This deliverable includes the initial software documentation of the HRADIO Client libraries. It describes the Android implementation of an easy to use application layer interface for radio applications, as well as a time-shift player component which will be used by a number of HRADIO scenarios. It also introduces the setup of the development environment, in order to enable a quick adoption of the client libraries in the project.
### Basic Information

<table>
<thead>
<tr>
<th>Basic Information</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work package</td>
<td>3</td>
</tr>
<tr>
<td>Due date</td>
<td>31/07/2018</td>
</tr>
<tr>
<td>Submission date</td>
<td>30/08/2018</td>
</tr>
<tr>
<td>Deliverable lead</td>
<td>Institut für Rundfunktechnik, IRT</td>
</tr>
<tr>
<td>Version</td>
<td>1.0</td>
</tr>
<tr>
<td>Authors</td>
<td>Alexander Erk (IRT)</td>
</tr>
<tr>
<td>Reviewers</td>
<td>Klaas Baert (VRT), Maximilian Knoop (Konsole)</td>
</tr>
</tbody>
</table>

### Document Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description of change</th>
<th>List of contributor(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>V0.1</td>
<td>19/07/2018</td>
<td>Initial version</td>
<td>Alexander Erk (IRT)</td>
</tr>
<tr>
<td>V0.2</td>
<td>20/07/2018</td>
<td>Updates from review</td>
<td>Alexander Erk (IRT)</td>
</tr>
<tr>
<td>V0.9</td>
<td>27/07/2018</td>
<td>Word editing</td>
<td>Alexander Erk (IRT)</td>
</tr>
<tr>
<td>V1.0</td>
<td>30/07/2018</td>
<td>Coordinator review</td>
<td>Simon Delaere (imec)</td>
</tr>
</tbody>
</table>
D3.3: HRADIO mobile and HTML client API implementations

Disclaimer

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 731677. This document reflects only the authors’ views and the Commission is not responsible for any use that may be made of the information it contains.

<table>
<thead>
<tr>
<th>Project co-funded by the European Commission in the H2020 Programme</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Nature of the deliverable:</strong></td>
</tr>
<tr>
<td>PU</td>
</tr>
<tr>
<td>CL</td>
</tr>
<tr>
<td>CO</td>
</tr>
</tbody>
</table>
EXECUTIVE SUMMARY

This document describes the 1st release of the WP3 HRADIO client libraries. Main focus of this initial version is the delivery of an OMRI implementation for the first pilot phase in HRADIO. According to the envisaged full HRADIO technical architecture, a working Android implementation of OMRI forms the foundation for later HRADIO developments, as components and modules developed using this release can be reused and connected with future WP3 developments in the metadata and playout tasks.

The provided implementation is based on the work of IRT in the area of the OMRI initiative from IDAG and WorldDAB. At the start of the HRADIO project, OMRI was only an API proposal with the idea to provide an easy to use application layer interface for radio applications. For this deliverable the work done in T3.3 was to provide a working implementation for OMRI.

The second component described in this deliverable is the development of an OMRI based time-shift player for Android devices. This time-shift player component can easily be used by the HRADIO project partners when developing the pilot applications. It fully takes care of the handling of time-shifted radio signals, the necessary buffers on the device and the accompanying metadata.

Finally, this deliverable introduces the setup of the development environment which focuses on an easy and reproducible setup, in order to enable a quick adoption of the client libraries in the project.
# TABLE OF CONTENTS

1. **INTRODUCTION** ................................................................................................................. 10
2. **OMRI OPEN MOBILE RADIO INTERFACE** ......................................................................... 12
   2.1. **SYSTEM OVERVIEW** .................................................................................................... 12
   2.2. **API OVERVIEW** ........................................................................................................... 14
   2.2.1. **PACKAGES** .............................................................................................................. 15
   2.2.2. **OBJECT DIAGRAM** ................................................................................................. 15
   2.2.3. **RADIO STATE MODEL** ............................................................................................. 16
   2.2.4. **TUNER STATE MODEL** ............................................................................................. 17
3. **TIMESHIFT PLAYER COMPONENT** .................................................................................... 20
   3.1. **CLASSES (PUBLIC METHODS ONLY):** ........................................................................ 20
   3.1.1. **TIMESHIFTPLAYER.JAVA** ...................................................................................... 20
   3.1.2. **SKIPITEM.JAVA** ..................................................................................................... 21
3.2. **INTERFACES** .................................................................................................................. 22
   3.2.1. **TIMESHIFTLISTENER.JAVA** ................................................................................ 22
4. **SOURCE CODE REPOSITORIES AND DEVELOPMENT SETUP FOR ANDROID** ........ 23
   4.1. **REPOSITORIES** ........................................................................................................... 23
   4.2. **IDE ANDROID STUDIO** ............................................................................................... 24
   4.3. **LIBRARY MANAGEMENT** ............................................................................................ 25
   4.4. **SIMPLE DEVELOPER WORKFLOW EXAMPLE** .......................................................... 26
5. **CONCLUSIONS** .................................................................................................................... 28

## APPENDIX A

A.1 **PACKAGE ORG.OMRI.RADIO** ......................................................................................... 29
   A.1.1 **RADIO** ..................................................................................................................... 29
   A.1.2 **RADIOERRORCODE** ............................................................................................... 30
   A.1.3 **RADIOLISTENER** ...................................................................................................... 31
   A.1.4 **RADIOSERVICE** ...................................................................................................... 31
   A.2 **PACKAGE ORG.OMRI.RADIOSERVICE** ...................................................................... 32
   A.2.1 **RADIOSERVICE** ...................................................................................................... 32
   A.2.2 **RADIOSERVICEAUDIODATALISTENER** ................................................................. 33
<table>
<thead>
<tr>
<th>A.2.3</th>
<th>RADIOSERVICEDAB</th>
<th>34</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.2.4</td>
<td>RADIOSERVICEDABCOMPONENT</td>
<td>35</td>
</tr>
<tr>
<td>A.2.5</td>
<td>RADIOSERVICEDABCOMPONENTLISTENER</td>
<td>36</td>
</tr>
<tr>
<td>A.2.6</td>
<td>RADIOSERVICEDABUSERAPPLICATION</td>
<td>37</td>
</tr>
<tr>
<td>A.2.7</td>
<td>RADIOSERVICEFM</td>
<td>37</td>
</tr>
<tr>
<td>A.2.8</td>
<td>RADIOSERVICEFEMPTY</td>
<td>38</td>
</tr>
<tr>
<td>A.2.9</td>
<td>RADIOSERVICEIP</td>
<td>39</td>
</tr>
<tr>
<td>A.2.10</td>
<td>RADIOSERVICEIPSTREAM</td>
<td>39</td>
</tr>
<tr>
<td>A.2.11</td>
<td>RADIOSERVICELISTENER</td>
<td>40</td>
</tr>
<tr>
<td>A.2.12</td>
<td>RADIOSERVICEMIMETYPE</td>
<td>40</td>
</tr>
<tr>
<td>A.2.13</td>
<td>RADIOSERVICETYPE</td>
<td>41</td>
</tr>
<tr>
<td>A.3</td>
<td>PACKAGE ORG.OMRI.RADIOSERVICE.METADATA</td>
<td>41</td>
</tr>
<tr>
<td>A.3.1</td>
<td>GROUP</td>
<td>41</td>
</tr>
<tr>
<td>A.3.2</td>
<td>LOCATION</td>
<td>42</td>
</tr>
<tr>
<td>A.3.3</td>
<td>PROGRAMMEINFORMATION</td>
<td>43</td>
</tr>
<tr>
<td>A.3.4</td>
<td>PROGRAMMEINFORMATIONTYPE</td>
<td>43</td>
</tr>
<tr>
<td>A.3.5</td>
<td>PROGRAMMESERVICEMETADATALISTENER</td>
<td>44</td>
</tr>
<tr>
<td>A.3.6</td>
<td>SERVICEINFORMATION</td>
<td>44</td>
</tr>
<tr>
<td>A.3.7</td>
<td>SPIPROGRAMMEINFORMATION</td>
<td>44</td>
</tr>
<tr>
<td>A.3.8</td>
<td>TERMD</td>
<td>45</td>
</tr>
<tr>
<td>A.3.9</td>
<td>TEXTUAL</td>
<td>45</td>
</tr>
<tr>
<td>A.3.10</td>
<td>TEXTUALDADB DYNAMICLABEL</td>
<td>46</td>
</tr>
<tr>
<td>A.3.11</td>
<td>TEXTUALDADB DYNAMICLABELPLUSCONTENTTYPE</td>
<td>46</td>
</tr>
<tr>
<td>A.3.12</td>
<td>TEXTUALDADB DYNAMICLABELPLUSITEM</td>
<td>48</td>
</tr>
<tr>
<td>A.3.13</td>
<td>TEXTUALFMRDSRADIOTEXT</td>
<td>48</td>
</tr>
<tr>
<td>A.3.14</td>
<td>TEXTUALIPRDNSSRADIOVIS</td>
<td>49</td>
</tr>
<tr>
<td>A.3.15</td>
<td>TEXTUALMETADATALISTENER</td>
<td>49</td>
</tr>
<tr>
<td>A.3.16</td>
<td>TEXTUALTYPE</td>
<td>50</td>
</tr>
<tr>
<td>A.3.17</td>
<td>VISUAL</td>
<td>50</td>
</tr>
<tr>
<td>A.3.18</td>
<td>VISUALDABSLIDESHOW</td>
<td>51</td>
</tr>
</tbody>
</table>
**A.3.19 VISUALIPRDNSRADIOVIS** ......................................................................................................................... 52
**A.3.20 VISUALMETADATALISTENER** ...................................................................................................................... 52
**A.3.21 VISUALMIMETYPE** ........................................................................................................................................ 53
**A.3.22 VISUALTYPE** .................................................................................................................................................. 53
**A.4 PACKAGE PACKAGE ORG.OMRI.TUNER** .............................................................................................................. 54
**A.4.1 TUNER** ............................................................................................................................................................... 54
**A.4.2 TUNERLISTENER** ............................................................................................................................................... 55
**A.4.3 TUNERSTATUS** .................................................................................................................................................. 56
**A.4.4 TUNERTYPE** ....................................................................................................................................................... 57
LIST OF FIGURES

FIGURE 1: HRADIO CLIENT LIBRARY ARCHITECTURE .......................................................... 10
FIGURE 2: RADIODNS HYBRID RADIO ............................................................................. 13
FIGURE 3: OMRI SOFTWARE STACK ................................................................................. 14
FIGURE 4: OMRI OBJECT DIAGRAM ............................................................................... 16
FIGURE 5: RADIO STATE MODEL ..................................................................................... 17
FIGURE 6: TUNER STATE MODEL ..................................................................................... 18
FIGURE 7: DEVELOPMENT WORKFLOW .......................................................................... 23
FIGURE 8: HRADIO GROUP LANDING PAGE ON GITLAB-EXT.IRT.DE ..................... 24
FIGURE 9: ANDROID STUDIO WITH TIMESHIFTPLAYER EXAMPLE ................................ 25
FIGURE 10: ARTIFACTORY FOR HRADIO ....................................................................... 26
FIGURE 11: SUCCESSFUL APK BUILD FOR “TIMESHIFTSAMPLEAPP” ......................... 27
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>DAB</td>
<td>Digital Audio Broadcasting</td>
</tr>
<tr>
<td>DAB-DL</td>
<td>DAB Dynamic Label</td>
</tr>
<tr>
<td>DAB-SPI</td>
<td>DAB Service and Programme Information</td>
</tr>
<tr>
<td>DAB-SLS</td>
<td>DAB Slideshow</td>
</tr>
<tr>
<td>DNS</td>
<td>Dynamic Name Server</td>
</tr>
<tr>
<td>EPG</td>
<td>Electronic Programme Guide</td>
</tr>
<tr>
<td>FM</td>
<td>Frequency modulation</td>
</tr>
<tr>
<td>HTML</td>
<td>Hyper Text Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>IDAG</td>
<td>International DMB Advancement Group</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>PI</td>
<td>Programme Information</td>
</tr>
<tr>
<td>MOT</td>
<td>Multimedia Object Transfer</td>
</tr>
<tr>
<td>SI</td>
<td>Service Information</td>
</tr>
<tr>
<td>SPI</td>
<td>Service and Programme Information</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

In deliverable D2.3 the selected HRADIO user scenarios have been analysed and technical system requirements have been derived. In these system requirements DAB/Radio centric technologies (e.g. DAB-DL+, DAB-SPI, DAB-SLS) play an important role and will be an absolute critical factor in successful pilot development. Therefore WP3 decided to concentrate its first effort on the implementation of a comprehensive set of the OMRI API for the widely used Android system.

![HRADIO Client Library Architecture](image)

In March 2016 LG introduced a Smartphone with built-in DAB+ receiver for the European and Australian market. Smartphones with built-in radio receivers (mostly FM) are no novelty, but a new feature of the LG presentation was that for the first time application developers were granted access to the hardware through a proprietary but publicly available API.

In order to enable the development of cross-system applications, independent of manufacturers or different tuner concepts, the IDAG (International DMB Advancement Group) decided to develop and standardize an API precisely for this purpose. The result of these efforts is OMRI (Open Mobile Radio Interface), which was first designed by IDAG members (IRT, Fraunhofer, BBC...) and will then be handed over for standardisation in WorldDAB.

One of OMRI’s main objectives was independence from transmission technologies. Whether the tuner to be used uses DAB, FM, HD radio or even AM as radio reception technology is of no importance in OMRI for the application developer. Although
initially defined as Java API, care was taken in the design phase to ensure that equivalent API functions can also be implemented in C++/Swift for the iOS platform and JavaScript for common HTML5 browsers.

At the start of the HRADIO project, OMRI was only an API proposal with the idea to provide an easy to use application layer interface for radio applications. One of the development goals in work package 3 of the HRADIO project is the implementation of this OMRI API for the intended platforms. Based on the existing C++ DAB library developments of the project partner IRT, a Java JNI layer was implemented to make the functions for demultiplexing and decoding of digital DAB+ services realized in C++ accessible from the Java layer in Android.

Based on the OMRI API, a player component was also developed to temporarily store radio services and their data components and play them back at different times – i.e. a timeshift component. To make it easier for application developers to get started with the libraries and components, sample code was written and a Maven repository (Artifactory) was set up to deploy the OMRI implementation and the player components.
2. OMRI OPEN MOBILE RADIO INTERFACE

2.1. SYSTEM OVERVIEW

Normally mobile devices have the capability to connect to IP based networks either over integrated WiFi or cellular communications systems (3G, 4G). This connectivity however only allows for point to point connectivity. Access to broadcast services such as DVB for TV or DAB/FM for Radio is often not possible. Even if mobile devices are equipped with broadcast receivers, application developers do not have access to the hardware tuner resources and therefore they cannot enhance their media centric applications with access to broadcast services.

Technologies such as IP audio streaming and Podcasts have enabled service offerings for on-demand experiences in radio consumption. Specifications such as RadioDNS [ref] will allow a combination of broadcast and IP based services today known as Hybrid Radio. In order to utilize this potential, it is important to combine broadcast media with individual accessed on-demand content in a seamless User eXperience (UX).

The following drawing depicts a system overview of mobile devices accessing Radio and hybrid services using RadioDNS as the “pathfinder” between them.
While application developers understand very well how to access and implement IP based services on target mobile platforms, such an easy access does not exist for broadcast media. This OMRI API specification closes this gap, by providing a standardized, technology agnostic API for application developers to develop hybrid radio applications.
The receiver product receives broadcast data via the tuner hardware which includes the audio services as well as additional metadata such as MOT-Slideshow, DL/DL+, MOT-EPG and other information. The tuner hardware usually is integrated into the device’s OS via a driver software usually provided by the manufacturer of the tuner hardware itself. A middleware software layer uses the data provided by the driver software to perform a range of tasks to extract the audio and metadata for presentation to a User Application (the App). Such tasks can include demodulation, demultiplexing and decoding of the different service components. Currently different manufacturers provide custom APIs for access to their tuner hardware and middleware software and consequently the user application must conform to individual tuner solutions. The OMRI API standardizes the access to tuner solutions and enables the development of comprehensive radio applications.

2.2. API OVERVIEW

The following section gives an overview of the OMRI API. Additionally the core Classes state models (org.omri.radio and org.omri.tuner) are described. For a detailed analysis, the complete definition of the API can be found in Annex A of this document.
2.2.1. Packages

The OMRI API currently consists of three different packages:

org.omri.radio:

The radio package acts as the entry point into the API for the developer. The main class in org.omri.radio is the Radio class which is designed as a singleton and provides a simple getInstance method for the application developer to obtain the Radio instance for further usage. Additionally in the org.omri.radio package Enums for error and status codes are defined. The access to broadcast data is highly asynchronous, therefore the org.omri.radio package defines the base class of all further interfaces of listeners in the OMRI API.

org.omri.radioservice:

The org.omri.radioservice package contains all the necessary definitions for applications developers to access radio service information such as Service labels, descriptions, logos, and many more. While the general radio service model in OMRI is agnostic to the underlying broadcasting technology, the org.omri.radioservice package contains the necessary sub-interfaces derived from the org.omri.radioservice. The RadioService interface reveals broadcast system specific information and metadata to the developer. Derived from the RadioListener interface, org.omri.radioservice and its sub-package metadata defines specific listener interface definitions for service data components such as DynamicLabel/DynamicLabel+, visuals (Slideshow) and programme information.

org.omri.tuner:

The org.omri.tuner package defines the abstract Tuner interface which enables the developer to access radio functionalities such as service scan. The OMRI API is designed to be able to handle devices which include multiple tuners even for different transmission technologies (e.g. DAB, FM and IP). The same package contains the TunerListener interface which is used to deliver highly asynchronous information (e.g. service scan status, signal levels).

2.2.2. Object Diagram

The following drawing shows an example OMRI instance diagram. Beginning with the singleton instance of the Radio class, Radio.getAvailableTuners returns a list of Tuner instances. Querying the TunerStatus reveals that one of the tuners is in TUNER_STATUS_INITIALIZED state and can return an instance of RadioServiceDab. Subscribed to the RadioServiceDab instance are two RadioServiceListener subclasses receiving events from the arrival of new Visual and Textual metadata.
2.2.3. Radio State Model

The following drawing depicts the state model of the Radio class. The Radio can have three different states:
Figure 5: Radio State Model

**not initialized:** This is the initial state of the Radio class. This means that when the OMRI application is started and a Radio instance is obtained through the getInstance() method the call to getRadioStatus() returns STATUS_RADIO_UNINITIALIZED. In this state, calls to getAvailableTuners() and/or getRadioServices() will return empty lists. Therefore also no Tuners can be initialized nor RadioServices can be started.

**running:** Calling one of the initialize() or resume() methods brings the Radio object into running status. If an ERROR_INIT_OK or ERROR_RESUME_OK is returned, the Radio object is in running state and ready for tuner initialisation and/or service selections.

**suspended:** Calling suspend() on a running Radio object brings it into suspended status. All activities by the Radio class will be suspended until its status changes back to running.

### 2.2.4. Tuner State Model

The following drawing depicts the state model of the Tuner object.
**not initialized**: This is the initial state of the Tuner object.

**initialized**: Calling the `initializeTuner()` or `resumeTuner()` methods brings the Tuner object into initialized status. When in the initialized state a service scans or service selection can be performed. When the tuner is scanning, the `stopServiceScan()` method can be used to terminate the scan and return to the initialized state.

**suspended**: Calling `suspendTuner()` on a running Tuner object brings it into suspended status. All activities by the Tuner will be suspended until its status changes back to initialized.

**scanning**: Calling the `startServiceScan()` method while in the initialize state will start a service scan on the Tuner. When finished the Tuner goes back into initialized state automatically. While in scanning state, the method `getCurrentRunningRadioService` will return null.

**error**: For many reasons the device or underlying driver software can cause a Tuner to go into error status. By calling the `deInitializeTuner()` method, the developer can try to bring the Tuner back into a defined initial state. However if the error cause is
still valid the Tuner can go instantly into an error state again. Tuner implementations shall provide a meaningful status description in the newStatus parameter when calling the tunerStatusChanged() method.
3. TIMESHIFT PLAYER COMPONENT

The time-delayed consumption of media is a widely used service in the TV sector. TV devices with hard disk storage allow a local buffer on the end device. Technologies such as HbbTV also provide mechanisms to enable network-based TimeShift implementations.

Time shift also makes sense in the radio sector. In deliverable D2.3 the technical requirements of the HRADIO use cases were examined and it was determined that time-shift components play a central role for many of the selected HRADIO use cases. Six of the scenarios required the availability of a local time shift player, in order to pause the live radio program.

Therefore, the development of a reusable local time shift player component has become a high priority in the early phase of T3.3.

For the Application developers the time-shift player API defines two classes and one interface for its functionality. Basically the implementation uses an android.media.AudioTrack object to play back the audio on the device and maintains a local buffer for timeshifting this data.

When the TimeshiftPlayer detects a DynamicLabel+ Item.toggle bit change, a new SkipItem object is created and added. These SkipItems provide a content based segmentation of the time shift buffer.

3.1. CLASSES (PUBLIC METHODS ONLY):

3.1.1. TimeshiftPlayer.java

```java
/**
 * Creates a (link TimeshiftPlayer)
 * @param context the Application (link Context). Must not be (code null).
 */
public TimeshiftPlayer(Context context)

/**
 * Prepares the (link TimeshiftPlayer) to timeshift
 * @param timeshiftService the (link RadioService) to timeshift
 * @throws IllegalArgumentException if the timeshiftService is not a DAB+ service. May change in further updates
 */
public void prepare(RadioService timeshiftService) throws IllegalArgumentException

/**
 * Stops the (link TimeshiftPlayer). You can reuse the instance by calling prepare.
 * @param deleteTmp (code true) if you want to delete the previously created timeshift file, (code false) otherwise
 */
public void stop(boolean deleteTmp)

/**
 * Call it when you wish that playback starts immediately without calling play()
 */
public void setPlayWhenReady()
```
3.3: HRADIO mobile and HTML client API implementations

/**
* Starts the timeshift playback. You can call this only once. Use pause toggle to
* pause/unpause playback
*/
public void play()
/**
* Pauses or un-pauses the player
* @param pause the wanted state: {@code true} for pause, {@code false} for un-pause
*/
public void pause(boolean pause)
/**
* Indicates if the {@link TimeshiftPlayer} is currently paused
* @return {@code true} if the {@link TimeshiftPlayer} is paused, {@code false} otherwise
*/
public boolean isPaused()
/**
* Seeks to the given position
* @param seekMilliseconds the wanted position in milliseconds
*/
public void seek(long seekMilliseconds)
/**
* Returns a list of {@link SkipItem}s or an empty list
* @return a list of {@link SkipItem}s or an empty list
*/
public List<SkipItem> getSkipItems()
/**
* Skips to the {@link SkipItem}s state
* @param item the {@link SkipItem} to skip to
*/
public void skipTo(SkipItem item)
/**
* Returns the current total timeshifted duration in milliseconds
* @return the current total timeshifted duration in milliseconds
*/
public long getDuration()
/**
* Adds a {@link TimeshiftListener} to receive status updates
* @param listener {@link TimeshiftListener} to add
*/
public void addListener(TimeshiftListener listener)
/**
* Removes a {@link TimeshiftListener}
* @param listener {@link TimeshiftListener} to remove
*/
public void removeListener(TimeshiftListener listener)

3.1.2. SkipItem.java

/**
* Returns the number of previous saved AUs
* @return the number of previous saved AUs
*/
public long getWrittenAus()
/**
* Returns the skip point
* @return the skip point
*/
public long getSkipPoint()
/**
* Returns the time point relative to the start timepoint in milliseconds
* @return the time point relative to the start timepoint in milliseconds
*/
public long getRelativeTimepoint()
/**
* Returns the Textual at the point of the skip item
* @return the Textual at the point of the skip item
*/
public Textual getSkipTextual()
/**
* Returns a Visual at the point of the skip item or {@code null} if not available
* @return a Visual at the point of the skip item or {@code null} if not available
*/
public Visual getSkipVisual()

### 3.2. INTERFACES

#### 3.2.1. TimeshiftListener.java

```java
void progress(long curPos, long totalDuration);
void started();
void paused();
void stopped();
void textual(Textual timeshiftLabel);
void visual(Visual timeshiftVis);
void skipItemAdded(SkipItem addedItem);
```

Together with the component implementation an example app has been developed, which uses the OMRI library to receive DAB+ Radio services (after a successful service scan), time shifts the selected service and provides a GUI with the received SkipItems (DL+ text messages and slide show images).
This section describes technical details for developers about the availability, usage and feedback regarding this initial release of the HRADIO client libraries.

**Figure 7: Development workflow**

The drawing above shows the various tools and platforms that allow developers to use the existing libraries and examples in their own applications as easily as possible.

### 4.1. REPOSITORIES

Source code for all developments in HRADIO is provided on a GitLab instance available on the network:

https://gitlab-ext.irt.de/

GitLab is a well-known platform for distributed software development projects. Amongst many other powerful features, it provides central access to the source repositories, an issue tracking and WIKI Pages for simple documentation purposes. For the HRADIO project a closed group HRADIO with members from all the project partners has been set up. This group will not only be the umbrella for the client library development tasks, but also host the developments for Metadata and Communication platform, for the pilots and the Lab-Playout systems.
4.2. IDE ANDROID STUDIO

For development software for the Android platform, Android Studio will be used.

https://developer.android.com/studio/

Android Studio is the first stop for development on this platform and provides not only source code editing, compilation and debugging tools. Additionally Android Studio supports the developer with advanced support and tools for:
4.3. LIBRARY MANAGEMENT

To facilitate the setup of a complete development environment for HRADIO application developers, in addition to GitLab and Android Studio a system that manages binary (fully compiled) dependencies has been set up. The JFrog Artifactory automatically resolves the required dependencies for the developer, loads and makes them automatically available to the build system.
Like on the GitLab platform all necessary HRADIO project members have accounts on the system to access or manage the different artifacts for the different products to build. This setup makes it extremely easy to switch between different library version during development, e.g. stable and beta releases.

![JFrog Artifactory](image)

**Figure 10: Artifactory for HRADIO**

### 4.4. SIMPLE DEVELOPER WORKFLOW EXAMPLE

The following section walks through the necessary steps a novice developer has to perform, in order to come to a fully working HRADIO development setup.

1. Create accounts to access GitLab and Artifactory
2. Clone the GitLab Source project

```bash
git clone git@gitlab-ext.irt.de:hradio/TimeshiftPlayerExample.git
```
Note: It’s not necessary to also clone the library projects in order to fulfil build dependencies, as this functionality is provided by the Artifactory.

3: In the “Authentication Settings” of the Artifactory profile generate an encrypted API Key for access.

4: Configure the .gradle properties file with the credentials created in step 3.

5: Open the cloned Android Studio project from step2 and build. During this 1st build process, the project configuration and the access to the Artifactory take care, that:

- the required OMRI API definition and API implementation is loaded and
- that the required TimeshiftPlayer library is loaded.

Figure 11: Successful APK build fot “timeshifsampleapp”
5. CONCLUSIONS

This deliverable is the first of three deliverables, documenting the progress of work in T3.3 “Client libraries”. In this first phase of T3.3 the goal was to quickly enable applications developers to use the OMRI API for their developments. Therefore T3.3 focused on the well-known and widely available Android platform. A fully working Android implementation of OMRI has been delivered. Additionally, a TimeShift player component has been developed and provided to the project partners. In order to enable a quick adoption of this technologies by the developers, an easy to setup but powerful working platform (GIT, Android Studio, Artifactory...) has been set up.

Future work of T3.3 will concentrate on the implementations of OMRI for the other platforms e.g. iOS and HTML/JavaScript and an integration of a IP based DAB streaming component.
APPENDIX A

The following section lists the formal API definition in Java.

Developers will find the Java skeletons of the org.omri packages on the specifications Git repository hosted on:

https://github.com/ebu

A.1 PACKAGE ORG.OMRI.RADIO

A.1.1 Radio

```java
package org.omri.radio;
import java.util.List;
import org.omri.radio.impl.RadioImpl;
import org.omri.radioservice.RadioService;
import org.omri.tuner.Tuner;
import org.omri.tuner.TunerListener;
import org.omri.tuner.TunerType;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License";
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * The main Radio class
 * The implementer has to implement RadioImpl implementation
 * @author FabianSattler, IRT GmbH
 */
public abstract class Radio {
    /** the singleton instance of Radio */
    private static Radio INSTANCE = new RadioImpl();
    /**
     * Returns the Radio instance or null if not implemented
     * @return the Radio instance or null if not implemented
     */
    public static Radio getInstance() {
        return INSTANCE;
    }
    /**
     * Initializes the Radio instance with an Java Object. Intention
     * of the appContext parameter is, that applications get the opportunity to
     * pass necessary platform specific objects into the OMRI implementation
     * i.e. Android Context
     * @param appContext the AppContext
     */
    public abstract RadioErrorCode initialize();
}
```

D3.3: HRADIO mobile and HTML client API implementations
public abstract RadioErrorCode initialize(Object appContext);

/**
 * Suspends the {@link Radio} and with it all {@link Tuner}s
 * @return {@link RadioErrorCode} indicating the success of the suspend.
 */
public abstract RadioErrorCode suspend();

/**
 * Resumes the {@link Radio} to the previous state before it was suspended.
 * @return {@link RadioErrorCode} indicating the success of the resume
 */
public abstract RadioErrorCode resume();

/**
 * Indicates the current status of the {@link Radio}
 * @return the current {@link RadioStatus}
 */
public abstract RadioStatus getRadioStatus();

/**
 * Deinitializes the Radio and all {@link Tuner}s
 */
public abstract void deInitialize();

/**
 * Returns the available {@link Tuner}s or an empty list
 * @return the available {@link Tuner}s or an empty list
 */
public abstract List<Tuner> getAvailableTuners();

/**
 * Returns the available {@link Tuner}s for a specific {@link TunerType} or an empty list
 * @return the available {@link Tuner}s for a specific {@link TunerType} or an empty list
 */
public abstract List<Tuner> getAvailableTuners(TunerType tunerType);

/**
 * Retrieve the currently known {@link RadioService}s of this {@link Radio} device
 * The method here is for the convenience of the application developer.
 * @return list of {@link RadioService}s or an empty list
 */
public abstract List<RadioService> getRadioServices();

/**
 * Start a {@link RadioService} on an available tuner
 * The method here is for the convenience of the application developer.
 * @param radioService the {@link RadioService} to start
 */
public abstract void startRadioService(RadioService radioService);

/**
 * Scans using all tuners and builds the combined service list.
 * The method here is for the convenience of the application developer.
 * If the application developer wants to perform service scans in the background
 * (in case the Radio exposes more than one {@link Tuner} instances), it's recommended
 * to use the dedicated method calls in the {@link Tuner} objects.
 */
public abstract void startRadioServiceScan();

/**
 * Stops the possible running service scan on all available tuners.
 * The method here is for the convenience of the application developer.
 * If the application developer wants to perform service scans in the background
 * (in case the Radio exposes more than one {@link Tuner} instances), it's recommended
 * to use the dedicated method calls in the {@link Tuner} objects.
 */
public abstract void stopRadioServiceScan();

A.1.2 RadioErrorCode

package org.omri.radio;
D3.3: HRADIO mobile and HTML client API implementations

```java
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

public enum RadioErrorCode {
    ERROR_INIT_OK(0, "No Error"),
    ERROR_INIT_NOT_OKAY(1, "Init error"),
    ERROR_INIT_FATAL_ERROR(2, "Fatal error"),
    ERROR_SUSPEND_OK(3, "Radio suspended"),
    ERROR_SUSPEND_FAILED(4, "Radio could not be suspended"),
    ERROR_RESUME_OK(5, "Radio resumed"),
    ERROR_RESUME_FAILED(6, "Radio could not be resumed"),
    
    private final int errorCode;
    private final String errorDescription;
    private RadioErrorCode(int errorCode, String errorDescription) {
        this.errorCode = errorCode;
        this.errorDescription = errorDescription;
    }
    public int getErrorCode() {
        return this.errorCode;
    }
    public String getErrorCodeDescription() {
        return this.errorDescription;
    }
}
```

A.1.3 RadioListener

```java
package org.omri.radio;
import org.omri.tuner.TunerListener;
/**
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

public interface RadioListener {
}
```

A.1.4 RadioStatus

```java
package org.omri.radio;
```
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Status codes for {@link Radio}
 * @author FabianSattler, IRT GmbH
 */
public enum RadioStatus {
    STATUS_RADIO_UNINITIALIZED(0, "Radio is uninitialized"),
    STATUS_RADIO_RUNNING(1, "Radio is running"),
    STATUS_RADIO_SUSPENDED(2, "Radio is suspended");

    private final int statusCode;
    private final String statusDescription;

    private RadioStatus(int statusCode, String statusDescription) {
        this.statusCode = statusCode;
        this.statusDescription = statusDescription;
    }

    public int getStatusCode() {
        return this.statusCode;
    }

    public String getStatusDescription() {
        return this.statusDescription;
    }
}

A.2 PACKAGE ORG.OMRI.RADIOSERVICE

A.2.1 RadioService
package org.omri.radioservice;
import java.util.List;
import org.omri.radioservice.metadata.Group;
import org.omri.radioservice.metadata.Location;
import org.omri.radioservice.metadata.TermId;
import org.omri.radioservice.metadata.Visual;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * Abstract base class for a radio service
 * @author FabianSattler, IRT GmbH
 * @author Erk, IRT GmbH
 */
public interface RadioService {
* Indicates the type of this RadioService
* @return the {@link RadioServiceType} of this RadioService
*/
public RadioServiceType getRadioServiceType();
/**
* Returns the label of this {@link RadioService} as {@link String}
* @return the label of this {@link RadioService} as {@link String}
*/
public String getServiceLabel();
/**
* Returns the short description of this {@link RadioService} as {@link String}
* @return The short description of this {@link RadioService} as {@link String}
*/
public String getShortDescription();
/**
* Returns the long description of this {@link RadioService} as {@link String}
* @return The long description of this {@link RadioService} as {@link String}
*/
public String getLongDescription();
/**
* Returns the available {@link Visual}s for this {@link RadioService} or an empty list
* @return the available {@link Visual}s for this {@link RadioService} or an empty list
*/
public List<Visual> getLogos();
/**
* Returns the available {@link TermId}s for this {@link RadioService} or an empty list
* @return the available {@link TermId}s for this {@link RadioService} or an empty list
*/
public List<TermId> getGenres();
/**
* Returns the available Links for this {@link RadioService} or an empty list
* @return the available Links for this {@link RadioService} or an empty list
*/
public List<String> getLinks();
/**
* Returns the available Locations for this {@link RadioService} or an empty list
* @return the available Locations for this {@link RadioService} or an empty list
*/
public List<Location> getLocations();
/**
* Returns the available keywords for this {@link RadioService} or an empty list
* @return the available keywords for this {@link RadioService} or an empty list
*/
public List<String> getKeywords();
/**
* Returns the available Groups for this {@link RadioService} or an empty list
* @return the available Groups for this {@link RadioService} or an empty list
*/
public List<Group> getMemberships();
/**
* Subscribe a {@link RadioServiceListener} to receive updates from this {@link RadioService}
* @param radioServiceListener the {@link RadioServiceListener} to subscribe
*/
public void subscribe(RadioServiceListener radioServiceListener);
/**
* Unsubscribe a {@link RadioServiceListener} from this {@link RadioService}
* @param radioServiceListener the {@link RadioServiceListener} to unsubscribe
*/
public void unsubscribe(RadioServiceListener radioServiceListener);
}

A.2.2 RadioServiceAudiodataListener

package org.omri.radioservice;
/**
* Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
* Licensed under the Apache License, Version 2.0 (the "License"); you may not use this
file except in compliance with the License. You may obtain a copy of the License at
* http://www.apache.org/licenses/LICENSE-2.0
* Unless required by applicable law or agreed to in writing, software

D3.3: HRADIO mobile and HTML client API implementations

* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.

* Interface to receive the decoded PCM audio stream
* @author Fabian Sattler, IRT GmbH
*/
public interface RadioServiceAudiodataListener extends RadioServiceListener {

/**
 * PCM audio data interface
 * @param pcmData the pcm data, encoded as interleaved signed 16 bit little endian PCM
 * @param numChannels the number of audio channels
 * @param samplingRate the sampling rate
 */
public void pcmAudioData(byte[] pcmData, int numChannels, int samplingRate);
}

A.2.3 RadioServiceDab

package org.omri.radioservice;
import java.util.List;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * @author Fabian Sattler, IRT GmbH
 */
public interface RadioServiceDab extends RadioService {

/**
 * Returns the extended country code (ECC) for the DAB Ensemble this @{link RadioServiceDab} belongs to
 */
public String getEnsembleEcc();

/**
 * Returns the DAB Ensemble ID, this @{link RadioServiceDab} belongs to, as hex-string
 */
public String getEnsembleId();

/**
 * Returns the label of the DAB Ensemble, this @{link RadioServiceDab} belongs to
 */
public String getEnsembleLabel();

/**
 * Returns the frequency in kHz of the DAB Ensemble, this @{link RadioServiceDab} belongs to
 */
public long getEnsembleFrequency();

/**
 * Returns the short label of this @{link RadioServiceDab}
 */
public String getShortLabel();
/**
 * Returns the service id as hex-string
 * @return the service id as hex-string
 */
public String getServiceId();

/**
 * Returns the service extended country code
 * @return the service extended country code
 */
public String getServiceEcc();

/**
 * Indicates if this RadioServiceDab is a DAB programme or a DAB data service
 * @return indication for programme or data service
 */
public boolean isProgrammeService();

/**
 * Returns a list with the RadioServiceDabComponent associated with this RadioServiceDab
 * @return list with the RadioServiceDabComponent associated with this RadioServiceDab
 */
public List<RadioServiceDabComponent> getServiceComponents();

A.2.4 RadioServiceDabComponent

package org.omri.radioservice;
import java.util.List;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License";
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Abstract class containing informations for a service component (SC) for a RadioServiceDab
 * @authorFabianSattler, IRT GmbH
 */
public interface RadioServiceDabComponent {
/**
 * Returns the bitrate in kbit/s for this RadioServiceDabComponent
 * @return bitrate in kbit/s for this RadioServiceDabComponent
 */
public int getScBitrate();

/**
 * Indicates if the CA (Conditional Access) flag is set
 * @return indication for conditional access
 */
public boolean isScCaFlagSet();

/**
 * Returns the RadioServiceDabComponent's channel ID
 * @return the RadioServiceDabComponent's channel ID
 */
public int getScChannelId();

/**
 * Indicates if the DG flag is set
 * @return indication for DG
 */
public boolean isScDgFlagSet();
/**
* Returns the [@link RadioServiceDabComponent]ID
* @return the [@link RadioServiceDabComponent]ID
*/
public int getScId();
/**
* Returns the label for this [@link RadioServiceDabComponent]
* @return the label for this [@link RadioServiceDabComponent]
*/
public String getScLabel();
/**
* Returns the packet address
* @return the packet address
*/
public int getScPacketAddress();
/**
* Indicates if this [@link RadioServiceDabComponent] is the primary component of this
* [@link RadioServiceDab]
* @return if this [@link RadioServiceDabComponent] is the primary component of this
* [@link RadioServiceDab]
*/
public boolean isScPrimary();
/**
* Returns the service component ID
* @return the service component ID
*/
public int getScServiceComponentId();
/**
* Returns the Transport Mode (TM) ID
* @return the Transport Mode (TM) ID
*/
public int getTmId();
/**
* Returns the service component type
* @return the service component type
*/
public int getScType();
/**
* Returns a list with [@link RadioServiceDabUserApplication]s for this SC
* @return a list with [@link RadioServiceDabUserApplication]s for this SC
*/
public List<RadioServiceDabUserApplication> getScUserApplications();
/**
* Subscribe a [@link RadioServiceDabComponentListener] to receive updates from this
* [@link RadioServiceDabComponentListener]
* @param dabComponentListener the [@link RadioServiceDabComponentListener] to subscribe
*/
public void subscribe(RadioServiceDabComponentListener dabComponentListener);
/**
* Unsubscribe a [@link RadioServiceDabComponentListener]
* @param dabComponentListener the [@link RadioServiceDabComponentListener] to unsubscribe
*/
public void unsubscribe(RadioServiceDabComponentListener dabComponentListener);

A.2.5 RadioServiceDabComponentListener
package org.omri.radioservice;
/**
* Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

* Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.

* Interface to receive raw data from a {@link RadioServiceDabComponent} (e.g., Journaline data)

* @author FabianSattler, IRT GmbH
*/

    public interface RadioServiceDabComponentListener {
        /**
         * Called when new data from a specific DAB Service Component was received
         *
         * @param serviceComponentChannelId Service component id
         * @param scData raw data from the service component
         */
        public void newServiceComponentData(int serviceComponentChannelId, byte[] scData);
    }

A.2.6 RadioServiceDabUserApplication

    package org.omri.radioservice;
    /**
     * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
     *
     * Licensed under the Apache License, Version 2.0 (the "License");
     * you may not use this file except in compliance with the License.
     * You may obtain a copy of the License at
     * http://www.apache.org/licenses/LICENSE-2.0
     * Unless required by applicable law or agreed to in writing, software
     * distributed under the License is distributed on an "AS IS" BASIS,
     * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
     * See the License for the specific language governing permissions and
     * limitations under the License.
     * Abstract class containing informations about one user application for a
     * @link RadioServiceDabComponent
     * @author FabianSattler, IRT GmbH
     */
    public interface RadioServiceDabUserApplication {
        /**
         * Returns the user application label
         * @return the user application label
         */
        public String getUappLabel();

        /**
         * Returns the user application type as numerical value
         * @return the user application type as numerical value
         */
        public int getUappType();

        /**
         * Returns the user application type as hex-string
         * @return the user application type as hex-string
         */
        public String getUappTypeString();
    }

A.2.7 RadioServiceFm

    package org.omri.radioservice;
    /**
     * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
     */
    import org.mobicore.traffic.news.data.*;
    import org.mobicore.traffic.news.data.TrafficNewsData;
    import org.mobicore.traffic.news.data.TrafficNewsDataPoint;
    import org.mobicore.traffic.news.data.TrafficNewsDataPoint.Order;
    import org.mobicore.traffic.news.data.TrafficNewsDataPoint.Status;
    import org.mobicore.traffic.news.data.TrafficNewsDataPoint.Type;
A.2.8 RadioServiceFmPty

package org.omri.radioservice;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Abstract class representing a RDS PTY (ProgrammeType)
 * @author FabianSattler, IRT GmbH
 */

public interface RadioServiceFmPty {

  /**
   * Returns the PTY code
   * @return the PTY code
   */
  public int getPtyCode();

  /**
   * Returns a human readable representation of the PTY code
   * @return a human readable representation of the PTY code
   */
  public String getPtyDescription();
}
A.2.9 RadioServiceIp

```java
package org.omri.radioservice;
import java.util.List;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Abstract interface for an IP delivered {@link RadioService}
 * @author FabianSattler, IRT GmbH
 */
public interface RadioServiceIp extends RadioService {
  /**
   * The list of available {@link RadioServiceIpStream}s for this IP delivered {@link RadioService}
   * @return a list of available {@link RadioServiceIpStream} for this {@link RadioService}
   */
  public List<RadioServiceIpStream> getIpStreams();
}
```

A.2.10 RadioServiceIpStream

```java
package org.omri.radioservice;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Abstract interface which defines the necessary data for a single 'stream' of a {@link RadioService}
 * @author FabianSattler, IRT GmbH
 */
public interface RadioServiceIpStream {
  /**
   * Returns the URL of this stream (e.g. 'http://somewebstream:1337/genre')
   * @return the URL of this stream
   */
  public String getUrl();

  /**
   * Returns the bitrate in kbit/s for this stream
   * @return the bitrate in kbit/s
   */
  public int getBitrate();

  /**
   * Return the MIME type of this stream
   * @return the MIME type of this stream
   */
  public RadioServiceMimeType getMimeType();

  /**
   * Indicates the cost for this stream. Higher means more 'expensive'
   */
```
A.2.11 RadioServiceListener

```java
package org.omri.radioservice;
import org.omri.radio.RadioListener;
import org.omri.radioservice.metadata.ProgrammeServiceMetadataListener;
import org.omri.radioservice.metadata.TextualMetadataListener;
import org.omri.radioservice.metadata.VisualMetadataListener;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
public interface RadioServiceListener extends RadioListener {
    /**
     * Indicates the offset in seconds from the fastest corresponding broadcast bearer (e.g.
     * FM).
     * @return the offset in seconds.
     */
    public int getOffset();
}
```

A.2.12 RadioServiceMimeType

```java
package org.omri.radioservice;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
public enum RadioServiceMimeType {
    UNKNOWN("mime/unknown"),
    AUDIO_MPEG("audio/mpeg"),
    AUDIO_OGG_VORBIS("audio/ogg"),
    AUDIO_FLAC("audio/flac"),
    AUDIO_ADTS("audio/adi")
}
```
D3.3: HRADIO mobile and HTML client API implementations

private final String contentTypeString;
private RadioServiceMimeType(String contentTypeString) {
  this.contentTypeString = contentTypeString;
}
public String getMimeTypeString() {
  return this.contentTypeString;
}

A.2.13 RadioServiceType
package org.omri.radioservice;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * @link RadioService]type definitions
 * @author FabianSattler, IRT GmbH
 */
public enum RadioServiceType {
  /** RadioService type DAB **/
   RADIOSERVICE_TYPE_DAB,
  /** RadioService type IP **/
   RADIOSERVICE_TYPE_IP,
  /** RadioService type FM **/
   RADIOSERVICE_TYPE_FM,
  /** RadioService type SiriusXM **/
   RADIOSERVICE_TYPE_SIRIUS,
  /** RadioService type iBiquity HD Radio **/
   RADIOSERVICE_TYPE_HDRADIO;
}

A.3 PACKAGE ORG.OMRI.RADIOSERVICE.METADATA
A.3.1 Group
package org.omri.radioservice.metadata;
import java.util.List;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * Abstract base class for a radio service
 * @author FabianSattler, IRT GmbH
D3.3: HRADIO mobile and HTML client API implementations

* @authorErk, IRT GmbH
* /
public interface Group {

* Returns the CRID of this {@link Group}
* @returnthe CRID of this {@link Group}
* /
public String getCRID();

/**
 * Returns the short name of this {@link Group}as {@link String}
 * @returnThe short name of this {@link Group}as {@link String}
 */
public String getShortName();

/**
 * Returns the medium name of this {@link Group}as {@link String}
 * @returnThe medium name of this {@link Group}as {@link String}
 */
public String getMediumName();

/**
 * Returns the long name of this {@link Group}as {@link String}
 * @returnThe long name of this {@link Group}as {@link String}
 */
public String getLongName();

/**
 * Returns the short description of this {@link Group}as {@link String}
 * @returnThe short description of this {@link Group}as {@link String}
 */
public String getShortDescription();

/**
 * Returns the medium description of this {@link Group}as {@link String}
 * @returnThe medium description of this {@link Group}as {@link String}
 */
public String getMediumDescription();

/**
 * Returns the available {@link Visual}s for this {@link Group}or an empty list
 * @returnthe available {@link Visual}s for this {@link Group}or an empty list
 */
public List<Visual> getLogos();

/**
 * Returns the available {@link TermId}s for this {@link Group}or an empty list
 * @returnthe available {@link TermId}s for this {@link Group}or an empty list
 */
public List<TermId> getGenres();

/**
 * Returns the available Links for this {@link Group}or an empty list
 * @returnthe available Links for this {@link Group}or an empty list
 */
public List<String> getLinks();

/**
 * Returns the available keywords for this {@link Group}or an empty list
 * @returnthe available keywords for this {@link Group}or an empty list
 */
public List<String> getKeywords();

A.3.2 Location

package org.omri.radioservice.metadata;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
A.3.3 ProgrammeInformation

```java
package org.omri.radioservice.metadata;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *     http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * @author FabianSattler, IRT GmbH
 */
public interface ProgrammeInformation {
    /**
     * Returns the type of this ProgrammeInformation metadata
     * @return the type of this ProgrammeInformation metadata
     */
    public ProgrammeInformationType getType();
}
```

A.3.4 ProgrammeInformationType

```java
package org.omri.radioservice.metadata;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *     http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Programmemetadata type definitions
 *
 * @author Erk, IRT GmbH
 */
public enum ProgrammeInformationType {
    /** Programmemetadata received via RadioDNS/DAB SPI **/
     METADATA_PROGRAMME_TYPE_SPI_EPG
}
```
### A.3.5 ProgrammeServiceMetadataListener

```java
package org.omri.radioservice.metadata;
import org.omri.radioservice.RadioServiceListener;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * @interface ProgrammeServiceMetadataListener
 * @author FabianSattler, IRT GmbH
 */
public interface ProgrammeServiceMetadataListener extends RadioServiceListener {
    /**
     * New ProgrammeInformation was received
     * @param programmeInformation the ProgrammeInformation received
     */
    public void newProgrammeInformation(ProgrammeInformation programmeInformation);
    /**
     * New ServiceInformation was received
     * @param serviceInformation the ServiceInformation received
     */
    public void newServiceInformation(ServiceInformation serviceInformation);
}
```

### A.3.6 ServiceInformation

```java
package org.omri.radioservice.metadata;
import org.w3c.dom.Document;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License"
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * @author FabianSattler, IRT GmbH
 */
public interface ServiceInformation {
}
```

### A.3.7 SpiProgrammeInformation

```java
package org.omri.radioservice.metadata;
import org.w3c.dom.Document;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License"
 * you may not use this file except in compliance with the License.
 */
```
D3.3: HRADIO mobile and HTML client API implementations

* You may obtain a copy of the License at
  * http://www.apache.org/licenses/LICENSE-2.0
  *
* Unless required by applicable law or agreed to in writing, software
  * distributed under the License is distributed on an "AS IS" BASIS,
  * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  * See the License for the specific language governing permissions and
  * limitations under the License.

* Abstract base class for a radio service
  * @author Erk, IRT GmbH
  */
publicinterface SpiProgrammeInformation extends ProgrammeInformation {
  /**
   * Returns the SPI as a DOM object
   * @return a SPI as a {@link Document}
   */
  public Document getSpiDocument();
}

A.3.8 TermId
package org.omri.radioservice.metadata;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
publicinterface TermId {
  /**
   * Returns the href of this termID
   * @return the href of this termID
   */
  public int getHref();
  /**
   * Returns text of this termID.
   * @return text of this termID
   */
  public String getText();
}

A.3.9 Textual
package org.omri.radioservice.metadata;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * Abstract base class for textual metadata
D3.3: HRADIO mobile and HTML client API implementations

* @author FabianSattler, IRT GmbH

```java
public interface Textual {
    /**
     * Returns the type of this textual metadata
     * @return the type of this textual metadata
     */
    public TextualType getType();

    /**
     * Returns the [String] representation of this textual metadata
     * @return the [String] representation of this textual metadata
     */
    public String getText();
}
```

A.3.10 TextualDabDynamicLabel

```java
package org.omri.radioservice.metadata;
import java.util.List;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Abstract class for a DAB Dynamic Label (Plus) [Textual] metadata
 * @author FabianSattler, IRT GmbH
 */
public interface TextualDabDynamicLabel extends Textual {
    /**
     * Indicates if this [TextualDabDynamicLabel] has DL+ tags
     * @return indication of DL+ tags
     */
    public boolean hasTags();

    /**
     * Returns the number of tags. Only applicable for DL+. Check 'hasTags()' for tags
     * @return the number of tags
     */
    public int getTagCount();

    /**
     * Returns a list of [TextualDabDynamicLabelPlusItem]s or an empty list
     * @return a list of [TextualDabDynamicLabelPlusItem]s or an empty list
     */
    public List<TextualDabDynamicLabelPlusItem> getDlPlusItems();
}
```

A.3.11 TextualDabDynamicLabelPlusContentType

```java
package org.omri.radioservice.metadata;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
```
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.

* Dynamic Label Plus Content type definitions
*
* @author FabianSattler, IRT GmbH
*/

public enum TextualDabDynamicLabelPlusContentType {
    DUMMY(0, "Dummy"),
    ITEM_TITLE(1, "Title"),
    ITEM_ALBUM(2, "Album"),
    ITEM_TRACKNUMBER(3, "Tracknumber"),
    ITEM_ARTIST(4, "Artist"),
    ITEM_COMPOSITION(5, "Composition"),
    ITEM_movement(6, "Movement"),
    ITEM_CONDUCTOR(7, "Conductor"),
    ITEM_COMPOSER(8, "Composer"),
    ITEM_BAND(9, "Band"),
    ITEM_COMMENT(10, "Comment"),
    ITEM_GENRE(11, "Genre"),
    INFO_NEWS(12, "News"),
    INFO_NEWS_LOCAL(13, "Local News"),
    INFO_STOCKMARKET(14, "Stockmarket"),
    INFO_SPORT(15, "Sport"),
    INFO_Lottery(16, "Lottery"),
    INFO_HOROSCOPE(17, "Horoscope"),
    INFO_DAILY_DIVERSION(18, "Daily Diversion"),
    INFO_HEALTH(19, "Health"),
    INFO_EVENT(20, "Event"),
    INFO_SCENE(21, "Scene"),
    INFO_CINEMA(22, "Cinema"),
    INFO_TV(23, "TV"),
    INFO_DATE_TIME(24, "Date and Time"),
    INFO_WEATHER(25, "Weather"),
    INFO_TRAFFIC(26, "Traffic"),
    INFO_ALARM(27, "Alarm"),
    INFO_ADVERTISEMENT(28, "Advertisement"),
    INFO_URL(29, "URL"),
    INFO_CURRENT(30, "Other"),
    STATIONNAME_SHORT(31, "Short Stationname"),
    STATIONNAME_LONG(32, "Long Stationname"),
    PROGRAMME_NOW(33, "Now"),
    PROGRAMME_NEXT(34, "Next"),
    PROGRAMME_PART(35, "Part"),
    PROGRAMME_HOST(36, "Host"),
    PROGRAMME_EDITORIAL_STAFF(37, "Editorial Staff"),
    PROGRAMME_FREQUENCY(38, "Frequency"),
    PROGRAMME_HOMEPAGE(39, "Homepage"),
    PROGRAMME_SUBCHANNEL(40, "Subchannel"),
    PHONE_HOTLINE(41, "Hotline"),
    PHONE_STUDIO(42, "Studio Telephone"),
    PHONE_OTHER(43, "Other Telephone"),
    SMS_HOTLINE(44, "Studio SMS"),
    SMS_OTHER(45, "SMS"),
    EMAIL_HOTLINE(46, "Hotline Email"),
    EMAIL_STUDIO(47, "Studio Email"),
    EMAIL_OTHER(48, "Email"),
    MMS_OTHER(49, "MMS"),
    CHAT(50, "Chat"),
    CHAT_CENTER(51, "Chat Center"),
    VOTE_QUESTION(52, "Vote Question"),
    VOTE_CENTRE(53, "Vote Centre"),
    RFU_1(54, "RFU1"),
    RFU_2(55, "RFU2"),
    PRIVATE_CLASS_1(56, "Private Data 1"),
    PRIVATE_CLASS_2(57, "Private Data 2"),
    PRIVATE_CLASS_3(58, "Private Data 3"),
    DESCRIPTOR_PLACE(59, "Place"),
    DESCRIPTOR_APPORTMENT(60, "Appointment"),
    DESCRIPTOR_IDENTIFIER(61, "Identifier"),
    DESCRIPTOR_PURCHASE(62, "Purchase"),
    DESCRIPTOR_GET_DATA(63, "Get Data"),
    DESCRIPTOR_PURCHASE(63, "Get Data")
}

private final int contentTypeId;
private final String contentTypeString;

private TextualDabDynamicLabelPlusContentType(int contentTypeId, String contentTypeString) {
    this.contentTypeId = contentTypeId;
    this.contentTypeString = contentTypeString;
}

public int getContentTypeId() {
    return this.contentTypeId;
}

public String getContentTypeString() {
    return this.contentTypeString;
}

A.3.12 TextualDabDynamicLabelPlusItem

package org.omri.radioservice.metamodel;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * * Licensed under the Apache License, Version 2.0 (the "License");
 * * you may not use this file except in compliance with the License.
 * * You may obtain a copy of the License at
 * * http://www.apache.org/licenses/LICENSE-2.0
 * * Unless required by applicable law or agreed to in writing, software
 * * distributed under the License is distributed on an "AS IS" BASIS,
 * * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * * See the License for the specific language governing permissions and
 * * limitations under the License.
 * *
 * * Abstract class containing information about one DL+ tag
 * * @author Fabian Sattler, IRT GmbH
 */
public interface TextualDabDynamicLabelPlusItem {
    /**
     * Returns the {TextualDabDynamicLabelPlusContentType}
     * @return the {TextualDabDynamicLabelPlusContentType}
     */
    public TextualDabDynamicLabelPlusContentType getDynamicLabelPlusContentType();

    /**
     * Returns a textual description of the content type
     * @return textual description of the content type
     */
    public String getDlPlusContentTypeDescription();

    /**
     * Returns the content category of this tag
     * @return the content category of this tag
     */
    public String getDlPlusContentCategory();

    /**
     * Returns the text of this DL+ tag
     * @return the text of this DL+ tag
     */
    public String getDlPlusContentText();
}

A.3.13 TextualFmRdsRadioText

package org.omri.radioservice.metamodel;

/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * * Licensed under the Apache License, Version 2.0 (the "License");
 * * you may not use this file except in compliance with the License.
 * * You may obtain a copy of the License at
 * */
D3.3: HRADIO mobile and HTML client API implementations

* http://www.apache.org/licenses/LICENSE-2.0

* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.

* Abstract class for a FM RDS Radiotext {@link Textual} metadata

* @author FabianSattler, IRT GmbH
*/
publicinterface TextualFmRdsRadiotext extends Textual {
}

A.3.14 TextualIPRdnsRadioVis

package org.omri.radioservice.metadata;
/**
* Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
* http://www.apache.org/licenses/LICENSE-2.0
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
* Abstract class for IP delivered RadioDNS RadioVIS {@link Textual} metadata
* @author FabianSattler, IRT GmbH
*/
publicinterface TextualIPRdnsRadioVis extends Textual {
}

A.3.15 TextualMetadataListener

package org.omri.radioservice.metadata;
import org.omri.radioservice.RadioServiceListener;
/**
* Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
* http://www.apache.org/licenses/LICENSE-2.0
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
* Interface to receive {@link Textual} programme associated metadata
* @author FabianSattler, IRT GmbH
*/
publicinterface TextualMetadataListener extends RadioServiceListener {
/**
* New {@link Textual} metadata was received
* @param textualMetadata the {@link Textual} received
*/
public void newTextualMetadata(Textual textualMetadata);
A.3.16 **TextualType**

```java
package org.omri.radioservice.metadata;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Textual metadata type definitions
 * @author FabianSattler, IRT GmbH
 */
public enum TextualType {
    /** Textual metadata received via DAB Dynamic Label service **/
     METADATA_TEXTUAL_TYPE_DAB_DLS,
    /** Textual metadata received via RadioDNS RadioVIS Text service **/
     METADATA_TEXTUAL_TYPE_RADIODNS_RADIOVIS,
    /** Textual metadata received via FM Radiotext **/
     METADATA_TEXTUAL_TYPE_FM_RDS_RADIOTEXT,
    /** Textual metadata received via ID3 parsing **/
     METADATA_TEXTUAL_TYPE_ID3_TEXT,
    /** Textual metadata received via ShoutcastICY parsing **/
     METADATA_TEXTUAL_TYPE_ICY_TEXT;
}
```

A.3.17 **Visual**

```java
package org.omri.radioservice.metadata;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 * Abstract base class for visual metadata (e.g. DAB Slideshow, RadioDNS RadioVIS, etc.)
 * @author FabianSattler, IRT GmbH
 */
public interface Visual {
    /**
     * Returns the type (source) of this visual metadata (e.g. DAB Slideshow, RadioDNS RadioVIS, etc.)
     * @return the type of this visual metadata
     */
    public VisualType getVisualType();
    /**
     * Returns the type of this visual metadata
     * @return the type of this visual metadata
     */
    public VisualMimeType getVisualMimeType();
    /**
     * Returns the actual image data
     * @return the actual image data
     */
    public byte[] getVisualData();
```
package org.omri.radioservice.metadata;
import java.net.URI;
import java.util.Calendar;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License"),
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * Abstract class for a DAB Slideshow metadata
 * @author Fabian Sattler, IRT GmbH
 */
public interface VisualDabSlideShow extends Visual {
    /**
     * Indicates if this VisualDabSlideShow has categorization information
     * @return indication for categorization
     */
    public boolean isCategorized();
    /**
     * The content name.
     * @return the content name
     */
    public String getContentName();
    /**
     * The ID of this VisualDabSlideShow
     * @return the ID
     */
    public int getSlideId();
    /**
     * The Triggertime
     * @return a Calendar for the Triggertime or 'null' if the triggertime is 'now'
     */
    public Calendar getTriggerTime();
    /**
     * The ExpiryTime
     * @return a Calendar for the Expirytime or 'null'
     */
    public Calendar getExpiryTime();
    /**
     * Returns the category description of this VisualDabSlideShow. Only applicable for a DAB Categorized Slideshow (check 'isCategorized()')
     * @return the category text or an empty String if it's not a categorized VisualDabSlideShow
     */
    public String getCategoryText();
    /**
     * Returns the category id of this VisualDabSlideShow. Only applicable for a DAB Categorized Slideshow (check 'isCategorized()')
     * @return the category id or '-1' if it's not a categorized VisualDabSlideShow
     */
    public int getCategoryId();
}
D3.3: HRADIO mobile and HTML client API implementations

* Returns the link associated with this {@link VisualDabSlideShow}. Only applicable for a DAB Categorized Slideshow (check 'isCategorized()').
* @return the link associated with this {@link VisualDabSlideShow} or an empty {@link String}
*="/"
public URI getLink();
/**
* Returns the click through link associated with this {@link VisualDabSlideShow}.
* @return the click through link associated with this {@link VisualDabSlideShow} or an empty {@link String}
*="/"
public URI getClickThroughUrl();
/**
* Returns the alternative location link associated with this {@link VisualDabSlideShow}.
* @return the alternative location link associated with this {@link VisualDabSlideShow} or an empty {@link String}
*="/"
public URI getAlternativeLocationURL();
/**
* The MOT object Content type of this {@link VisualDabSlideShow}
* @return the Content Type
*/
public int getContentType();
/**
* The MOT object content subtype of this {@link VisualDabSlideShow}
* @return the content subtype
*/
public int getContentSubType();

A.3.19 VisualIpRdnsRadioVis

package org.omri.radioservice.metadata;
/**
* Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
* Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
* http://www.apache.org/licenses/LICENSE-2.0
* Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
* Abstract class for a IP delivered RadioDNS RadioVis {@link Visual} metadata
* @author Fabian Sattler, IRT GmbH
*/
public interface VisualIpRdnsRadioVis extends Visual {
/**
* Returns the trigger time as POSIX time (seconds elapsed since 1.1.1970)
* @return the trigger time as POSIX time
*/
public long getTriggerTime();

A.3.20 VisualMetadataListener

package org.omri.radioservice.metadata;
import org.omri.radioservice.RadioServiceListener;
/**
* Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
* Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*/
public interface VisualMetadataListener;
D3.3: HRADIO mobile and HTML client API implementations

* http://www.apache.org/licenses/LICENSE-2.0
* Unless required by applicable law or agreed to in writing, software
  * distributed under the License is distributed on an "AS IS" BASIS,
  * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  * See the License for the specific language governing permissions and
  * limitations under the License.
* Interface to receive {@link Visual} programme associated metadata
* @author FabianSattler, IRT GmbH
*/

public interface VisualMetadataListener extends RadioServiceListener {
  /**
   * New {@link Visual} metadata was received
   * @param visualMetadata the {@link Visual} received
   */
  public void newVisualMetadata(Visual visualMetadata);
}

A.3.21 VisualMimeType

package org.omri.radioservice.metadata;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * Visual metadata types
 * @author FabianSattler, IRT GmbH
 */

public enum VisualMimeType {
  METADATA_VISUAL_MIMETYPE_UNKNOWN,
  METADATA_VISUAL_MIMETYPE_JPEG,
  METADATA_VISUAL_MIMETYPE_PNG,
  METADATA_VISUAL_MIMETYPE_TIFF,
  METADATA_VISUAL_MIMETYPE_BMP,
  METADATA_VISUAL_MIMETYPE_WEBP,
  METADATA_VISUAL_MIMETYPE_SVG,
  METADATA_VISUAL_MIMETYPE_GIF,
  METADATA_VISUAL_MIMETYPE_ANIMATED_GIF;
}

A.3.22 VisualType

package org.omri.radioservice.metadata;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * Visual type definition
 */
A.4 PACKAGE PACKAGE ORG.OMRI.TUNER

A.4.1 Tuner

class Tuner
{
    //... public interface methods...
}
* Retrieve the currently known (@link RadioService)s of this (@link Tuner) *
@returns a list of (@link RadioService)s or an empty list */
public List<RadioService> getRadioServices();
/** *
* Start a scan for available (@link RadioService)s *
*/
public void startRadioServiceScan();
/** *
* Stops a currently running (@link RadioService) scan *
*/
public void stopRadioServiceScan();
/** *
* Start a (@link RadioService) on this tuner *
* @param radioService the (@link RadioService) to start *
*/
public void startRadioService(RadioService radioService);
/** *
* Stop the currently running (@link RadioService) on this tuner *
*/
public void stopRadioService();
/** *
* Retrieve the currently running (@link RadioService) *
* @return the currently running (@link RadioService) or null if no (@link RadioService) is running *
*/
public RadioService getCurrentRunningRadioService();
/** *
* Subscribe a (@link TunerListener) to receive updates of this tuner *
* @param tunerListener the (@link TunerListener) to subscribe *
*/
public void subscribe(TunerListener tunerListener);
/** *
* Unsubscribe a (@link TunerListener) *
* @param tunerListener the (@link TunerListener) to unsubscribe *
*/
public void unsubscribe(TunerListener tunerListener);

A.4.2 TunerListener
package org.omri.tuner;
import org.omri.radio.RadioListener;
import org.omri.radioservice.RadioService;
/** *
* Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group *
* Licensed under the Apache License, Version 2.0 (the "License"); *
* you may not use this file except in compliance with the License. *
* You may obtain a copy of the License at *
* http://www.apache.org/licenses/LICENSE-2.0 *
* Unless required by applicable law or agreed to in writing, software *
* distributed under the License is distributed on an "AS IS" BASIS, *
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. *
* See the License for the specific language governing permissions and *
* limitations under the License. *
* The (@link TunerListener) interface *
* @author FabianSattler, IRT GmbH *
*/
public interface TunerListener extends RadioListener {
    /** *
     * Tuner changed its operating state *
     * @param newStatus the new (@link TunerStatus) *
     */
    public void tunerStatusChanged(Tuner tuner, TunerStatus newStatus);
public void tunerScanProgress(Tuner tuner, int percentScanned) {
    // Tuner scan progress indicator
    // @param percentScanned the percentage finished so far
}

public void tunerScanServiceFound(Tuner tuner, RadioService service) {
    // A RadioService started
    // @param service the RadioService which has been found
}

public void radioServiceStarted(Tuner tuner, RadioService service) {
    // A RadioService started
    // @param service the RadioService which has started
}

public void radioServiceStopped(Tuner tuner, RadioService service) {
    // A RadioService stopped
    // @param service the RadioService which has stopped
}

public void tunerReceptionStatistics(Tuner tuner, boolean rfLock, int rssi) {
    // Updates on RF reception statistics
    // @param rfLock RF tuner frontend gained lock
    // @param rssi the Received Signal Strength Indicator (in dBµV[LS1] ???)
}

public void tunerRawData(Tuner tuner, byte[] data) {
    // Implementation and TunerType dependent raw data (e.g. in case of a DAB Tuner raw Fast Information Blocks)
    // @param tuner the Tuner from which the raw data was received
    // @param data the raw data
}

---

package org.omri.tuner;

/**
 * Status codes for the Tuner
 * @author Fabian Sattler, IRT GmbH
 */
public enum TunerStatus {
    /** Tuner is not initialized **/
     TUNER_STATUS_NOT_INITIALIZED(0, "Tuner not initialized"),
    /** Tuner is ready **/
     TUNER_STATUS_INITIALIZED(1, "Tuner ready"),
    /** Tuner is in an error state **/
     TUNER_STATUS_ERROR(2, "Tuner is in an error state"),
    /** Tuner is in suspended state **/
     TUNER_STATUS_SUSPENDED(3, "Tuner is suspended"),
    /** Tuner started scan for services **/
     TUNER_STATUS_SCANNING(4, "Tuner is scanning for services")
};
D3.3: HRADIO mobile and HTML client API implementations

```java
this.statusDescription = statusDescription;
}
public int getStatusCode() {
    return this.statusCode;
}
public String getStatusDescription() {
    return this.statusDescription;
}
```}

A.4.4 TunerType

```java
package org.omri.tuner;
/**
 * Copyright (C) 2016 Open Mobile Radio Interface (OMRI) Group
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 * http://www.apache.org/licenses/LICENSE-2.0
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 *
 **/enum TunerType {
    /** DAB Tuner **/ TUNER_TYPE_DAB,
    /** IP Tuner **/ TUNER_TYPE_IP,
    /** FM Tuner **/ TUNER_TYPE_FM,
    /** SiriusXM Tuner **/ TUNER_TYPE_SIRIUS,
    /** iBiquity HD Radio Tuner **/ TUNER_TYPE_HDRADIO;
}
```