ETSI EN 303 146-1 V1.3.1 (2018-06)

EUROPEAN STANDARD

Reconfigurable Radio Systems (RRS); Mobile Device (MD) information models and protocols; Part 1: Multiradio Interface (MURI)

Reference

REN/RRS-0218

Keywords

interface, mobile, SDR

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

Copyright Notification

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This European Standard (EN) has been produced by ETSI Technical Committee Reconfigurable Radio Systems (RRS).

The present document is part 1 of a multi-part deliverable covering the Mobile Device Information Models and Protocols, as identified below:

Part 1: "Multiradio Interface (MURI)";

Part 2: "Reconfigurable Radio Frequency Interface (RRFI)";

Part 3: "Unified Radio Application Interface (URAI)";

Part 4: "Radio Programming Interface (RPI)".

National transposition dates	
Date of adoption of this EN:	5 June 2018
Date of latest announcement of this EN (doa):	30 September 2018
Date of latest publication of new National Standard or endorsement of this EN (dop/e):	31 March 2019
Date of withdrawal of any conflicting National Standard (dow):	31 March 2019

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document defines an information model and protocol for multiradio interface for reconfigurable mobile devices. The work is based on the Use Cases defined in ETSI TR 102 944 [i.1], on the system requirements defined in ETSI EN 302 969 [i.7] and on the radio reconfiguration related architecture for mobile devices defined in ETSI EN 303 095 [i.2].

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

Not applicable.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TR 102 944: "Reconfigurable Radio Systems (RRS); Use Cases for Baseband Interfaces for Unified Radio Applications of Mobile Device".

[i.2] ETSI EN 303 095: "Reconfigurable Radio Systems (RRS); Radio Reconfiguration related Architecture for Mobile Devices".

[i.3] ETSI EN 303 146-2: "Reconfigurable Radio Systems (RRS); Mobile Device Information Models and Protocols; Part 2: Reconfigurable Radio Frequency Interface (RRFI)".

[i.4] ETSI TR 102 839: "Reconfigurable Radio Systems (RRS); Multiradio Interface for Software Defined Radio (SDR) Mobile Device Architecture and Services".

[i.5] IEEE 1900.4$^{\text{TM}}$-2009: "IEEE Standard for Architectural Building Blocks Enabling Network-Device Distributed Decision Making for Optimized Radio Resource Usage in Heterogeneous Wireless Access Networks".

[i.6] Recommendation ITU-T X.680: "Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation".

[i.7] ETSI EN 302 969: "Reconfigurable Radio Systems (RRS); Radio Reconfiguration related Requirements for Mobile Devices".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

association: logical communication link to a Radio Access Network or a peer equipment

NOTE 1: Typically, some control signalling is necessary to maintain the association. No user data transfer may occur with only an association present, but a data flow may be established into an association for this purpose.

NOTE 2: Peer equipment is any communication counterpart of a reconfigurable mobile device. It can be reached by establishing a logical communication link (i.e. an association) between the reconfigurable mobile device and peer equipment.

channel: designated part of the information transfer capability having specified characteristics, provided at the user network interface

NOTE: It is the over-the-air wireless propagation channel which is used to convey an information signal from transmitter to receiver. This definition is specified in ETSI EN 303 095 [i.2].

communication services layer: layer related to communication services supporting generic applications

NOTE: A communication services layer supports generic applications like Internet access. In the present document, it consists of Administrator, Mobility Policy Manager (MPM), Networking stack and Monitor.

link: connecting one location to another through a given Radio Access Technology for the purpose of transmitting and receiving digital information

NOTE: Each link is conveyed over a given Channel.

Radio Application (RA): software which enforces the generation of the transmit RF signals or the decoding of the receive RF signals

NOTE 1: The Software is executed on a particular radio platform or an RVM as part of the radio platform.

NOTE 2: RAs might have different forms of representation. They are represented as:

- source codes including Radio Library calls of Radio Library native implementation and Radio HAL calls;
- IRs including Radio Library calls of Radio Library native implementation and radio HAL calls;
- executable codes for a particular radio platform.

radio computer: part of mobile device hardware working under ROS control and on which RAs are executed

NOTE: A radio computer typically includes programmable processors, hardware accelerators, peripherals, etc. RF part is considered to be part of peripherals.

Radio Control Framework (RCF): control framework which, as a part of the OS, extends OS capabilities in terms of radio resource management

NOTE: RCF is a control framework which consists of Configuration Manager (CM), Radio Connection Manager (RCM), Flow Controller (FC) and Multiradio Controller (MRC). The Resource Manager (RM) is typically part of OS.

reconfigurable mobile device: mobile device with radio communication capabilities providing support for radio reconfiguration

NOTE: Reconfigurable Mobile Devices include but are not limited to: Smartphones, Feature Phones, Tablets, and Laptops.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ASN.1	Abstract Syntax Notation One
BLER	BLock Error Rate
CM	Configuration Manager
CSL	Communication Services Layer
FC	Flow Controller
ID	IDentification
IR	Intermediate Representation
MD	Mobile Device
MDRC	Mobile Device Reconfiguration Class
MPM	Mobility Policy Manager
MRC	MultiRadio Controller
MURI	MUltiRadio Interface
OS	Operating System
RA	Radio Application
RAN	Radio Access Network
RAP	Radio Application Package
RAT	Radio Access Technology
RCF	Radio Control Framework
RCM	Radio Connection Manager
RF	Radio Frequency
RM	Resource Manager
ROS	Radio Operating System
RPI	Radio Programming Interface
RRFI	Reconfigurable Radio Frequency Interface
SINR	Signal to Interference plus Noise Ratio
SW	SoftWare
TCP/IP	Transmission Control Protocol/Internet Protocol
UML	Unified Modeling Language
URA	Unified Radio Applications
URAI	Unified Radio Application Interface

4 Introduction

A reconfigurable MD is capable of running multiple radios simultaneously and of changing the set of radios by loading new Radio Application Package (RAP). All Radio Applications (RAs) are called Unified Radio Applications (URAs) when they exhibit a common behaviour from the reconfigurable MD's point of view in ETSI EN 303 095 [i.2]. In order to run multiple URAs, the reconfigurable MD will include Communication Services Layer (CSL), Radio Control Framework (RCF), Radio Platform and 4 sets of interfaces for their interconnection.

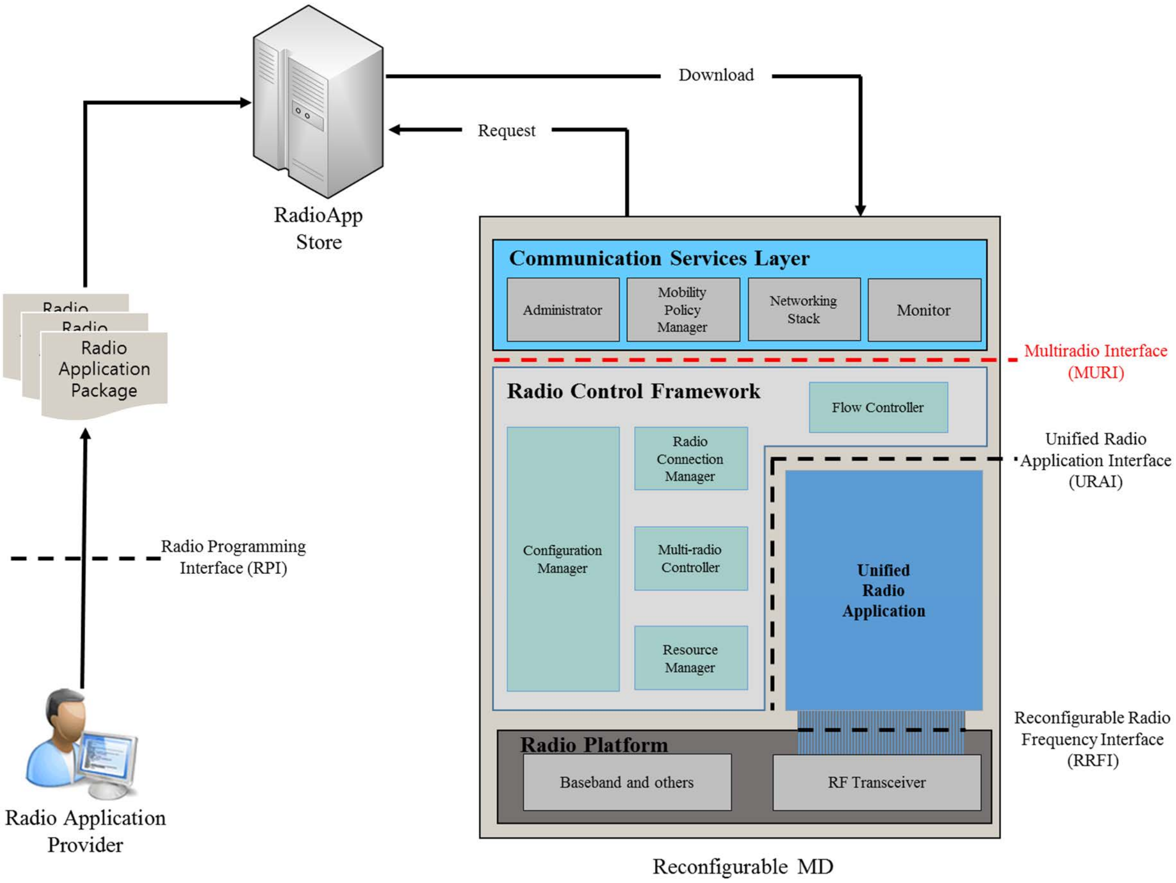

Figure 4.1: Four sets of interfaces for Reconfigurable MD

Figure 4.1 illustrates the Reconfigurable MD architecture with the 4 sets of interfaces, i.e.:

- MURI for interfacing CSL and RCF which is the scope of the present document;
- RRFI for interfacing URA and RF Transceiver in ETSI TS 103 146-2 [i.3];
- URAI for interfacing URA and RCF in ETSI TR 102 839 [i.4];
- RPI for allowing an independent and uniform production of RAs in ETSI TR 102 839 [i.4].

The present document defines MURI.

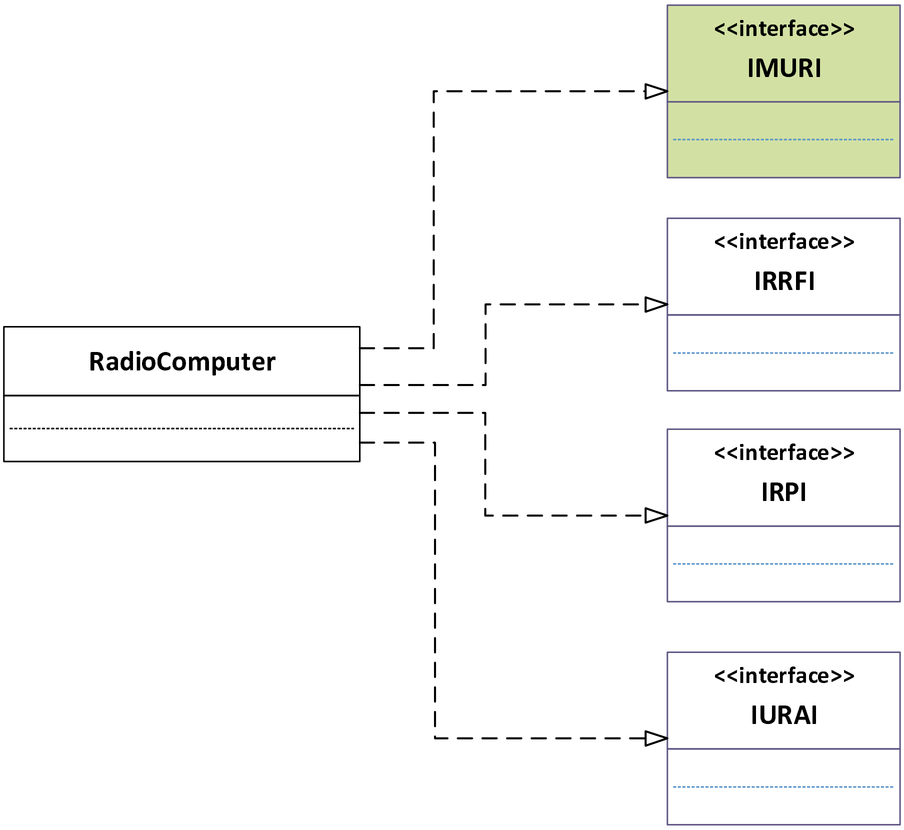

Figure 4.2: UML class diagram for Radio Computer interfaces

Figure 4.2 illustrates UML class diagram for Radio Computer interfaces. The reconfigurable MD may be seen as a Radio Computer where individual URAs are engineered as software entities in ETSI EN 303 095 [i.2].

The present document is organized as follows:

- clause 5 describes the system identification;
- clause 6 describes the notational tool for defining both information model classes and interface classes;
- clause 7 describes the information model for radio computer; and
- clause 8 describes the interface definition.

While UML is used for defining the information model and protocol related to MURI, other modeling languages could be used as well.

5 System Identification

5.1 Radio Computer Structure

Figure 5.1 illustrates how CSL and RCF interact with each other using MURI.

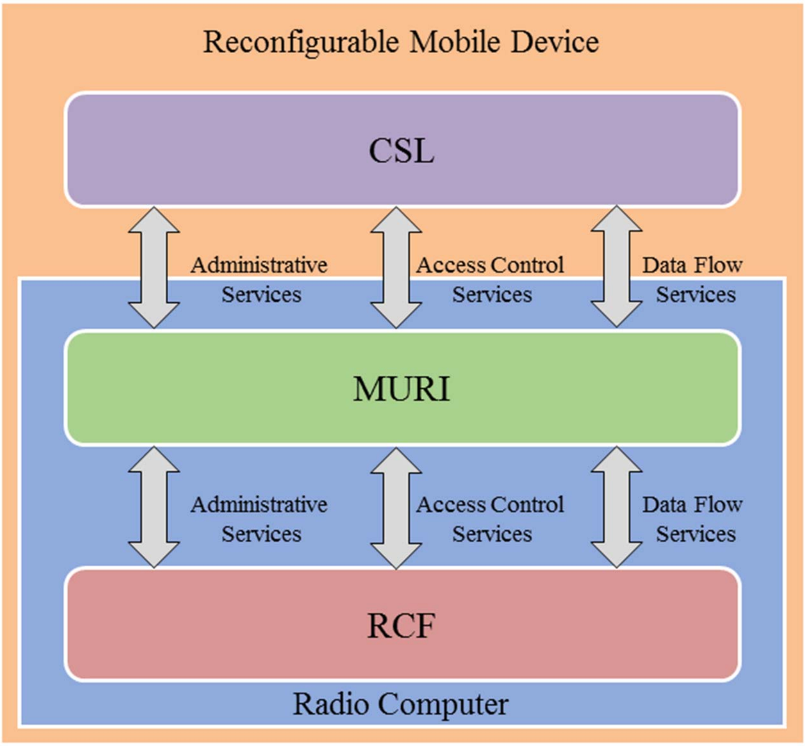

Figure 5.1: Interconnection between CSL and RCF using MURI for Reconfigurable MD

As shown in figure 5.1, MURI supports 3 kinds of services:

- **Administrative Services**

These services are used by some device configuration application i.e. Administrator which is included in the CSL, to (un)install a new URA into the Reconfigurable MD and create/delete an instance of the URA. Installation and loading may take place both at device start-up time to set up the network connection as well as during run-time, whenever reconfiguration of available URAs is needed. MURI does not make any assumption on how and when the mobile device will detect the need of the reconfiguration.

- **Access Control Services**

These services are used by the MPM to maintain the user policies and preferences related to the usage of different RATs and to make a selection between them. Modelling of such preferences and selection algorithms is not in the scope of the present document; however, the MURI specification covers the information exchange of RAT selection decisions between CSL and RCF. The preferences themselves may originate either locally from applications or end user settings as well as in a distributed manner from network operator or from a cognitive radio management framework.

- **Data Flow Services**

These services are used by the networking stack of the Reconfigurable MD, such as the TCP/IP stack. Therefore data flow services represent the set of (logical) link layer services, which are provided in a uniform manner regardless of which URAs are active.

The Communication Services Layer (CSL) and Radio Control Framework (RCF) are defined in ETSI EN 303 095 [i.2].

5.2 MURI System Requirement Mapping

The Radio Computer components above described shall support the MURI system requirements shown in table 5.1 and described in clauses 6.1 and 6.2 of ETSI EN 302 969 [i.7].

Table 5.1: Mapping of Radio Computer Components to the system requirements described in ETSI EN 302 969

Entity/Component/Unit	System Requirements [i.7]	Comments
Administrator	R-FUNC-RAT–01	The requirement is described in clause 6.1.1 of [i.7].
	R-FUNC-RA-01	The requirement is described in clause 6.2.1 of [i.7].
	R-FUNC-MDR-01	The requirement is described in clause 6.4.1 of [i.7].
	R-FUNC-MDR-02	The requirement is described in clause 6.4.2 of [i.7].
	R-FUNC-MDR-03	The requirement is described in clause 6.4.3 of [i.7].
Mobility Policy Manager	R-FUNC-RAT–01	The requirement is described in clause 6.1.1 of [i.7].
	R-FUNC-RAT–04	The requirement is described in clause 6.1.4 of [i.7].
	R-FUNC-RA-03	The requirement is described in clause 6.2.3 of [i.7].
Networking Stack	R-FUNC-RAT–05	The requirement is described in clause 6.1.5 of [i.7].
	R-FUNC-RA-04	The requirement is described in clause 6.2.4 of [i.7].
Configuration Manager	R-FUNC-RAT–01	The requirement is described in clause 6.1.1 of [i.7].
	R-FUNC-RAT–02	The requirement is described in clause 6.1.2 of [i.7].
	R-FUNC-RA-01	The requirement is described in clause 6.2.1 of [i.7].
	R-FUNC-MDR-01	The requirement is described in clause 6.4.1 of [i.7].
	R-FUNC-MDR-02	The requirement is described in clause 6.4.2 of [i.7].
	R-FUNC-MDR-03	The requirement is described in clause 6.4.3 of [i.7].
Radio Connection Manager	R-FUNC-RAT–01	The requirement is described in clause 6.1.1 of [i.7].
	R-FUNC-RAT–02	The requirement is described in clause 6.1.2 of [i.7].
	R-FUNC-RAT–03	The requirement is described in clause 6.1.3 of [i.7].
	R-FUNC-RAT–04	The requirement is described in clause 6.1.4 of [i.7].
	R-FUNC-RAT–06	The requirement is described in clause 6.1.6 of [i.7].
	R-FUNC-RA-03	The requirement is described in clause 6.2.3 of [i.7].
Flow Controller	R-FUNC-RAT–01	The requirement is described in clause 6.1.1 of [i.7].
	R-FUNC-RAT–03	The requirement is described in clause 6.1.3 of [i.7].
	R-FUNC-RAT–05	The requirement is described in clause 6.1.5 of [i.7].
	R-FUNC-RAT–06	The requirement is described in clause 6.1.6 of [i.7].
	R-FUNC-RA-04	The requirement is described in clause 6.2.4 of [i.7].

6 Notational Tools

6.1 Notational Tool for Information Model Classes

Table 6.1 shows a template for defining information model classes in IEEE 1900.4^{TM}-2009 [i.5]. Each information model class is defined in clause 7.2 in accordance with the template shown in table 6.1.

NOTE: ASN.1 is used throughout the present document for abstract type definitions; however, alternative ways are possible and are not excluded.

Table 6.1: Template for defining Information Model Classes

Class<Class name>[(abstract class)]			
<Description of the class>			
DERIVED FROM	<List of super-classes>		
ATTRIBUTES			
<Attribute name> [<optional>]	Value type: <Attribute value type>	Possible access: <Attribute access qualifier>	Default value: <Default value>
<Description of the attribute>			
CONTAINED IN	<List of classes, whose instances may contain an instance of this class. If this class is an abstract class, that is, it is used for further refinement only and will never be instantiated, then this list is empty.>		
CONTAINS	<List of classes, whose instances may be contained in an instance of this class. Constraints used are: [*] - zero or more instances, [+] - one or more instances, [<n>] - exactly n instances, [<m> - <n>] - not less than m and not more than n instances.>		
SUPPORTED EVENTS	<List of event names that are detected by this class and lead potentially to a corresponding event report>		

Further details on the template in table 6.1 are given below:

- <Class name> is the name of the Class as it appears in the corresponding model. Additional information is also included in case the class in question has been specified as an abstract one.
- DERIVED FROM field identifies the super class of the class in case of sub-classing.
- ATTRIBUTES field describes the attributes that have been defined in the class. More specifically:
  - <Attribute name> identifies the name of an attribute, as it is included in the class definition.
  - <Attribute value type> holds the type of the attribute specified in Abstract Syntax Notation One (ASN.1). Details related to the ASN.1 module are specified in Annex A of the present document.
  - <Attribute access qualifier> provides information about the level of accessibility of the attribute. This may include: 'Read', 'Write', 'Read-Write', 'Add-Remove' (for list-type attributes), 'Read-Add-Remove', and 'None' (for internal access only).
- CONTAINED IN field includes a list of classes whose instances may contain an instance of this class; containment is a strong aggregation relationship, that is, a contained instance is for its lifetime bound to its container object and it is contained only in this one container.
- CONTAINS field provides a list of classes whose instances may be contained in an instance of the class in question.
- SUPPORTED EVENTS field includes a list of event names that are detected by this class and lead potentially to a corresponding event report.

6.2 Notational Tool for Interface Classes

Table 6.2 shows a template for defining interface classes for MURI. Each interface class for MURI will be defined in clause 8.5 in accordance with the template shown in table 6.2.

Table 6.2: Template for defining Interface Classes

Class<Class name>[(abstract class)]		
<Description of the class>		
OPERATIONS		
<Operation name>	Return type: <Operation return type>	Value type: <Operation value type>
<Description of the operation>		

The template fields in table 6.2 are described below:

- <Class name> is the name of the Class as it appears in the corresponding model. Additional information is also included in case the class in question has been specified as an abstract one.
- OPERATIONS field describes the operations that have been defined in the class. More specifically:
  - <Operation name> identifies the name of an operation, as it is included in the class definition.
  - <Return type> identifies the type of return value at the corresponding operation. Details related to the ASN.1 module are specified in annex B of the present document.
  - <Value type> identifies the access levels for member functions: public, private, protected.

7 Information Model for Radio Computer

7.1 Radio Computer

Figure 7.1 shows the UML class diagram for Radio Computer classes related to MURI which are required to support Software Reconfiguration.

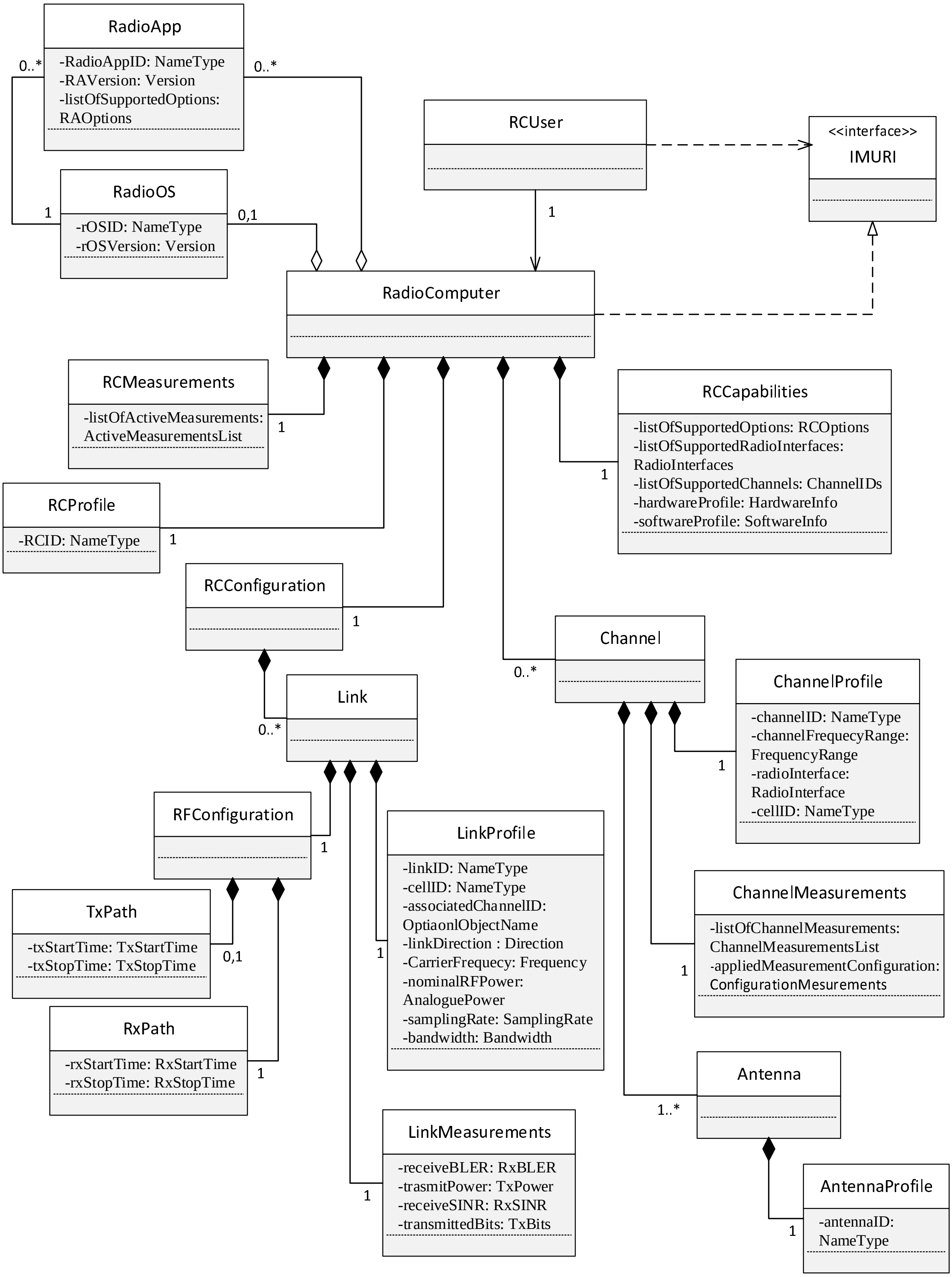
RadioApp
-RadioAppID: NameType
-RAVersion: Version
-listOfSupportedOptions: RAOptions
0..*
0..*
RCUser
<<interface>>
IMURI
RadioOS
-rOSID: NameType
-rOSVersion: Version
1
0,1
1
RadioComputer
RCMeasurements
-listOfActiveMeasurements: ActiveMeasurementsList
1
RCCapabilities
-listOfSupportedOptions: RCOptions
-listOfSupportedRadioInterfaces: RadioInterfaces
-listOfSupportedChannels: ChannelIDs
-hardwareProfile: HardwareInfo
-softwareProfile: SoftwareInfo
1
RCProfile
-RCID: NameType
1
RCConfiguration
1
Channel
0..*
Link
0..*
ChannelProfile
-channelID: NameType
-channelFrequecyRange: FrequencyRange
-radioInterface: RadioInterface
-cellID: NameType
1
RFConfiguration
1
LinkProfile
-linkID: NameType
-cellID: NameType
-associatedChannelID: OptiaonlObjectName
-linkDirection : Direction
-CarrierFrequecy: Frequency
-nominalRFPower: AnaloguePower
-samplingRate: SamplingRate
-bandwidth: Bandwidth
1
TxPath
-txStartTime: TxStartTime
-txStopTime: TxStopTime
0,1
ChannelMeasurements
-listOfChannelMeasurements: ChannelMeasurementsList
-appliedMeasurementConfiguration: ConfigurationMesurements
1
RxPath
-rxStartTime: RxStartTime
-rxStopTime: RxStopTime
1
Antenna
1..*
LinkMeasurements
-receiveBLER: RxBLER
-trasmitPower: TxPower
-receiveSINR: RxSINR
-transmittedBits: TxBits
1
AntennaProfile
-antennaID: NameType
1

Figure 7.1: UML class diagram for Radio Computer classes related to MURI

The Radio Computer classes related to MURI are defined as follows:

- **RCUser**

This class describes information related to a user of the Radio Computer. Each instance of RCUser class depends on one instance of RadioComputer class.

- **RadioApp**

This class describes an installed Radio Application. Each instance of a "Radio Computer" class can relate to zero or several instances of RadioApp class (0..*). Each instance of RadioApp class is associated with one instance of Radio OS class.

- **RadioOS**

This class describes an installed Radio OS. Each instance of a "Radio Computer" class can relate to zero or one instance of RadioOS class (0,1). Each instance of RadioOS class is associated with zero or several instances of RadioApp class (0..*).

- **RadioComputer**

This class contains all URA(s) related information about resources and interactions related to hardware and software of a reconfigurable MD, for example, computational/spectral resource usage, collection of context information, channel measurement results, etc.

- **RCProfile**

This class contains general information about the Radio Computer, for example, terminal Identification (ID). Each instance of a "RadioComputer" class can have only one instance of RCProfile class as a member.

- **RCCapabilities**

This class contains information about Radio Computer capabilities including hardware, software, transmission and measurement capabilities such as supported RATs and maximum transmission power. Each instance of RadioComputer class shall have only one instance of RCCapabilities class as a member.

- **Channel**

This class contains one radio channel that may or may not be used by an active radio link. Each instance of RadioComputer class can have zero, one or several instances of Channel class as members (0..*). In case of an active radio link, at least one Channel class is available.

- **ChannelProfile**

This class contains general information about the radio channel such as channel ID, center frequency, bandwidth, and used RAT. Each instance of Channel class shall have only one instance of Channel Profile class as a member.

- **ChannelMeasurements**

This class contains current measurements (instantaneous measurement data and related metadata) and the applied measurement configuration related to this radio channel such as interference and load measurements. Each instance of Channel class shall have only one instance of ChannelMeasurements class as a member.

- **Antenna**

This class contains information about antenna selection. Each instance of Channel class shall have at least one instance of Antenna class as a member. (1..*).

- **AntennaProfile**

This class contains general information about this antenna, such as antenna port, applicable frequency range and antenna gain. Each instance of Antenna class shall have only one instance of AntennaProfile class as a member.

- **RCConfiguration**

This class contains information about the current configuration of Radio Computer. Each instance of RadioComputer class shall have only one instance of RCConfiguration class as a member.

- **Link**

This class contains information about one active URA and the corresponding connection between the Reconfigurable MD and the Radio Access Network (RAN). Each instance of RCConfiguration class has zero, one or several instances of Link class as members (0..*). Each instance of Link class is associated with one instance of Channel class.

- **LinkProfile**

This class contains general information about this active connection, for example, link Identification (ID), serving cell ID, channel used, etc. Each instance of Link class shall have only one instance of LinkProfile class as a member.

- **LinkMeasurements**

This class contains current measurements (instantaneous measurement data and related metadata) related to this active connection, such as Block Error Rate (BLER), power, and Signal to Interference plus Noise Ratio (SINR) measurements. Each instance of Link class shall have only one instance of LinkMeasurements class as a member.

- **RFConfiguration**

This class contains information about the configuration of the RF transceiver. Each instance of Link class shall have only one instance of RFConfiguration class as a member.

- **TxPath**

This class contains information about one transmit path. Each instance of RFConfiguration class has zero or one instance of TxPath class as a member (0,1).

- **RxPath**

This class contains information about one receive path. Each instance of RFConfiguration class shall have only one instance of RxPath class as a member.

- **RCMeasurements**

This class contains current measurements (instantaneous measurement data and related metadata) related to Reconfigurable MD such as battery capacity, user mobility, MD location determination, and connection history information. Each instance of RadioComputer class shall have only one instance of RCMeasurements class as a member.

NOTE: The Channel Class is separate from the Link Class, but the Channel Measurements may be based on any MD configuration which may or may not be used for the final Link Configuration.

7.2 Class Definitions for Information Model

Each class of Radio Computer can be defined using the template presented in clause 6.1 and in accordance with the UML diagram of figure 7.1 which specifies the relations among all the classes of Radio Computer. Radio Computer classes defined in this way are shown in tables 7.1 to 7.18.

Table 7.1: RadioComputer Class

Class **RadioComputer**	
This class contains all URA related information about resources and interactions related to hardware and software of a reconfigurable MD.	
DERIVED FROM	
ATTRIBUTES	
CONTAINED IN	
CONTAINS	**RCCapabilities [**1**], RCConfiguration [**1**], RCMeasurements [**1**], Channel [*], RCProfile [**1**], RadioAPP [*], RadioOS [0-1]**
SUPPORTED EVENTS	

Table 7.2: RadioApp Class

Class **RadioApp**			
This class describes installed Radio Application.			
DERIVED FROM			
ATTRIBUTES			
RadioAppID	*Value type:* NameType	*Possible access:* Read	*Default value:* Not specified
This attribute describes ID of installed Radio Application.			
RAVersion	*Value type:* Version	*Possible access:* Read	*Default value:* Not specified
This attribute describes a version of Radio Application.			
listOfSupportedOptions	*Value type:* RAOptionsList	*Possible access:* Read	*Default value:* Not specified
This attribute contains a list of supported options.			
CONTAINED IN	**RadioComputer**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.3: RadioOS Class

Class **RadioOS**			
This class describes installed Radio OS.			
DERIVED FROM			
ATTRIBUTES			
rOSID	*Value type:* NameType	*Possible access:* Read	*Default value:* Not specified
This attribute describes ID of Radio OS.			
rOSVersion	*Value type:* Version	*Possible access:* Read	*Default value:* Not specified
This attribute describes a version of Radio OS.			
CONTAINED IN	**RadioComputer**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.4: RCProfile Class

Class **RCProfile**			
This class contains general information about the Radio Computer.			
DERIVED FROM			
ATTRIBUTES			
RCID	*Value type:* NameType	*Possible access:* Read	*Default value:* Not specified
This attribute describes ID of radio computer.			
CONTAINED IN	**RadioComputer**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.5: RCCapabilities Class

Class **RCCapabilities**			
This class contains information about Radio Computer capabilities including hardware, software, transmission and measurement capabilities.			
DERIVED FROM			
ATTRIBUTES			
listOfSupportedOptions	*Value type:* RCOptionsList	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes a list of supported options.			
listOfSupportedRadioInterfaces	*Value type:* RadioInterfacesList	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes radio interfaces supported by this Radio Computer.			
listOfSupportedChannels	*Value type:* ChannelIDsList	*Possible access:* Read-Write	*Default value:* Not specified
This attributes describes frequency channels supported by this Radio Computer.			
hardwareProfile	*Value type:* HardwareInfo	*Possible access:* Read-Write	*Default value:* Not specified
This attributes describes hardware capabilities of this Radio Computer.			
softwareProfile	*Value type:* SoftwareInfo	*Possible access:* Read-Write	*Default value:* Not specified
This attributes describes software capabilities of this Radio Computer.			
CONTAINED IN	**RadioComputer**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.6: Channel Class

Class **Channel**	
This class describes one frequency channel that may or may not have active connections on it.	
DERIVED FROM	
ATTRIBUTES	
CONTAINED IN	**RadioComputer**
CONTAINS	**ChannelProfile [**1**], ChannelMeasurements [**1**], Antenna [+]**
SUPPORTED EVENTS	

Table 7.7: ChannelProfile Class

Class **ChannelProfile**			
This class contains general information about this frequency channel.			
DERIVED FROM			
ATTRIBUTES			
channelID	*Value type:* NameType	*Possible access:* Read	*Default value:* Not specified
This attribute describes ID of channel.			
channelFrequencyRange	*Value type:* FrequencyRange	*Possible access:* Read	*Default value:* Not specified
This attribute describes a value of channel frequency range.			
radioInterface	*Value type:* RadioInterface	*Possible access:* Read	*Default value:* Not specified
This attribute describes a radio interface.			
cellID	*Value type:* NameType	*Possible access:* Read	*Default value:* Not specified
This attribute describes ID of connected cell.			
CONTAINED IN	**Channel**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.8: ChannelMeasurements Class

Class **ChannelMeasurements**			
This class contains current measurements related to this frequency channel.			
DERIVED FROM			
ATTRIBUTES			
listOfChannelMeasurements	*Value type:* ChannelMeasurementsList	*Possible access:* Read	*Default value:* Not specified
This attribute describes a list of channel measurements.			
appliedMeasurementsConfiguration	*Value type:* ConfigurationMeasurements	*Possible access:* Read	*Default value:* Not specified
This attribute describes configuration option of the MD, e.g. which Antenna(s) have been used, which RF front-end(s) have been used, etc.			
CONTAINED IN	**Channel**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.9: Antenna Class

Class **Antenna**	
This class contains information about antenna selection.	
DERIVED FROM	
ATTRIBUTES	
CONTAINED IN	**Channel**
CONTAINS	**AntennaProfile [**1**]**
SUPPORTED EVENTS	

Table 7.10: AntennaProfile Class

Class **AntennaProfile**			
This class contains general information about this antenna.			
DERIVED FROM			
ATTRIBUTES			
antennaID	*Value type:* NameType	*Possible access:* Read	*Default value:* Not specified
This attribute describes ID of antenna.			
CONTAINED IN	**Antenna**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.11: RCConfiguration Class

Class **RCConfiguration**	
This class contains information about the current configuration of Radio Computer.	
DERIVED FROM	
ATTRIBUTES	
CONTAINED IN	**RadioComputer**
CONTAINS	**Link [*]**
SUPPORTED EVENTS	

Table 7.12: Link Class

Class **Link**	
This class contains information about one active Radio Application and corresponding connection between Reconfigurable Radio terminal and RANs.	
DERIVED FROM	
ATTRIBUTES	
CONTAINED IN	**RCConfiguration**
CONTAINS	**LinkProfile** [1], **LinkMeasurements** [1], **RFConfiguration** [1]
SUPPORTED EVENTS	

Table 7.13: LinkProfile Class

Class **LinkProfile**			
This class contains general information about this active connection.			
DERIVED FROM			
ATTRIBUTES			
linkID	*Value type:* NameType	*Possible access:* Read	*Default value:* Not specified
This attribute describes ID of link about activated connection.			
cellID	*Value type:* NameType	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes ID connected cell.			
associatedChannelID	*Value type:* OptionalObjectName	*Possible access:* Read-Add-Remove	*Default value:* Not specified
This attribute describes ID of associated channel.			
linkDirection	*Value type:* Direction	*Possible access:* Read	*Default value:* Not specified
This attribute describes a direction of link.			
carrierFrequency	*Value type:* FrequencyRange	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes a value of carrier frequency.			
nominalRFPower	*Value type:* AnaloguePower	*Possible access:* Read	*Default value:* Not specified
This attribute describes a value of nominal power.			
samplingRate	*Value type:* SamplingRate	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes a value of sampling rate.			
Bandwidth	*Value type:* Bandwidth	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes a value of bandwidth.			
CONTAINED IN	**Link**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.14: LinkMeasurements Class

Class **LinkMeasurements**			
This class contains current measurements related to this active connection.			
DERIVED FROM			
ATTRIBUTES			
receiveBLER	*Value type:* RxBLER	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes a value of BLER for received data.			
transmitPower	*Value type:* TxPower	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes a power of transmit signal.			
receiveSINR	*Value type:* RxSINR	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes a value of SINR for received data.			
transmittedBits	*Value type:* TxBits	*Possible access:* Read-Write	*Default value:* Not specified
This attribute describes transmitted bits.			
CONTAINED IN	**Link**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.15: RFConfiguration Class

Class **RFConfiguration**	
This class contains information about the configuration of RF transceiver.	
DERIVED FROM	
ATTRIBUTES	
CONTAINED IN	**Link**
CONTAINS	**TxPath [0-1], RxPath [**1**]**
SUPPORTED EVENTS	

Table 7.16: TxPath Class

Class **TxPath**			
This class describes one transmit path.			
DERIVED FROM	DERIVED FROM		
ATTRIBUTES			
txStartTime	*Value type:* TxStartTime	*Possible access:* Read-Write	*Default value:* Not specified
This attribute defines the time when the transceiver start transmission.			
txStopTime	*Value type:* TxStopTime	*Possible access:* Read-Write	*Default value:* Not specified
This attribute defines the time when the transceiver stop transmission.			
CONTAINED IN	**RFConfiguration**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.17: RxPath Class

Class **RxPath**			
This class describes one receive path.			
DERIVED FROM	DERIVED FROM		
ATTRIBUTES			
rxStartTime	*Value type:* RxStartTime	*Possible access:* Read-Write	*Default value:* Not specified
This attribute defines the time when the transceiver start reception.			
rxStopTime	*Value type:* RxStopTime	*Possible access:* Read-Write	*Default value:* Not specified
This attribute defines the time when the transceiver stop reception.			
CONTAINED IN	**RFConfiguration**		
CONTAINS			
SUPPORTED EVENTS			

Table 7.18: RCMeasurements Class

Class **RCMeasurements**			
This class contains current measurements related to Reconfigurable Radio terminal.			
DERIVED FROM			
ATTRIBUTES			
listOfActiveMeasurements	*Value type:* ActiveMeasurementsList	*Possible access:* Read-Add-Remove	*Default value:* Not specified
This attribute describes a list of active measurements.			
CONTAINED IN	**RadioComputer**		
CONTAINS			
SUPPORTED EVENTS			

8 Interface Definition

8.1 Interface Overview

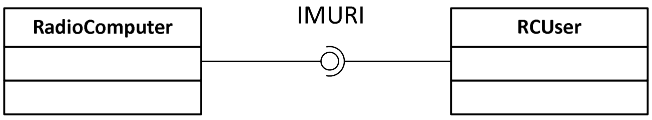

Figure 8.1: Multiradio interface (MURI)

Figure 8.1 illustrates the relationship among RadioComputer, RCUser, and MURI. As shown in figure 8.1, MURI is the provided interface to Radio Computer, while the MURI is the required interface to RCUser. Figure 8.2 illustrates a UML diagram for MURI. MURI supports 3 basic services (i.e. Administrative Services, Access Control Services, and Data Flow Services) which are further detailed in clauses 8.2, 8.3 and 8.4 respectively.

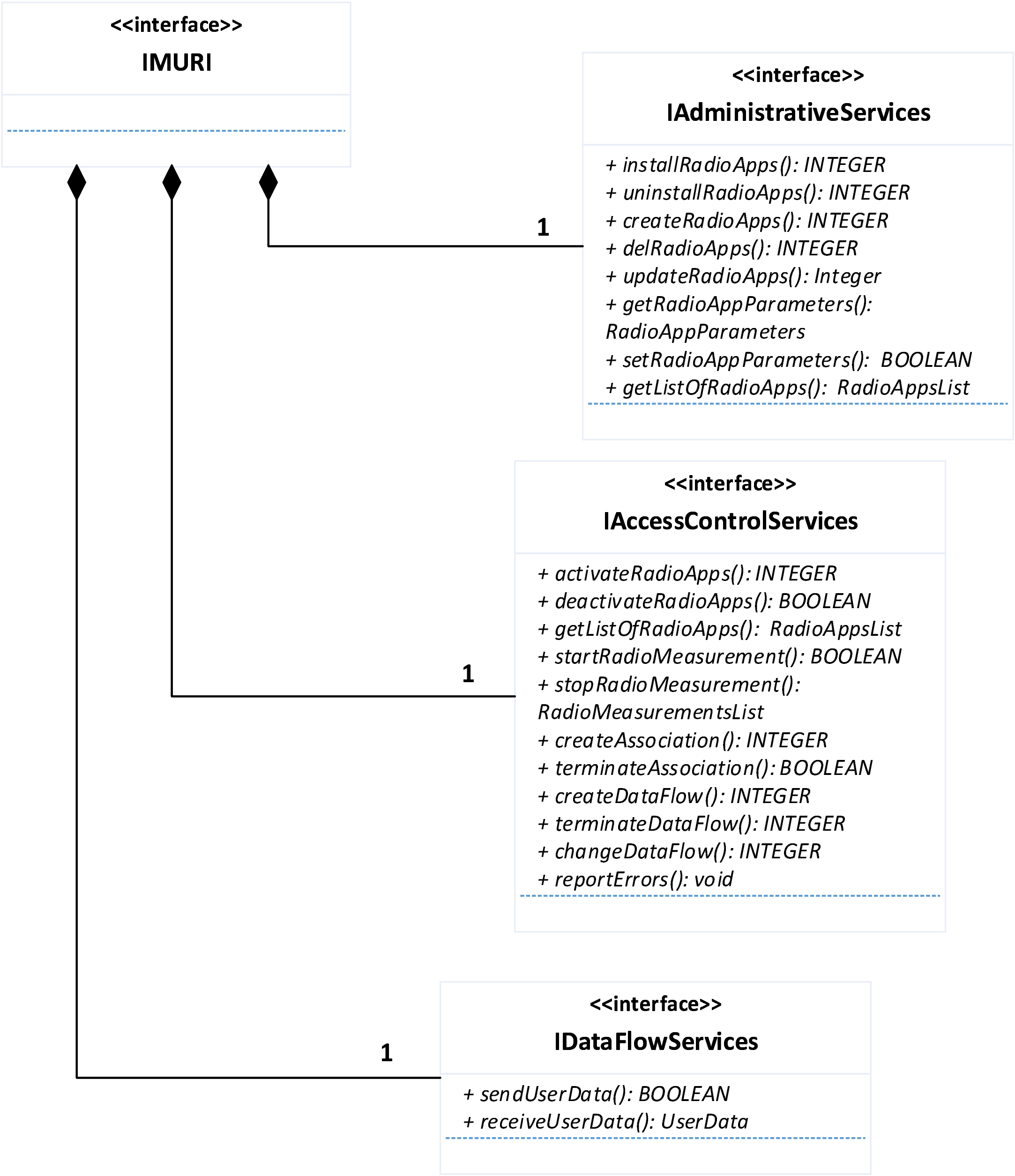
<<interface>>
IMURI
<<interface>>
IAdministrativeServices
+ installRadioApps(): INTEGER
+ uninstallRadioApps(): INTEGER
+ createRadioApps(): INTEGER
+ delRadioApps(): INTEGER
+ updateRadioApps(): Integer
+ getRadioAppParameters(): RadioAppParameters
+ setRadioAppParameters(): BOOLEAN
+ getListOfRadioApps(): RadioAppsList
1
<<interface>>
IAccessControlServices
+ activateRadioApps(): INTEGER
+ deactivateRadioApps(): BOOLEAN
+ getListOfRadioApps(): RadioAppsList
+ startRadioMeasurement(): BOOLEAN
+ stopRadioMeasurement(): RadioMeasurementsList
+ createAssociation(): INTEGER
+ terminateAssociation(): BOOLEAN
+ createDataFlow(): INTEGER
+ terminateDataFlow(): INTEGER
+ changeDataFlow(): INTEGER
+ reportErrors(): void
1
<<interface>>
IDataFlowServices
+ sendUserData(): BOOLEAN
+ receiveUserData(): UserData
1

Figure 8.2: UML diagram for MURI

8.2 Administrative Services

8.2.1 Overview on Administrative Services

Table 8.1 describes an overview on Administrative Services which are associated with Administrator. Class definition and related operations are described in clause 8.5.

Table 8.1: Overview on Administrative Services

Administrative Services	Explanation
installation/uninstallation of URA	Reconfigurable MD described in the present document sets up its configuration through software download and installation. For the support of MD reconfiguration, Administrator requests installation/uninstallation of URA to CM.
creating instance of URA	For activating the installed URA(s), instance(s) of the corresponding URA(s) has (have) to be created. Administrator requests CM to create instance(s) of the corresponding URA(s). CM creates the instance(s) and returns the confirmation of the instance creation to Administrator, when CM judges the instance(s) creation is valid.
deleting instance of URA	Since the instance(s) of instantiated URA(s) occupies (occupy) the memory resources of the Reconfigurable MD, it might be desired to delete URA(s) instance(s) that is (are) not needed. Administrator requests CM to delete instance(s) of such URA(s).
getting/configuring URA Parameters	Administrator requests CM to provide parameter(s) of URA(s) such as required computational/spectral resources, antenna ports, etc. in order to manage the created URA(s) instance(s).
Updating instance of URA	Instance(s) of (an) installed URA(s) is/are replaced, typically by a SW Update. During the replacement process, the configuration is maintained.
URA List	In reconfigurable MD, the status of URA(s) might be installed, instantiated, or activated. In order to manage each URA properly, Administrator requests CM to provide URA list which includes ID and name as well as the status of each URA.

8.2.2 Messages for Administrative Services

The interfaces for Administrative Services are used to transmit the following messages:

- From CSL to RCF:
  - Request of installation/uninstallation of an URA
  - Request of creating/deleting an instance of an URA
  - Request of updating an instance of an URA
  - Request of getting/configuring parameters of an URA
  - Request of installed/instantiated/activated URA(s) list
- From RCF to CSL:
  - Confirmation of installation/uninstallation of URA
  - Confirmation of the creation/deletion of a URA instance
  - Confirmation of the updating of a URA instance
  - Failure of URA installation/uninstallation
  - Failure of the creation/deletion of an URA instance
  - Failure of the updating of a URA instance
  - Information of URA parameters
  - URA(s) list retrieving

8.3 Access Control Services

8.3.1 Overview on Access Control Services

Table 8.2 describes an overview on Access Control Services which are associated with MPM. Class definition and related operations are described in clause 8.5.

Table 8.2: Overview on Access Control Services

Access Control Services	Explanation
URA List	When the MPM of reconfigurable MD activates URA(s) and/or creates association of URA(s), MPM needs to know the URA list. For this purpose, MPM requests CM to provide the URA list. Upon receiving the URA list request from MPM, CM returns the URA list which includes ID and name as well as the status of each URA.
Activation/deactivation of URAs	MPM requests RCM to activate/deactivate URA(s) depending upon the contents of the URA list provided from CM. RCM performs the activation/deactivation of the URA(s) and acknowledges the confirmation of the request.
Radio environments measurement	URA may perform a certain level of measurements autonomously or upon a trigger. In the second case, MPM requests RCM to start radio environment measurements. RCM request URA to start the radio environment measurements and acknowledges the success of start radio environment measurements. In the case of stopping radio environment measurements, MPM requests RCM to stop radio environment measurements. Then, RCM requests URA to stop radio environment measurements and sends the measurement information.
Creation/termination of associations	MPM requests RCM to create/terminate association of URA(s) because activated URA(s) set(s) up the association.
Creation/termination of data flows into/from associations	MPM requests RCM to create/terminate data flow(s) into/from association(s). When the created data flow is terminated, RCM acknowledges the termination of the data flow to MPM.
Flexible Data flow	In some communication environments such as Vertical Handover, the data flow of one association may have to be moved to another association or partitioned into many associations. In some other cases, the data flow of many associations may have to be combined into a single association. In those instances, MPM requests RCM to move/partition/combine of the data flow.
Errors reporting	During the procedure of handling multi-RAT in reconfigurable MD, various kinds of errors may take place in RCF. When the error occurs, CM has to report it to MPM. Example: In the case of spectral resource collision, MRC informs of this error to CM, which reports the error to MPM.

8.3.2 Messages for Access Control Services

The interfaces for Access Control Services are used to transmit the following messages:

- From CSL to RCF:
  - Request of activation/deactivation of an URA
  - Request of update of an URA
  - Request of installed/instantiated/activated URA(s) list
  - Request of start/stop measurements for radio environment
  - Request of measurements for MD capabilities
  - Request for the creation of a data flow
  - Request for the termination of a data flow
  - Request for the creation of a network and logical radio link association
  - Request for changing a data flow

- From RCF to CSL:
    - Confirmation of an URA activation/deactivation
    - Confirmation of an URA update
    - Confirmation of data flow creation
    - Confirmation of data flow termination
    - Confirmation of the creation of a network and logical radio link association
    - Confirmation of changing a data flow
    - Confirmation of starting radio environment measurements
    - Failure of an URA activation/deactivation
    - Failure of data flow creation
    - Failure of data flow termination
    - Failure of the creation of a network and logical association
    - Failure of changing a data flow
    - Failure of starting radio environment measurements
    - Failure of an URA update
    - URA(s) list retrieving
    - Information related to the radio environment
    - Information about MD capabilities
    - Information about errors

8.4 Data Flow Services

8.4.1 Overview on Data Flow Services

Table 8.3 describes an overview on Data Flow Services which are associated with networking stack. Class definition and related operations are described in clause 8.5.

Table 8.3: Overview on Data Flow Services

Data Flow Services	Explanation
Sending User Data	In order to transmit user data through a particular data flow among multiple data flows, Networking stack requests FC to perform the transmission of the user data. Then FC acknowledges the confirmation of the transmission to Networking stack.
Receiving User Data	When receiving user data through multiple data flows, FC transfers the received user data together with the data flow ID to the Networking stack.

8.4.2 Messages for Data Flow Services

Interfaces for Data Flow Services are used to transmit the following messages:

- From CSL to RCF:
    - Request of user data transfer (transmission: SendUserData, reception: ReceiveUserData)

- From RCF to CSL:
    - Confirmation of user data transfer
    - Failure of user data transfer
    - User data

8.5 Class Definitions for Interface

Each interface class related to MURI can be defined using the template presented in clause 6.2 and in accordance with the UML diagram of figure 8.2 which specifies the interface classes related to MURI. Tables 8.4 to 8.6 specify all the operations related to the three interface classes above described.

Table 8.4: IAdministrativeServices Class

*Class***IAdministrativeServices**		
This class describes interfaces supporting Administrative Services.		
OPERATIONS		
installRadioApps	*Return type:* INTEGER	*Value type:* public
This operation is related to the installation of an URA.		
uninstallRadioApps	*Return type:* INTEGER	*Value type:* public
This operation is related to the uninstallation of an URA		
updateRadioApps	Return type: INTEGER	Value type: public
This operation is related to the update of an URA		
createRadioApps	*Return type:* INTEGER	*Value type:* public
This operation is related to the creation of an instance of an URA.		
delRadioApps	*Return type:* INTEGER	*Value type:* public
This operation is related to the deletion of an instance of an URA.		
getRadioAppParameters	*Return type:* RadioAppParameters	*Value type:* public
This operation is needed for retrieving URA parameters.		
setRadioAppParameters	*Return type:* BOOLEAN	*Value type:* public
This operation is needed for setting URA parameters.		
getListOfRadioApps	*Return type:* RadioAppsList	*Value type:* public
This operation is needed for getting a list of the installed/instantiated/activated URA(s).		

Table 8.5: IAccessControlServices Class

*Class***IAccessControlServices**		
This class describes interfaces supporting Access Control Services.		
OPERATIONS		
activateRadioApps	*Return type:* INTEGER	*Value type:* public
This operation is needed for activating a URA.		
deactivateRadioApps	*Return type:* BOOLEAN	*Value type:* public
This operation is needed for deactivating a URA.		
getListOfRadioApps	*Return type:* RadioAppsList	*Value type:* public
This operation is needed for getting a list of the installed/instantiated/activated URA(s).		
startRadioMeasurement	*Return type:* BOOLEAN	*Value type:* public
This operation starts the measurements related to radio environments and MD capabilities.		
stopRadioMeasurement	*Return type:* RadioMeasurementsList	*Value type:* public
This operation stops the measurements related to radio environments and MD capabilities.		
createAssociation	*Return type:* INTEGER	*Value type:* public
This operation is related to the creation of a network association.		
terminateAssociation	*Return type:* BOOLEAN	*Value type:* public
This operation terminates a network association previously created.		
createDataFlow	*Return type:* INTEGER	*Value type:* Public
This operation creates a data flow.		
terminateDataFlow	*Return type:* INTEGER	*Value type:* public
This operation terminates a data flow.		
changeDataFlow	*Return type:* INTEGER	*Value type:* Public
This operation move/separate/combine data flow.		
reportErrors	*Return type:* Void	*Value type:* public
This operation is needed for reporting errors.		

Table 8.6: IDataFlowServices Class

*Class***IDataFlowServices**		
This class describes interfaces supporting Data Flow Services.		
OPERATIONS		
sendUserData	*Return type:* BOOLEAN	*Value type:* public
This operation is needed for sending user data.		
receiveUserData	*Return type:* UserData	*Value type:* public
This operation is needed for receiving user data.		

Annex A (informative): Abstract Data Definitions

The following ASN.1 in Recommendation ITU-T X.680 [i.6] module contains all necessary abstract data definitions used in the attribute definitions in clause 7.2 and clause 8.5.

```
ETSI-EN-303-146-1-Type-Definitions DEFINITIONS ::= BEGIN
            ------------------------------------------------------------------
            ------------------------------------------------------------------
            -- START Common Data Types

            ------------------------------------------------------------------
            -- START Name Related Data Types

            NameType    ::= CHOICE      {
                    number      INTEGER,
                    string      PrintableString
            }

            ObjectName      ::= SEQUENCE OF NameType


            OptionalObjectName      ::= CHOICE  {
                    id          ObjectName,
                    void        NULL
            }

            ObjectNameList  ::= SEQUENCE OF ObjectName


            -- END Name Related Data Types
            ------------------------------------------------------------------

            ------------------------------------------------------------------
            -- START Version Related Data Types

            Version     ::= CHOICE  {
                intVersion          INTEGER,
                stringVersion       PrintableString
            }

            -- END Version Related Data Types
            ------------------------------------------------------------------

            -- END Common Data Types
            ------------------------------------------------------------------
            ------------------------------------------------------------------



            ------------------------------------------------------------------
            ------------------------------------------------------------------
            -- START Radio Application Related Data Types

            RAOptionID      ::= ENUMERATED      {
                lte5Mhz,lte10Mhz, lte20Mhz, ...
            }

            RAOptionsList       ::= SEQUENCE OF SEQUENCE        {
                rAOptionName        RAOptionID,
                rAOptionValue       ANY
            }

            --  END Radio Application Related Data Types
            ------------------------------------------------------------------
            ------------------------------------------------------------------



            ------------------------------------------------------------------
            ------------------------------------------------------------------
            -- START Radio Computer Related Data Types

            RadioApplicationIDList      ::= SEQUENCE OF OptionalObjectName
```

```
RCOptionID      ::= ENUMERATED      {
    mdrc-0, mdrc-1, mdrc-2, maximumTxPower, ...
}

RCOptionsList       ::= SEQUENCE OF SEQUENCE        {
    rCOptionName        RCOptionID,
    rCOptionValue       ANY
}


RadioInterfaceID        ::= ENUMERATED      {
    umts, hsdpa, wimax, lte, wifi, gsm, ...
}

RadioInterface      ::= CHOICE  {
    id          RadioInterfaceID,
    void        NULL
}

RadioInterfacesList     ::= SEQUENCE OF RadioInterfaceID


ChannelIDsList      ::= SEQUENCE OF OptionalObjectName


HardwareInfo        ::= ENUMERATED      {
    fixedPipeline, programmablePipeline, hybridPipeline, ...
}

SoftwareInfo        ::= ENUMERATED      {
    rOSVersion, compiler, ...
}


Direction       ::= ENUMERATED      {
    downlink, uplink
}


RxBLER      ::= SEQUENCE    {
    accBLER     REAL,
    period      REAL OPTIONAL,
    instBLER    REAL OPTIONAL
}


TxPower     ::=     SEQUENCE    {
    power       REAL,
    unit        CHARACTER
}


RxSINR      ::= SEQUENCE    {
    accSINR     REAL,
    period      REAL OPTIONAL,
    instSINR    REAL OPTIONAL
}


TxBits      ::= SEQUENCE    {
    transmittedBit      REAL,
    unit                        CHARACTER
}


ActiveMeasurementID ::= ENUMERATED      {
    transmitPower, transportLoad, processingLoad, ...
}

ActiveMeasurementIDs        ::= SEQUENCE OF {
    activeMeasurementID
}

ActiveMeasurementsList      ::=     SEQUENCE OF SEQUENCE    {
    activeMeasurementName       ActiveMeasurementID,
    activeMeasurementValue      ANY
}
```

```
FrequencyRange      ::= SEQUENCE    {
    centralFrequency    REAL,
    frequencyBand           REAL
}

AnaloguePower       ::=     SEQUENCE    {
    power       REAL,
    unit        CHARACTER
}

SamplingRate        ::=     SEQUENCE    {
    samplingRate        REAL,
    unit        CHARACTER
}

Bandwidth       ::=     SEQUENCE    {
    bandWidth       REAL,
    unit        CHARACTERs
}

TxStartTime         ::=         CHOICE  {
    absoluteTime    GeneralizedTime,
    relativeTime    INTEGER
}

TxStopTime          ::=         CHOICE  {
    Undefined       NULL,
    absoluteTime    GeneralizedTime,
    relativeTime    INTEGER
}


RxStartTime         ::=         CHOICE  {
    absoluteTime    GeneralizedTime,
    relativeTime    INTEGER
}


RxStopTime          ::=         CHOICE  {
    Undefined       NULL,
    absoluteTime    GeneralizedTime,
    relativeTime    INTEGER
}


ChannelMeasurementID        ::= ENUMERATED  {
    channelInterference, channelLoad, ...
}

ChannelMeasurementsList     ::= SEQUENCE OF SEQUENCE    {
    channelMeasurementName      ChannelMeasurementID,
    channelMeasurementValue     ANY
}

ConfigurationMeasurements::=    ENUMERATED  {
    antennaProt, RFfrontend, ...
}


-- END Radio Computer Related Data Types
------------------------------------------------------------------
------------------------------------------------------------------


------------------------------------------------------------------
------------------------------------------------------------------
-- START Multiradio Interface Related Data Types

RadioAppParameterID ::= ENUMERATED {
    A, b, c, ...
}

RadioAppParameters      ::= SEQUENCE OF SEQUENCE    {
    radioAppParameterName   RadioAppParameterID,
    radioAppParameterValue  ANY
}
```

```
RadioAppsList       ::= SEQUENCE OF SEQUENCE    {
    RadioAppID          INTEGER,
    RadioAppName        PrintableString
}


RadioMeasurementID      ::= ENUMERATED {
    A, B, C, ...
}

RadioMeasurementsList   ::= SEQUENCE OF SEQUENCE {
    radioMeasurementName        RadioMeasurementID,
    radioMeasurementValue   ANY
}


UserData        ::= SEQUENCE OF {
    userDataID          INTEGER,
    userDataValue       OBJECT
}


-- END Multiradio Interface Related Data Types
-------------------------------------------------------------------
-------------------------------------------------------------------
```

Annex B (informative): MURI Qualification Methods for Validation

The MURI requirements are basis for qualification methods to validate that the requirements can be met. A feature list exposing MURI capabilities is created. Qualification methods correspond to the feature list and they qualify features of a particular MURI implementation against the feature list.

The following qualification methods might be typically applied:

- Demonstration - The operation of interfacing entities that rely on observable functional operation.
- Test - The operation of interfacing entities using specialist test equipment to collect data for analysis.
- Analysis - The processing of data obtained from methods, such as reduction, interpretation, or extrapolation of test results.
- Inspection - The visual examination of interfacing entities, documentation, etc.
- Special qualification methods - Methods for the interfacing entities, such as specialist tools, techniques, procedures, facilities, etc.

History

Document history		
V1.1.1	November 2013	Publication as ETSI TS 103 146-1
V1.2.1	November 2015	Publication
V1.2.7	March 2018	EN Approval Procedure AP 20180605: 2018-03-07 to 2018-06-05
V1.3.1	June 2018	Publication