D4.4: Report on final user interfaces and integrations

Editor: Klaes Baert (VRT)

This deliverable provides a report of the preparations for the final pilot, including user interface mock-ups, implemented user interface description and specifications, and a report on the integration between user interfaces and the technical components.
<table>
<thead>
<tr>
<th>Work package</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Due date</td>
<td>30/11/2019</td>
</tr>
<tr>
<td>Submission date</td>
<td></td>
</tr>
<tr>
<td>Deliverable lead</td>
<td>VRT</td>
</tr>
<tr>
<td>Version</td>
<td>1.0</td>
</tr>
<tr>
<td>Authors</td>
<td>Klaas Baert (VRT)</td>
</tr>
<tr>
<td>Reviewers</td>
<td>Alexander Erk (IRT), Benjamin Schmidt (Konsole Labs)</td>
</tr>
</tbody>
</table>

<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>12/11/2019</td>
<td>Initial version</td>
<td>Klaas Baert (VRT)</td>
</tr>
<tr>
<td>V0.2</td>
<td>18/12/2019</td>
<td>Draft for review</td>
<td>Klaas Baert (VRT) Alexander Erk (IRT)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Benjamin Schmidt (Konsole Labs)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Simone Hollederer (RBB)</td>
</tr>
<tr>
<td>V0.3</td>
<td>19/12/2019</td>
<td>IRT Review</td>
<td>Alexander Erk (IRT)</td>
</tr>
<tr>
<td>V0.4</td>
<td>08/01/2020</td>
<td>Konsole Labs Review</td>
<td>Benjamin Schmidt (Konsole Labs)</td>
</tr>
<tr>
<td>V1.0</td>
<td>15/01/2020</td>
<td>Final version after coordinator review</td>
<td>Simon Delaere (imec)</td>
</tr>
</tbody>
</table>
Disclaimer

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 761813. 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>Nature of the deliverable:</th>
<th>PU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dissemination Level</td>
<td></td>
</tr>
<tr>
<td>PU</td>
<td>Public, fully open, e.g. web</td>
</tr>
</tbody>
</table>
EXECUTIVE SUMMARY

This deliverable provides an overview of three HRADIO applications that will be tested during the final pilot. Building upon the learnings from the first and second pilot, the 2 applications that have been tested before, the Car and chat view application, underwent a second iteration. An already tested concept during pilot 1 called ‘Daylist’ is developed as a third working application using HRADIO components. All three applications are Android applications as the OMRI library is currently available for Android only.

The three applications each target a different scope. The first has a ‘platform centric’ scope offering most of the HRADIO features like service recommendation, substitution and time shifting, across different broadcasters. This product targets car and hybrid radio manufacturers.

The second application is a broadcaster centric application where the user can listen to different radio stations belonging to a single broadcaster. And the third is brand centric, where the focus lies on bringing a station or brand specific experience to the user. The three approaches represent the different forms in which HRADIO components may be packaged for commercialization, as well as the complementary strategic interests with the different pilot leads.

The platform centric application will be piloted as an open pilot and will be available from the Google Play Store. The brand centric app will also be made available through the Google Play Store but as a semi-open pilot. The broadcaster centric app will be distributed as an Android apk-file to the participants of a closed pilot.
# TABLE OF CONTENTS

1. HRADIO APPLICATIONS FOR FINAL PILOT .................................................. 6  
   1.1. From car-application to platform centric app ........................................... 6  
   1.2. Broadcast centric app .............................................................................. 6  
   1.3. Brand centric app implementing ‘Daylist’ ................................................. 7  
2. USER & TECHNICAL TESTING APPLICATIONS ............................................. 8  
   2.1. Technical basis for final pilot applications ............................................... 8  
   2.2. Application 1: Platform centric app .......................................................... 9  
      2.2.1. Features ............................................................................................. 9  
      2.2.2. Integrations and backend .................................................................. 13  
      2.2.3. Pilot user data .................................................................................... 14  
   2.3. Application 2: broadcaster centric app ....................................................... 15  
      2.3.1. Features ............................................................................................. 15  
      2.3.2. Interfaces history .............................................................................. 20  
      2.3.3. Integrations and backend .................................................................. 21  
   2.4. Application 3: Brand centric app ............................................................... 23  
      2.4.1. Features ............................................................................................. 23  
      2.4.2. Interfaces history .............................................................................. 24  
      2.4.3. Integrations and backend .................................................................. 28  
      2.4.4. Pilot user data .................................................................................... 29  
3. DEPLOYMENT VIA GOOGLE PLAY ................................................................. 31
1. HRADIO APPLICATIONS FOR FINAL PILOT

This document describes the user interfaces of the products and integrations of the technical components that have been developed for final pilot testing.

We start with an overview and motivation for each of the products.

1.1. FROM CAR-APPLICATION TO PLATFORM CENTRIC APP

In pilot 2, the focus of a general radio app, was the car use case. Goal of this pilot was to gain knowledge on:

- The simplicity of user interfaces and their applicability in car scenarios
- The incorporation of voice control

The car application from pilot 2 always has been designed to be a general radio application. Station search (through the HRADIO metadata platform), favourites and service recommendations as well as a seamless integration of the time shift and item skipping scenarios have been on the feature list for pilot 3 to be tested. The implementation of this extensive set of new functionalities could not be achieved with the simplistic car centric UI from pilot 2. Additionally, due to the open nature of pilot 3, the HRADIO project decided to not take the risk of simultaneously testing highly interactive features in an environment where safety is critical. Driver distraction constitutes a significant risk. Therefore, the HRADIO partners decided to go for a new approach for the application UI in pilot 3. Still designed to be a general radio application, it focuses on HRADIO centric features in an android application designed mainly for mobile devices. Portrait and landscape UI designs have been implemented. Additionally, a “clean” car mode has been implemented, which could be activated in the UI. However, in the open pilot this car mode will be deactivated for the above reasons.

1.2. BROADCAST CENTRIC APP

This final pilot will be the further development of the concept used in pilot 1 and pilot 2. The goal is to develop a fully branded application for the broadcaster partner RBB, which will be developed by Konsole Labs. Following the results of the user tests, the features skipping and time shifting were the most important personalization functionalities for radio. These features are very popular among radio professionals. Due to the high effort in workforce needed to keep a chat view alive in an actual production environment, it was decided by RBB to not further develop this feature. With the new appearance, the hybrid features are now more in focus: Time shifting...
gets a completely new intuitive interface. The player will be displayed directly in front of the start screen. Depending on the context of the audio, the different additional metadata are displayed. The status is now better visualized. Overall, the acceptance of the concept will be further established.

1.3. BRAND CENTRIC APP IMPLEMENTING ‘DAYLIST’

For the final pilot, VRT is developing an app targeting Radio 1 listeners. The focus of the app is on the combination of live radio with the concept of Daylist, of which the principles have been tested first during pilot 1 but which was not yet implemented in Pilot 2.

By implementing this third prototype, HRADIO is not only filling in a gap between one of its “products” of Pilot 1 and the final project results. It also allows:

- VRT to be in the pilot seat of a hybrid radio feature set that is of most interest to its production environment (which will develop the eventual real-world app), so that internal traction and the opportunity to valorize are maximized. This includes both the strategic focus on the brand (Radio 1) for the app and the inclusion of the daylist personalization features;

- To remain in line with operational choices made by VRT since Pilot 1: since then, Radio 1 has been chaptering their radio shows to a certain amount and has been offering these fragments since then into their current app. This current momentum at the broadcaster, where curation of this type of content is proven to be of strategic importance, constitutes an opportunity to go further on the concept of combining live with on-demand radio.
2. USER & TECHNICAL TESTING APPLICATIONS

2.1. TECHNICAL BASIS FOR FINAL PILOT APPLICATIONS

The technical libraries and API implementations of the HRADIO project will form the foundation of the further implementation of the two existing pilot 2 applications as well as the new VRT “Daylist” app. Starting with the OMRI Libraries, which handle the tasks of receiving the radio signals (either DAB broadcast or IP), decoding audio and data services and finally, enable the player implementation for time shift and skipping use cases.

On top of the OMRI Libraries, the Pilot application makes use of APIs and implementations for the following functionalities:

RadioDNS Library

Discovers and downloads RadioDNS/WorldDAB SPI/VIS/WEB signalizations and information such as station Logos, additional IP only services, programme information and other additional metadata. On top of this, application developers can use this library to handle RadioVIS STOP messages.

Time shift player

The time shift player library provides a ready to use player component for integration in Android applications. The time shift is saved as a pointer inside the time shift server. Making it possible to time shift without the use of local cache. This also makes it possible to share the current listening information (e.g. which service at what time) with other users or devices enabling cross device time shifting.

RadioWEB View

For the integration of interactive components, defined by the broadcasters and dedicated for the individual radio station that the applications are currently tuned to, the RadioWEB view offers a ready to use HTMLView and provides a JavaScript interface for the HTML pages to control the OMRI based radio services (pause, mute...). Additionally, callback methods can be added from the JavaScript code to add listener functions for radio services such as DynamicLabel+ and Slideshow.

A detailed description of the libraries and APIs is given in D3.3 “HRADIO mobile and HTML client API implementations”
HRADIO Metadata platform

This HRADIO component provides distributed station and programme search features and is used in the platform-centric pilot application for service and programme searches and recommendations for alternative radio stations. Furthermore, it is responsible for the (privacy-preserving) collection of user behavior reports which can also be visualized with the platform’s data visualization & dashboarding module. In addition, it is used for the token distribution needed for the cross-device time shift functionality. A detailed description of the platform modules and its REST-APIS is given in deliverable D3.2: “HRADIO Communication Platform”.

![Figure 1: Overview over HRADIO technical components](image)

2.2. APPLICATION 1: PLATFORM CENTRIC APP

2.2.1. Features

The following table contains the app’s features:

<p>| Feature |</p>
<table>
<thead>
<tr>
<th>Time shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rewinding up to 2 hours on what was missed</td>
</tr>
<tr>
<td>Pause content and return back to live</td>
</tr>
<tr>
<td>Control and handling: Intuitive shifting via slider (to indicate how far user can shift in time)</td>
</tr>
<tr>
<td>Skipping: Skip unwanted content/segment of a programme with an individual item from a streaming provider</td>
</tr>
<tr>
<td>Choose from user created playlist for skips</td>
</tr>
<tr>
<td>Start from the beginning of the programme, catch-up-radio, time shift</td>
</tr>
<tr>
<td>Service following: App chooses the best way (DAB / Streaming) to receive a particular service based on user preferences</td>
</tr>
<tr>
<td>Keep a single time shift-buffer during service following.</td>
</tr>
<tr>
<td>Cross-Device Time shift: Listen to your time shifted content from different devices and share your findings with your friends</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Favourites</th>
</tr>
</thead>
<tbody>
<tr>
<td>Favoriting individual items e.g. song, podcast</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transparent &amp; controllable station recommendations (settings menu provides information and configuration options for each recommender module)</td>
</tr>
<tr>
<td>Service coverage (more services are part of the platform, recommender coverage was increased due to the availability of different recommendation strategies based on audio features, keywords, trends, expert opinions)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finding services: Service search can be conducted selecting name, genre, location, programme keywords, ...</td>
</tr>
</tbody>
</table>
Federated service search: Search node filters can be specified to select a certain subset of nodes for search.

**Spotify Integration**

Replace the currently running programme with songs from Spotify.

Like/Un-like particular items in the linear programme: Like a song in the linear programme and save it to a personal Spotify playlist to remember/relisten.

Skip a song and it will be added to a ‘blacklist’ playlist

Manage this playlist from the Apps–UI

Programme Item Recommendation: Listen to a Spotify podcast that was recommended based on the content of the currently running programme

**Podcast Integration**

Play available podcast for a specific programme directly: Start to listen to a podcast from a particular programme directly from the EPG or RSS Feed.
User Data Collection

User data is collected for the pilot. Demographic data that should be used for the report can be configured in the settings menu.

Car Player

A simplified UI dedicated to in car usage can easily be accessed and activated.
2.2.2. Integrations and backend

The following platforms and backends are integrated in the platform centric app:

- **HRADIO Communication Platform:** The station and programme search implemented in the app uses the platform’s metadata search & storage module (also using the federated search capabilities as described in D3.2). The station recommendations displayed in the app stem from the platform’s station recommendation module. Pilot user data collected by the app is stored using the (optionally privacy-preserving) user report module which is provided by the platform. User reports are visualized using the platform’s integrated data visualization module.

- **EDI Streaming Server / Server-Based Time shift:** For the realization of time shift and cross device use cases the existing DAB over IP streaming server has been extended to also act as the time shift storage for the TimeShiftPlayer component from pilot 2, which realized the time shift functionalities locally on the device. The EDI streaming server for pilot 3 not only takes the existing EDI ensemble from the HRADIO broadcast partners and splits it up into single service EDI streams. It also provides a configurable time shift buffer for the application component.

- **Spotify:** The app’s programme substitution functionality uses the Spotify music streaming service for replacing the programme with songs from Spotify playlists. It is also used to recommend podcasts based on the currently running programme’s content.

The app uses the following client libraries developed in the HRADIO project:

- **OMRI:** The OMRI Libraries still build the core for all the HRADIO pilot applications. The OMRI libraries handle the general search, tune and playback functionalities for the applications as well as the notification of real time events (DL messages, SLS images, alarm announcement …) to the application. The storage of service lists along with service metadata such as logos or genre and bearer information is also in the scope of the OMRI implementation.

- **Substitution API:** The pilot application uses the installed Spotify application as a source for the substitution of broadcast items. In the case, that the listener is not satisfied with the current live content, skipping over songs or other radio items, is handled through the public APIs by Spotify. However, the choice of Spotify as the substitution provider is only one of many possible candidates and can easily be exchanges due to the abstract design of the HRADIO substitution API.
• **TimeShiftPlayer API:** From pilot 2 to pilot 3 remained the same to a great extent. Applications which used the TimeShiftPlayer component for local time shift, worked in the same way once the application has been built with the new server-based time shift player implementation. Application developers can query the time shift period, get a list of toggle items and accordion metadata and can freely skip through the buffer forward and backwards. Additionally, a new callback for notifications regarding local and server timestamps has been introduced.

• **Platform Search API:** This Android API is used to access the HRADIO Communication Platform’s functionalities (search, recommendations, ...).

### 2.2.3. Pilot user data

User behavior is tracked and sent from the App based on the actions a user is performing. This so-called user report contains information about the action itself and some context information (location, demographics, time). It is encoded in JSON and is sent to the corresponding Communication Platform endpoint. An example can be seen in the following table:

```
{
    "context":{
        "demographics":{
            "ageGroup":"FiftyOneToSixty",
            "gender":"Male",
            "location":{
                "countryCode":"GB",
                "distance":100,
                "geoPoint":{
                    "lat":48.186248779296875,
                    "lon":11.628196716308594
                },
                "timezone":"GMT+00:00"
            }
        },
        "time":"Nov 18, 2019 09:46:26"
    },
    "data":{
        "actions":[
            {
                "action":"Pause",
                "actionlabel":"
            },
            "actiontime":"Nov 18, 2019 09:46:11"
        ],
        ...,
        "id":1969772122
    },
    "description":"userreport"
}
```
For the pilot, privacy-preserving features are disabled.

2.3. APPLICATION 2: BROADCASTER CENTRIC APP

2.3.1. Features

Start Screen (Menu)

The start screen gives the user an overview of all available radio stations. The user can select the desired station and navigate to the player display for this radio.

Radio Player

As soon as the user has selected their favourite station, he starts in a player centered view. Compared to pilot 2, the radio player receives a more prominent position. Context information is displayed below the player. The player consists of the important controls in the center (play/pause/substitution/time shift). Above the controls, details about the current content are displayed, such as information about the current show. In between there is a new horizontal time shift slider. The slider represents the waveform of the audio with event markings.

Depending on the actual state, the user can start and stop the stream or skip (substitute) the actual item.
**Channel screen**

The Channel screen provides a way to see all important information about the currently played stream. This includes information about the currently played song,
show or a general segment. The information will be displayed to the user via multiple UI elements.

**Time shifting**

There are three ways to activate time shift mode.

- Time shifting by clicking on song elements at the playlist below/Time shifting metadata-based:
  During playback of the DAB stream, all associated metadata is collected and stored. This allows for skipping (time shifting) at certain times within the stream being played, e.g. the beginning of a song.

- Slide the horizontal slider and select a time position.

- Click on one of the player buttons / Time shifting time-based: Possible time-based time shift interactions are: pause/continue, jump 30s back, jump to the beginning of the stream, jump to the live stream.
Substitution Feature

Replace the currently running programme with songs from Spotify. Bookmark particular items in the linear programme: Bookmark a song in the linear programme and save it to a personal Spotify playlist to remember/ relisten them. Skip a song and it will be added to a 'blacklist' playlist.

Figure 4: Mock-up of Substitution Feature
Podcast Integration

The user can access a list of available podcast series via the navigation point "Podcasts". As soon as the user selects a series, the episodes are displayed with the latest episode in focus. As soon as the episode is selected and played, additional information about the episode is displayed.

Figure 5: Mock-up of Podcast Integration

Bookmarks

Via the bookmark button the user can mark their favourite songs and save them in his own playlist. The playlist is created on the user’s Spotify account.
D4.4: Report on final user interfaces and integrations

Featured Content

A carrousel will be provided with the last played news segments available in the stream. General News, Traffic News and Weather items will provide a shortcut to the exact position in the live stream of the last played segment of this type.

2.3.2. Interfaces history

Pilot 1 was tested based on the initial concept, a first set of technical requirements and initial mockups. Audio clips were presented in a list view as shown below. The initial idea was to change the order individually and to store the user-defined order automatically in a personalized programme scheme.

Figure 7: RBB Pilot 1 design
Following the users input pilot 2 was developed. The radio station screen is built as an information feed to make it highly customizable with different features in one view.

2.3.3. Integrations and backend

The following platforms and backends are integrated in the broadcast centric app:

**EDI Streaming Server / Server-Based Timeshift**

Based on the EDI DAB-over-IP Technology from Pilot 2, the functionality has been extended from a purely local service to a server based service. The EDI stream hosting server now stores an ongoing buffer which enables timeshift functionality for the user without the need to having listened to the stream locally beforehand.

**Spotify**

The previously used local-storage-only substitution functionality has been replaced with a Spotify based substitution. The goal is to provide the user with a multitude of substitution services, starting with one of the most popular music streaming services: Spotify. To achieve a Spotify based substitution the official Spotify streaming API has to be utilized. After logging into the Spotify service, the user can select one of their...

Figure 8: RBB Pilot 2
previously created playlists. The Spotify substitution service can now randomly select one of the titles in the selected playlist as substituted songs.

**RadioDNS**

The previously in pilot 2 used RadioDNS service will be used in the same capacity in pilot 3. After a RadioEPG lookup specific metadata like current show title and description will be used to enrich the information provided to the user.

**RBB–Konsole Labs Interface**

To provide the user with additional metadata a private IP-based service between RBB and Konsole Labs will be used. It’s goal is to act as fallback should other services like RadioDNS or DAB DL+ not suffice in supplying metadata.

The app uses the following client libraries developed in the HRADIO project and are described in 2.2.2:

- OMRI
- Substitution API
- TimeShiftPlayer API
- RadioDNS
2.4. APPLICATION 3: BRAND CENTRIC APP

2.4.1. Features

Personalized list of on-demand radio

The brand centric app builds upon the daylist concept which is a personalized playlist of Radio 1 fragments. 2 methods of personalization are implemented, and A/B tested in the final pilot.

A. The user selects the amount of time he/she has and the categories of interest.

B. The user gets recommendations based upon his listening behaviour. For this VRT is using a recommendation engine developed within another H2020 project, called CPN, VRT is involved in.

OMRI DAB radio player

The app uses the OMRI library for listening to Radio 1 over DABoverIP.

The OMRI library offers the following features that are used in the app:

- **Show current item**: DL+ labels are used to send artist and song information over DAB. These are decoded and shown in the app. Togglebits are used to create different items so the user can see a history of what has been playing.

- **Pause radio**: the user can pause the radio show and start where they left off.

- **Time shift**: By clicking on a created item, triggered by the togglebit, the radio is time shifted to the location in the buffer where that item started.

- The current **slideshow** image is shown as a spinning record on the radio detail screen. Once a togglebit has occurred and an item as been created, the current image is used as a thumbnail for that item.

RadioEPG

The EPG from Radio 1 is fetched from the corresponding RadioDNS service. This way, the app can show more details on the current programme.

In the EPG data, we have also added categories to each radio show that correspondent to the categories that are linked to the stored radio fragments.

Item suggestion on live stream

By using the metadata that come with the programme details from RadioEPG, the app shows relevant radio fragments related to the current live radio show.
Favourites

On-demand radio fragments can be added to a favourites list for easy reference.

2.4.2. Interfaces history

Concept version for first pilot

During pilot 1, the concept of ‘Daylist’, has been tested using mock data.

In this first version, the user was able to select their categories by tapping circles. This was a way to give weight to a certain category in relation to another category. The mock data was a list of radio fragments that were also given weight.

This approach was received very well by participants but is quite duty heavy on editorial side as there exists no such a system of categorization at VRT that can do this automatically.

The actual list itself was received with mixed responses. From UX perspective it was quite clear, but the overall design was perceived as functional but bland.

Figure 9: Concept version for pilot 1: on the left; category selection, to the right; the daylist
Daylist App

For pilot three, an Android app with actual and live data is developed by VRT. Due to practical reasons, some changes have been made. In this version, the user will only choose categories in an all-or-nothing kind of way. No weights can be defined in the interface. This is because this kind of metadata will not be available on the live fragment data during the pilot period.

The user can also select the amount of time they have. This will limit the list of items to be of that specified length.

Another difference is that we added the OMRI radio player with time shift capabilities.

As this is supposed to be a working Radio 1 app, it is logical that the user can listen to live radio.

Pre-pilot evaluation and final version

Before the start of the pilot, during development, there was a period of pre-pilot evaluation by friendly users to collect the first feedback. This version had a wheel to select the amount of time that the playlist should be. This wasn’t clear or didn’t work as expected from a UX point of view for most users.
The minimized player on the bottom shows what is playing now and could be maximized to show more info. This spawned another comment that it was not possible to see at a glance what was on live radio while listening to an on-demand fragment. The only way to know was to stop listening to the fragment. There were also no controls on the minimized player, so the user needed to maximize the view to be able to pause or stop.

Apart from minor UI remarks, another big change was the daylist screen itself. There was too much generic information on the top of the screen. The actual list, which is the main component of the app was pushed down. This led to a much more streamlined design and also to the removal of the latest news, traffic and weather feature.
Figure 13: Left first design of the daylist screen, Right: Final design
2.4.3. Integrations and backend

**EDI HTTP Server**

Since pilot 2, VRT has an instance of the EDI HTTP server from IRT running at http://edihttp.vrt.be:8187. This is updated to the newest version which has time shift capabilities.

There was some development needed to upgrade the Dynamic Labels to Dynamic Labels +

The communication between app and EDI server is contained within the OMRI library.

**RadioEPG**

The RadioDNS services of VRT are hosted at metadata.radio. The services have been moved from EBU because of the RadioWEB support. The RadioEPG is a static EPG at the moment that will be updated during the course of pilot 3.
The app does a RadioEPG lookup in order to know the current playing radio show. Based on the meta-data of the current show the app can give suggestions of on-demand fragments that are related.

**Recommendation engine**

The Daylist app is using an in-house developed recommendation engine for the fragments. This development was done in the context of the CPN project, another H2020 project around personalized news.

Every time a user listens to a fragment, the internal ID of the fragment and the user ID is logged. With a heartbeat of 5 seconds, an update containing the percentage listened is sent to the recommender in order to recreate the model for this user.

**2.4.4. Pilot user data**

The Daylist app uses several methods to track different types of data.

**Adobe Launch**

This tag manager directs events to 2 different points: Adobe Analytics for anonymous page views and the Peach platform from EBU to track media events.

Following events are tracked with Peach: media_start, media_pause, media_stop, media_like and media_seek.

This is an example of the payload we send with the events:

```json
{
    'type': 'audio',
    'touchpoint_brand': "daylist",
    'format': "on-demand|live",
    'categories': "list of categories",
    'show_id': "internal ID of the fragment",
    'title': "title of the fragment or radio item",
    'duration': "length of the fragment",
    'date': "publication date",
    'programme': "programme the fragment was aired in",
    'programme_live': "live programme",
    'test_stream': "A/B test group",
}```
Adobe Qualtrics

There is an integration with Adobe Qualtrics to serve up a small two weekly survey to the user. Along with the responses, the e-mail of the respondent is also stored.
3. DEPLOYMENT VIA GOOGLE PLAY

Google Play Store is an app store that lets you install apps and games on Android smartphones, tablets and Android TV. The apps offered come almost exclusively from third-party companies and free software developers. Both free and paid apps are offered, with free apps outnumbering paid apps.

For the deployment of the HRADIO applications, an HRADIO dedicated Google Play Store account has been set up. Normally, the aim of the Google Play store is to distribute apps to millions of customers and handle the payment for commercial applications. For developers there is a one-time registration fee and the possibilities to deploy applications to a closed user group for testing only.

Daylist and the platform centric app will be launched as an open Beta test.

<table>
<thead>
<tr>
<th>App-naam</th>
<th>Actieve installaties</th>
<th>Nieuwe Google Play-update</th>
<th>Laatste update</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>DabMpg23Plugin de.int.dabmpg23decoderplugin</td>
<td>2</td>
<td>—</td>
<td>10 mei 2019</td>
<td>Gepubliceerd</td>
</tr>
<tr>
<td>Daylist be.vrt.daylist</td>
<td>2</td>
<td>—</td>
<td>8 dec. 2019</td>
<td>Niet meer gepubliceerd</td>
</tr>
<tr>
<td>Daylist be.vrt.pilotzone.daylist</td>
<td>68</td>
<td>—</td>
<td>10 dec. 2019</td>
<td>Gepubliceerd</td>
</tr>
<tr>
<td>HRadio tms.hradio.hradioskcase</td>
<td>8</td>
<td>—</td>
<td>16 okt. 2019</td>
<td>Gepubliceerd</td>
</tr>
<tr>
<td>HRADIO Chat rbb eu.hradio.phoneapp.rbb</td>
<td>7</td>
<td>—</td>
<td>12 sep. 2019</td>
<td>Gepubliceerd</td>
</tr>
<tr>
<td>HRADIO Chat VRT eu.hradio.phoneapp.vrt</td>
<td>13</td>
<td>—</td>
<td>17 jun. 2019</td>
<td>Gepubliceerd</td>
</tr>
<tr>
<td>HRadio SPI Sample int.de.spiasample</td>
<td>6</td>
<td>—</td>
<td>10 mei 2019</td>
<td>Gepubliceerd</td>
</tr>
</tbody>
</table>

Figure 15: Google Play console with HRADIO sample and chat pilot apps