No content (acknowledgement of the server being responsive on the control
plane)
```

---------------------------- End: Yaml code ---

Editor’s note : the JSON schemas of Data structures in http messages are derived from the corresponding ASN.1 structures in Annex A and can be added as informative text in this Annex.

Annex D. Interoperability requirements in EU

This annex is the placeholder for identifying the requirements relevant for interoperability in the European Union, i.e. the requirements, with respect to the authorisation in the EU according to the TSI, that are considered in the European Directives to be relevant for interoperability as fulfilling the essential requirements for the Control-Command and Signalling (CCS) subsystem related to safety and technical compatibility which must be met by the rail system, the subsystems, and the interoperability constituents, including interfaces according to the corresponding conditions set out in Directive (EU) 2016/797. It is mandatory that each railway subsystem in the EU meets these requirements on lines under the scope of the Directive and the CCS TSI to ensure technical compatibility between Member States and safe integration between train and track.

At this stage, the version of this specification is not considered complete for the purpose of tendering On-Board FRMCS equipment, and the identification of all requirements relevant for interoperability is for further study.

This annex is therefore only informative.

D.1 Scope and Purpose

D.1.1 This document lists a classification into categories of all the clauses in the Form Fit Functional Interface Specification (FRMCS FFFIS).

D.1.2 The purpose of this document is to ease the assessment of the compliance of a FRMCS on-board and trackside equipment with the FFFIS.

D.1.3 To that effect, this document comprehensively identifies which clauses contain requirements allocated to entities or application making use of the OBapp interface and conversely which ones do not.

D.2 Definitions

D.2.1 The following categories are used:

D.2.1.1 Candidate MI requirement: a requirement that, as expressed in this FRMCS V2 specification, is considered to be related to interoperability (MI=mandatory for interoperability in Europe). This pre-assessment can be used to focus on the requirements that shall be completed in FRMCS V3 and also, to drive attention to the fact that a latter inclusion of the identified functionality will have to be done with careful attention to the compatibility between new and previous installations.

D.2.1.2 Note that the identification of “candidate MI” requirements is just an indication to the reader, and, since this FRMCS V2 version is not to be part of a CCS TSI and the specifications are not yet ready for product production, it has no impact on the certification tasks of the Notified Bodies.

Section	(M) Requirements	Candidate MI for OB FRMCS	Candidate MI for OB Application	Candidate MI for FRMCS TS GW	Candidate MI for TS Application
6.3 OB$_{APP}$ Security requirements	6.3.1	MI (NOTE 1)	Not applicable	Not applicable	Not applicable
	6.3.2 6.3.3 6.3.4	MI	MI		
6.6 TS$_{APP}$ Security requirements	6.6.1 6.6.2 6.6.3	Not applicable	Not applicable	MI	MI
6.7 TLS requirements	6.7.1	MI	MI	Not applicable	Not applicable
	6.7.2	Not applicable	Not applicable	MI	MI
7 OB$_{APP}$ Low layers specifications and protocol stacks	7.2.2 7.3.2i 7.4.1* 7.4.2	MI	Not applicable	Not applicable	Not applicable
8 TS$_{APP}$ Low layers specifications and protocol stacks	8.3.2i 8.4.1	Not applicable	Not applicable	MI	Not applicable
9 OB$_{APP}$ API Services	Clauses 9.1 to 9.15	The candidate MI for OB$_{APP}$ API services is defined in Table D-2. Where an API service is MI all corresponding M requirements are MI as well.		Not applicable	
10 TS$_{APP}$ API Services	Clauses 10.1 to 10.13	Not applicable		The candidate MI for TS$_{APP}$ API services is defined in Table D-3. Where an API service is MI all corresponding M requirements are MI as well.	
NOTE 1: only if FRMCS On-Board is connected to an Ethernet Consist Network compliant with [SUBSET-147]					

Table D-1. Candidate MI for OBAPP and TSAPP

The following table represents the OB$_{APP}$ API services (represented by a method over an endpoint) which are candidate MI for:

- On-Board FRMCS
- OB Tight-Coupled application
- OB Loose-Coupled application requiring OB-Originated session
- OB Loose-Coupled application requiring OB-Terminated session

Index	Endpoint	Method	Purpose	OB FRMCS	OB Tight-Coupled application	OB Loose-Coupled application	
						OB-originated	OB-terminated
1	/versions	GET	Obtain supported API versions by the On-Board FRMCS	MI	MI	MI	MI
2	/registrations	POST	Register an application	MI	MI	MI	MI
3	/registrations/{dynamicId}	DELETE	De-register an application	MI			
4	/sessions/{dynamicId}	GET	List of sessions for an application	MI			
5		POST	Create a session for an application	MI		MI	
6	/sessions/{dynamicId}/{sessionId}	GET	Get information on a session of an application	MI			
7		PUT	Accept an incoming session for an application	MI			MI
8		DELETE	Terminate a session for an application	MI		MI	MI
9	/notifications/{dynamicId}/events	GET	Subscribe to the event stream to receive notifications	MI	MI	MI	MI
10	/ notifications/{dynamicId}/channels	GET	Obtain list of notifications to which application has subscriptions	MI			
11		DELETE	Unsubscribe all the notification channels (except general notifications linked to the event stream)	MI			
12	/notifications/{dynamicId}/channels/location	POST	Subscribe to the location reporting channel	MI	MI (NOTE 1)		
13		DELETE	Unsubscribe from a specific channel for an application	MI	MI (NOTE 1)		
14	/notifications/{dynamicId}/channels/{subscriptionId}	DELETE	Unsubscribe from a specific notification subscription	MI	MI (NOTE 1)		

15	/keepalive/{dynamicId}/	GET	Request a life signal from On-Board FRMCS	MI		MI (NOTE 2)	
NOTE 1: only for those OB Tight-Coupled applications for which the location service common function is (mandatorily) used according to Appendix G of FRMCS FRS V2.0.0. NOTE 2: only for Automatic Train Protection communication (FRMCS FRS clause 5.9)							

Table D-2. Candidates MI for OB$_{APP}$ API services

The following table represents the TS$_{APP}$ API services (represented by a method over an endpoint) which are candidate MI for:

- FRMCS Trackside Gateway
- Trackside Tight-Coupled application
- Trackside Loose-Coupled application requiring TS-Originated session
- Trackside Loose-Coupled application requiring TS-Terminated session

Index	**Endpoint**	**Method**	**Purpose**	**FRMCS Trackside Gateway**	**TS Tight-Coupled application**	**TS Loose-Coupled application**	
						TS-originated	**TS-terminated**
1	/versions	GET	Obtain supported API versions by the On-Board FRMCS	MI	MI	MI	MI
2	/registrations	POST	Register an application	MI	MI	MI	MI
3	/registrations/{dynamicId}	DELETE	De-register an application	MI			
4	/sessions/{dynamicId}	GET	List of sessions for an application	MI			
5		POST	Create a session for an application	MI		MI	
6	/sessions/{dynamicId}/{sessionId}	GET	Get information on a session of an application	MI			
7		PUT	Accept an incoming session for an application	MI			MI
8		DELETE	Terminate a session for an application	MI		MI	MI
9	/notifications/{dynamicId}/events	GET	Subscribe to the event stream to receive notifications	MI	MI	MI	MI
10	/keepalive/{dynamicId}/	GET	Request a life signal from On-Board FRMCS	MI			MI (NOTE 1)

NOTE 1: only for Automatic Train Protection communication (FRMCS FRS clause 11.4)

Table D-3. Candidates MI for TSAPP API services

Printed by

International Union of Railways

16, rue Jean Rey 75015 Paris - France

April 2025

Legal deposit April 2025

ISBN 978-2-7461-3120-0