MARCONI: Towards an Integrated, Intelligent Radio Production Platform

Rik Bauwens
Sandy Claes

VRT, Belgium

Abstract

We develop integrated, intelligent tools for the creative industry to boost radio-making into 21st century experiences.

Establishing conversation between radio producers and their listeners has been part of radio broadcasting for a long time: first via telephone, then via text messages, e-mail and social media. Today, dedicated smartphone radio applications allow us to interact on a more personal level; e.g. through chatbot services. Radio producers are now able to make a real-time connection with listeners not only on-air, e.g. by reading their text messages or calling them directly, but also offline by answering them instantly and personally via their radio consumption applications. This trend introduced a new challenge: listeners expect and demand personalized feedback and radio teams need to be able to cope with this. Moreover, radio stations (often unconsciously) collect data of their listeners, but do not fully exploit its potential. In the MARCONI (i.e. Multimedia and Augmented Radio Creation: Online, iNteractive and Individual) project, we have developed a platform with an integrated set of tools for radio producers to deal with these issues and prepare them for the next-generation 360° radio experience. The platform offers tools ranging from chatbots, curation systems and search engines, to the integration of user-generated content in the rundown of the actual radio show. By working together with producers in their own environment, we were able to make the workflow and user interface of the MARCONI tools as fit as possible. We also gained insights into the current technical workflow of radio systems, which allows us to seamlessly integrate the MARCONI platform. To stress the importance of the integration possibilities of the MARCONI platform, we will organize a 6-month piloting period with external radio stations, starting September 2019. This paper focuses on our iterative design and development process, the evaluation of results, feedback and future plans to be exploited during the piloting period and beyond.

Introduction

Today, the live radio experience is increasingly augmented with social and other interactive media. For instance, a radio station might organise daily polls on Instagram, or provide the possibility to chat with their DJ's through WhatsApp or their own application, such as VRT's

dedicated Radio Apps [2]. Instead of serving single listeners on the actual radio show, e.g. via an on-air interview, radio producers now also serve listeners off-air. The distance between radio producers and their listeners is thus decreasing, resulting in higher workload for the producers to manage these interactions. Unfortunately, the tools to enable radio-makers to capitalize on these new kinds of media and interaction opportunities are premature at best, and are in addition ill-aligned with contemporary radio production workflows. The DJ's desk at the radio station often represents what in software design is referred to as 'spaghetti'; a cluttered, unconnected collection of screens displaying the show rundown, phone calls, SMS, social media and so on.

In this paper, we will present the design and development of MARCONI [1], which aims to build a smart and unified editorial application, which at all times shows relevant information, tailored to the radio station. MARCONI also wants to offer a unified API to make it easy to integrate both studio workflows and new external services. MARCONI also includes five elements to support radio producers in listener interaction processes: (1) a conversational interface, (2) a search interface, (3) an automations interface, (4) a curation interface and (5) a poll interface. Through different iterations, MARCONI has been deployed by 3 different radio stations of VRT, which is the public broadcaster of the Flanders region in Belgium; i.e. Studio Brussel, MNM and Radio 1. We report on the iterative design and evaluation process and present the results of this study. The work described has been developed within the EU-funded H2020 ICT project MARCONI [1].

The paper is structured as follows: first we give a short overview of how we organized the development process. Second, we describe the different steps of this process. Then, we deepen into the actual technical construction of the five elements described above and finally, we conclude with identifying future opportunities.

Background

Because radio production is changing the way they create and distribute content, technological solutions also need to change at a fast pace. Some broadcasters are currently outsourcing their infrastructure to all-in solutions, but in the process locking out other third-party solutions and creating a vendor lock-in on their choices.
Another approach is to develop the infrastructure internally, within the radio stations, by connecting multiple software solutions manually through a middleware layer. The problem with the latter approach is that this is hard to maintain and broadcasters need a big team to maintain the integrity of the infrastructure decisions. Most broadcasters end up with many tools that need their own accounts and need to be used simultaneously by different teams. One graphical interface to rule all the services at one time is often wanted but very hard to create with the current state of services.

Incremental approach

Throughout the design and development process, we deployed a user-centric approach. More specifically, before actual development was started, we organised observations at one of VRT's radio stations (i.e. Studio Brussel) and held co-design workshops; one focused on radio producers and one on listeners (which results are reported in reference). Different stakeholders of VRT (who also participated in the co-design workshop) were then consulted

on a regular basis to inform and influence the direction of the design and development of the prototypes iteratively. The prototypes were also implemented *in-the-wild* [3], i.e. the production activities of VRT's radio stations. This way, we were able to collect real-world feedback on our developed prototypes.

To support this incremental approach, the prototype interfaces are built with different user interface (UI) components, which offer user-facing functionalities and act as building blocks that can be integrated to yield more multi-faceted, composited UIs. The advantages of such a modular, component-based design that is centred around loosely coupled services include:

- isolated development and testing of individual features;
- increased flexibility and modularity (i.e. substituting the implementation of a service will not impact the operation of the services that depend on it);
- the ease of composing complex UIs by mashing-up verified and validated components.

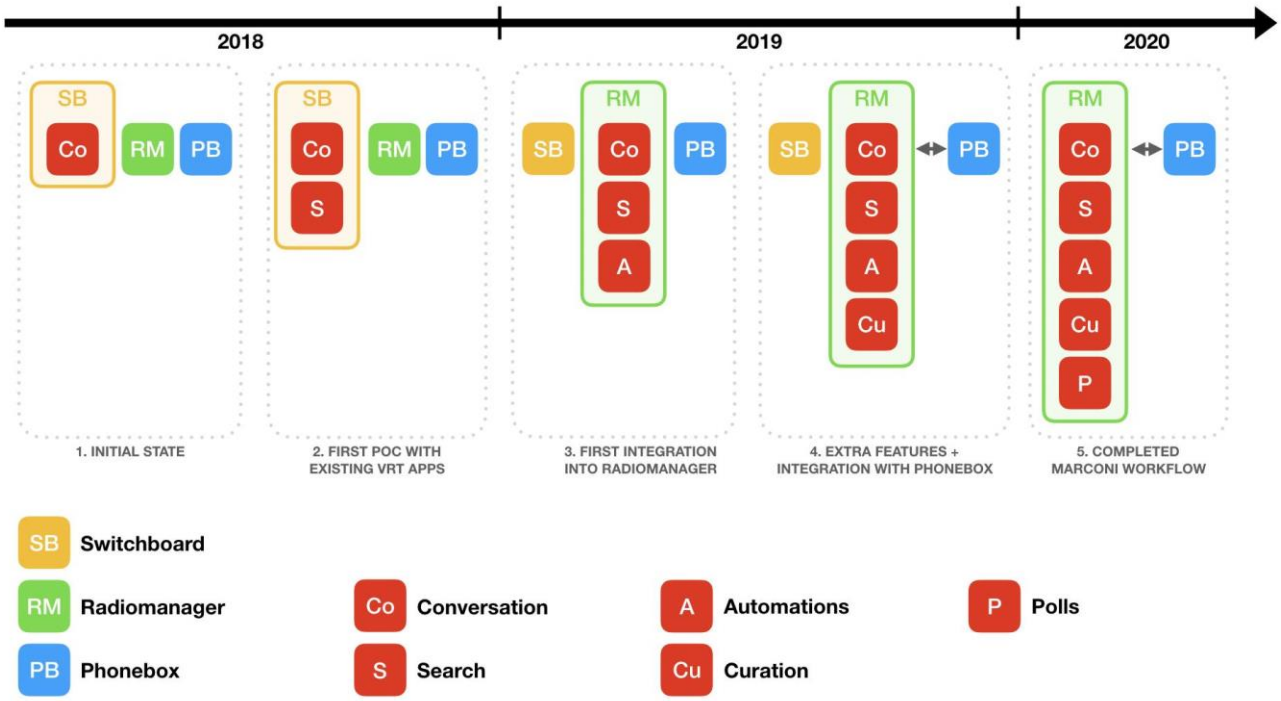

Figure 1. Timeline of the development process, including the different iterations of the five interfaces.

A timeline of the different steps in this development process is outlined below in Figure 1, and discussed in more detail in the following:

1. Before project MARCONI started, VRT already deployed three different applications to create radio programmes: (1) Switchboard to handle user interactions of the app via a simplified Conversation Interface, (2) Radiomanager to prepare the rundown of radio

shows and (3) Phonebox to handle interactions via SMS and Twitter. In previous work, and as an initial step within the MARCONI project, we report on observations of how radio DJ's use this variety of tools and how this slows down their activities, directing attention away from focusing on the actual listener. In co-design workshops, they also provide solutions to overcome this in one integrated tool [4].

2. Working towards this integration, we had set up a first study to design and develop the Search Interface. A first working prototype that included basic filtering functions (e.g. type of content, date, geographical location of the message) was presented to 3 radio DJ's of radio station Studio Brussel. Based on their suggestions, we further refined the functions, e.g. to allow them to add labels to the messages. Then, we asked the radio production team of both radio station Studio Brussel and MNM to deploy the second working prototype in their daily activities during one month. This prototype was not integrated within the existing tools and thus required effort of the production team to deploy this prototype. The team regularly reported errors to our development team via e-mail. We also organised individual interviews with the lead digital radio producers of both Studio Brussels and MNM (2 persons, male) to evaluate their and their teams experience with the prototype. Although they were initially very interested to deploy a search function, they did not use it during a live radio broadcast, only during their pre-production phase. The deployment of an additional prototype caused them to open too many browser tabs simultaneously. For evaluating the prototype in a live setting, they expressed it was crucial to integrate this search functionality in Switchboard. Concerning its use in pre-production, they deployed the search function for combining messages, e.g: "When I saw at 6 o'clock that Axel sent something, at 7:30 I don't have to scroll back in the feed to look for that message, but I can just search for 'Axel'." Then, they also had suggestions towards the design of the interface, e.g. "What I didn't find very straightforward, was the little clock to adjust the time. I think a digital clock is easier to use than an analog one. But this may be personal taste." The producers also mentioned the prototype lacked the ability to search for images, e.g #animalday, only show images (images are always separate images as of now). They mentioned that it would be nice to have the ability to mark all found messages to reply.

   Finally, the resulting Search interface was integrated into Switchboard. As Switchboard is built and developed in-house, it makes it easy to modify in a quick way.

3. The next step was deciding on the main architecture of the MARCONI backend and setting up the database and the APIs. Once the main data structure was laid out, we started unifying user interfaces as well. In the end, it was the aim to reduce the number of screens and applications and create a connected production environment. We chose Radiomanager to be this unified, connected interface for two reasons: Radiomanager already bundles a lot of features designed for radio-makers, including the core functionality of creating the rundown of a radio show. Secondly, it supports adding custom features as plugins, which makes it easy to try new functionality.

   Steps 4 and 5 are not integrated yet, however, they give an insight into our future development plans:

4. With the basic integration of the Conversation Interface, Search Interface and Automations, we are able to start connecting additional services. The most obvious is the phone service, provided by Phonebox. We will also connect to the visual radio environment. This introduces a new interface to decide which content could be used for this purpose: the Curation Interface.

5. Lastly, we will look into the possibility to automate contests in a new Poll Interface. This will mark the end of development features within the MARCONI project as well. By that time, we are confident to discard the Switchboard interface and reduce the use of Phonebox by limiting its use to an API to send and receive text messages. All functionality, including new features, will be integrated into Radiomanager as plugins, making them usable by other radio stations in Europe as well.

The MARCONI platform

The core of MARCONI is built on top of GraphQL technology. Pluxbox created a framework that is able to create and maintain complex infrastructures with much more ease. By combining newly created microservices like messages and full blown 3rd party solutions as Chatlayer [5], we have created a single maintainable solution for developers, sysops and infrastructure architects. Thanks to the open source frontend framework we can create custom solutions fast and give the radio broadcaster the full power to alter and create new solutions on top of the framework.

The platform offers the data subject easy access to his existing data, its sharing status and the datasets that have been actually shared. At any time, the data subject can change the share status and conditions of any data at will.

Privacy as a core aspect is addressed with several elements of the MARCONI approach. The move from an anonymous service to a personalised platform with evident advantages for the user must be based on a best-practice privacy policy. The data subject has to give informed consent if they would like to participate through this interactive radio platform.

The framework is based on privacy by design and built on top of PriVaults technology. Profile information can be progressively stored and used for the purposes they are intended while giving the organisation the flexibility to alter these in time. End users like listeners have control over their information and logs about access are created on the fly.

Five interfaces

We handle a component-based approach for the UI development in MARCONI. Every aspect of the UI is a custom MARCONI element (e.g. textfield, button). An interface is a composition of one or more of these components. The reasoning behind this way of working is that every radio station wants to handle this interaction with their listeners in a slightly different way. By offering every component separately, a radio station can easily compose custom interfaces

for their needs. We also plan to open-source these components, allowing the radio maker community to conduct even further experiments with them.

As a basis, we offer the five boilerplate interfaces from the diagram. Below, we lay them out in more detail.

The Conversational Interface

The Conversational Interface is the basis of all communication with listeners of the radio station. This interface is focused on a live experience for the radio team: the default view is a feed of messages coming in right now. It aggregates messages from several platforms (originating from the app, but also SMS and social media posts), abolishing the need for multiple screens. This feed can be customized per radio programme by creating several Lanes with specific filters.

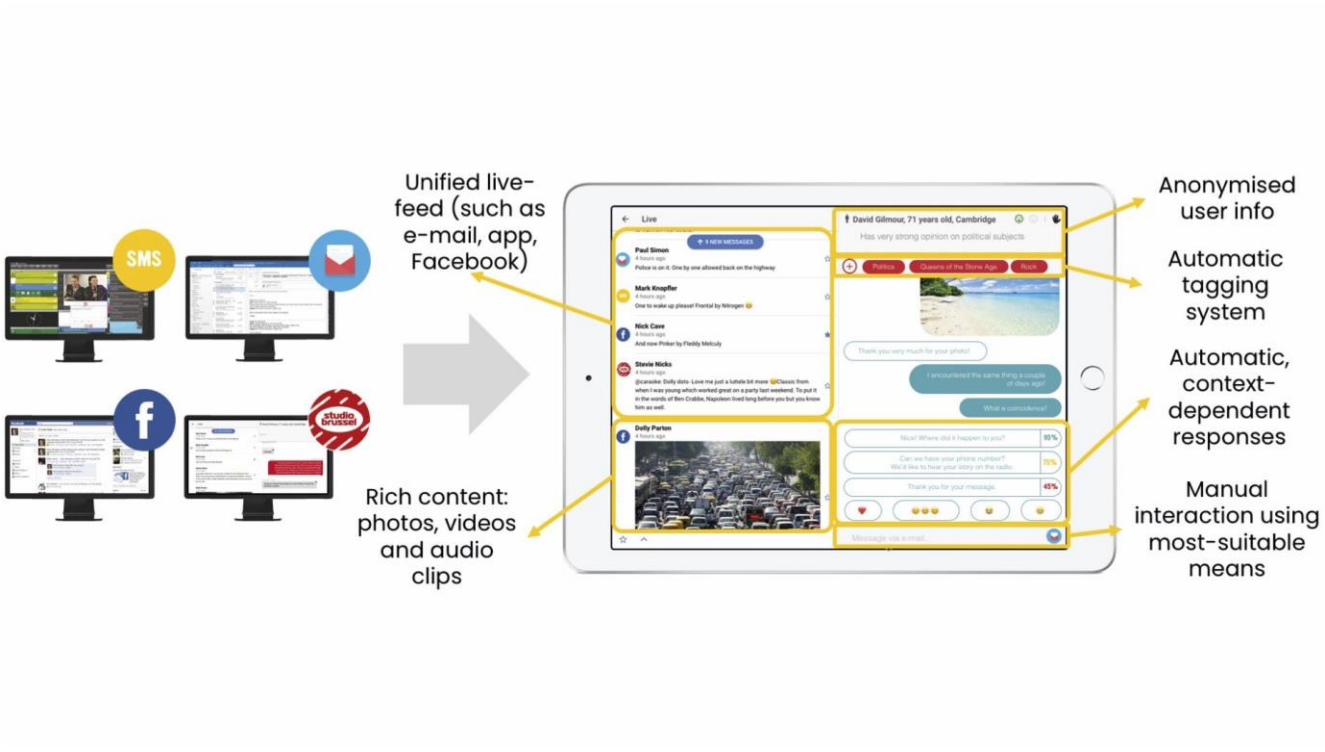

Figure 2. Illustration of the current state with a different, unconnected window per communication channel (left) versus the integrated view developed as the Conversation Interface (right).

Messages have multiple options as well: jump into a conversation with the sender, show user info and go to linked content. The aim for this interface is to be able to handle content quickly and to organise it in an efficient way. To help create radio programmes with user-generated content, messages can be linked to radio items from a rundown. This makes it easy to build a story and to ask a listener follow-up question on a radio item if necessary.

When a radio editor wants more context on a message or post, s/he can zoom into the actual conversation behind. A chat interface shows past communication between the listener and the radio station. A radio team member can type an answer manually, but also click a

suggested answer which is partially derived from the conversation context. By using these smart suggestions, an editor can quickly go through conversations with listeners and still offer them the personalized service they expect when going into a two-way conversation.

The Search Interface

Some radio programmes, however, depend on research prior to the live show. To cope with this, MARCONI also offers mature editorial tools. All conversations, texts and posts can be searched through effortlessly using a Google-like search interface. If needed, extra parameters such as a timeframe can be provided.

The Search Interface is developed by IN2, who integrated their powerful search core with MARCONI. The implementation is done using Apache Solr and thus Lucene. This is a very reliable and scalable technology used by most text search engine applications. For each radio station we create a full-text index and several indexes for filtering and faceting results. The full-text index is primarily used as part of a search request, while facets allow us to refine a result set based on select characteristics. As part of the indexing process compound word splitting and stemming is performed together with tokenizers to break words that e.g. contain numbers. Furthermore, we extract and index mentions (e.g. @username) and hashtags (e.g. #hashtag). Depending on the application use case we can activate domain-dependent stopword lists, synonyms and protected word lists. Furthermore, the service allows for running multiple environments in parallel. The Search Interface is flexible in terms of the data model that it can handle and the type of facets that it can provide alongside the search results. The user can search by entering either simple queries (e.g. keywords, @usernames, #hashtags) or by using complex queries (e.g. using boolean operators, nesting, wildcards, facets).

As with the Conversation Interface, editors can easily drag and drop search results into a rundown of a radio show and use them to enrich their radio programme. Using facets, trends can also be detected in search results.

Automations

Automations are where all the advantages of having connected data come together. Based on this data, MARCONI can take over a lot of the repetitive tasks normally carried out by the radio team and do extra things the radio team isn't able to do right now.

Bots

Allowing interaction with the radio station also means more editorial work for the radio team, because (most of the) listeners will expect some feedback if they share something. Bots are a tool to reduce this workload (e.g. to help with repetitive tasks) or to simplify interactions listeners have with the radio station.

<u>As a conversational search for listeners</u>

A case where chatbots can be useful for the radio station's listeners is searching with fuzzy details. For example: a listener recalls a song s/he heard some time ago but can only remember some vague details. Scrolling through playlists to find this song can be very time-consuming. Here, a conversational interface can help to narrow down possible matches.

Because the conversation can also handle rich content, previews of the possible matches are shown within this conversation to ensure the correct fragment is quickly found.

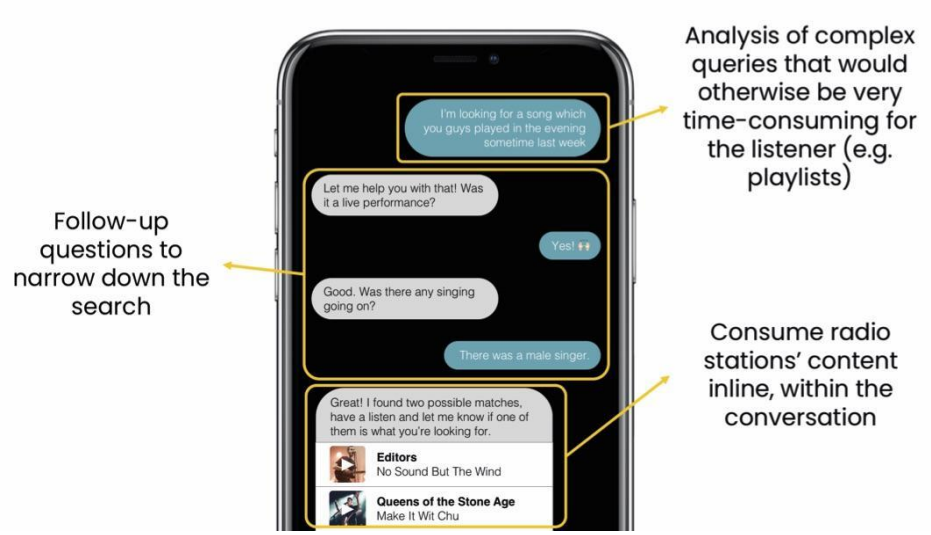

Figure 3. Example of a conversation between a radio listener and a chatbot in which the listener is in search of a particular song aired on the radio.

<u>As a digital assistant for the editorial team</u>

Most of the time, a conversation between editor and user follows a fixed pattern, e.g. asking for a consent. For these patterns, a chatbot can be very useful to lower the workload of the editor. If a user sends in an interesting story, for example, the editor can hand over the task of asking for consent for using and storing the user's personal details to the bot service. The bot will initiate a short conversation from the editorial side to ask the user for his/her consent. To make it transparent for the user, this will be reflected in the settings of the app.

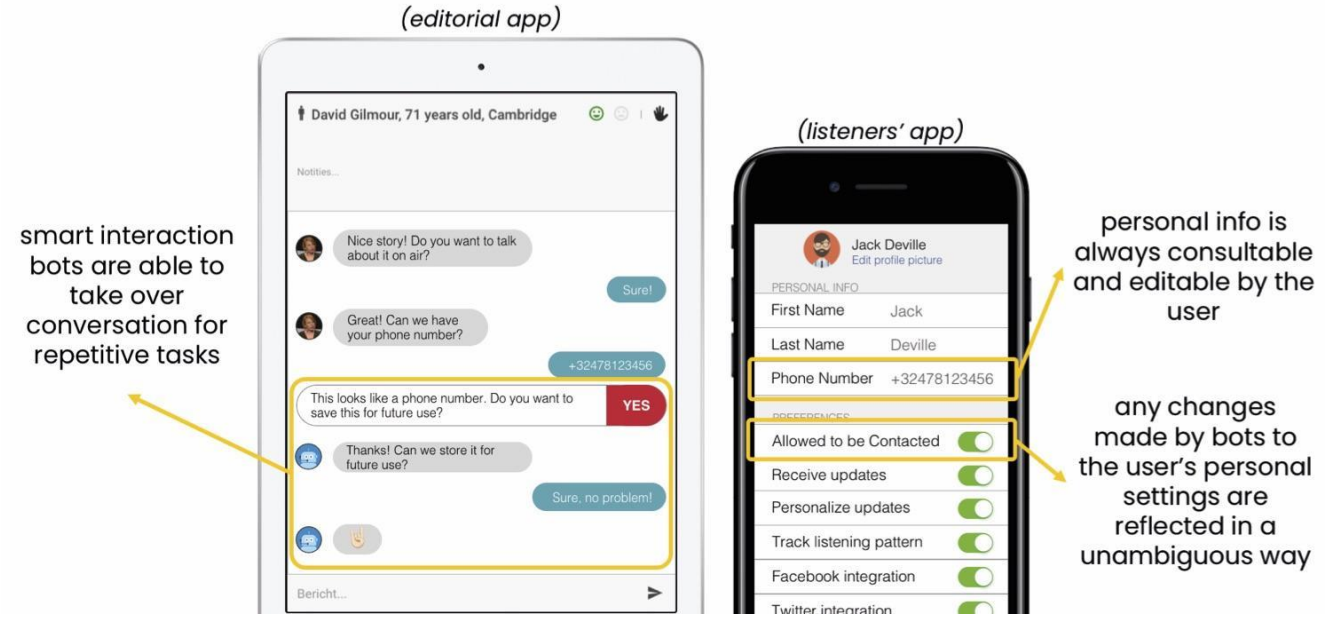

Figure 4. Example of a radio editor calling in the help of a bot to handle a conversation with a radio station's listener to ask for his/her phone number (left). The result of this conversation is reflected in the profile of the listener in the app (right).

The Curation Interface

MARCONI handles text as well as indexing audio-visual material (e.g. photos submitted by listeners) and classifies it in automatic way. For the use of this content on visual radio, it must be curated. This is done in the Curation interface. It *indicates* which photos are probably safe to broadcast and the ones that are not. However, the final decision of broadcasting is still up to the editorial team. The tool makes it as straightforward as possible by placing the most qualitative content first. It can also assess the visual quality and recognize faces and objects, which can be searched through using the Search Interface. Content analysis and classification is carried out by Joanneum Research (JRS).

For face detection and recognition, we integrate a state-of-the art approach [6], that is robust to partial occlusion and able to detect small faces. The recognition approach supports quickly adding new faces (e.g., after a user made a correction). This is achieved by making use of features obtained from deep learning, but using online random forests (as proposed in [7] as a lightweight classifier that can be trained quickly and with a single or few examples). With around 5 examples we achieve comparable performance as other methods on standard benchmarking sets, but requiring significantly lower computational effort for training. The approach also enables automatic training of unknown but repeatedly occurring persons, which can be associated with names later be performing similarity search in the face set.

Searching and organising images benefits from simple and well-structured metadata, such as tagging objects in visual content. The performance of automatic classification of objects in

images has significantly improved with the use of deep convolutional neural networks (CNNs). While some of these approaches have high computational requirements even for inference, fast and lightweight approaches have been proposed. We have implemented an object detection approach using YoloV3 [8] and adding tracking of objects for further robustness.

In cases where classification is not feasible (e.g., lack of training examples up front, no distinct classes but a continuum of content that can be grouped differently), matching based on similarity is a powerful tool to determine (partial) similarity of content items, which can then be used as needed in the application, e.g., for example-based search or clustering. We integrated an approach based on compact video descriptors [9], which has also been standardized in MPEG CDVA, thus providing interoperable descriptors. Lossless and lossy compression of visual descriptors results in descriptors having about 40% of the size of keyframe-based descriptors at the same performance (84% true positive matching rate at 1% false positive rate). The descriptors also include feature descriptors obtained from deep neural networks, which can be represented very compactly.

One important cue for organising content is the location where it has been recorded. While geotags are useful (if present), they often lack the required precision to organise visual content recorded in close proximity, e.g., to discriminate between stages at a festival, or between indoor and outdoor shots at one location. The visual descriptors are used to determine the similarity of the background for items or segments of visual content. As the relevant grouping depends on the content set, the resulting similarity scores can be used to perform clustering of the content, that can adapt to the diversity of the incoming content.

Figure 5. Conceptual sketch of a 'lively' production environment in which listeners' interactions are displayed in the actual radio studio.

By using the analysis results above, we are able to recommend user-generated content that

is probably good for use by the radio stations. This helps to minimize the time spent on content curation, which is important because we want to stress the live experience of radio by using this user-generated content. Instead of a grey, dull background banner in the studio, the aim is to equip it with screens which display relevant, context-aware content. This can be content provided by DJ him/herself, but also feature user-generated content. When a poll is currently going on, live results will be displayed there. This makes the radio show not only interesting to listen to, but also interesting to watch. A larger audience can be involved as well, e.g. by broadcasting a video stream of a show on Facebook Live and using input from that platform as well.

The Poll Interface

To spur interaction with listeners, MARCONI also maximizes the potential of the app and social media for live radio making. An example of this are polls. Polls are used frequently during a radio show: e.g. Which song do you want to hear? What's your opinion on this (pro/con)?

MARCONI created a dedicated Poll interface to help with the handling of these polls. Firstly, an editor can add a poll item to a radio show's rundown. When creating a poll to vote for song A or B from a certain artist, for example, MARCONI goes looking online for quick facts about that artist. These facts are displayed to the editor, who can highlight the most striking ones. Next, s/he can pick two songs from the MARCONI proposal and complete the preparation.

During the radio show, the sidekick of the DJ can start the poll from within MARCONI and follow the results live. These results can be also sent to the screens hanging in the studio for listeners who are watching the live stream. Moreover, the DJ can weave the highlights the editor previously selected into his/her radio show whilst the voting is ongoing. When the poll has ended, the sidekick can send a custom message to the A and B group to conclude the item. Aside from the straightforward and automated flow to create these polls, the voting data is also used by MARCONI. Preferences for artists and songs can be deducted from the results and are stored in the audience database (if a user has given his/her consent).

Conclusions and Further Work

The five interfaces described in this paper are currently still in development. Because of our incremental and in-the-wild approach, they are put into the real world as fast as possible within VRT radio stations (amongst others) and are continuously refined. As a result, radio makers support our project. Every interface consists of smaller components which allows technical partners that are involved in developing this platform to work on particular components without influencing the entire workflow. Starting September 2019, we will organize open pilots in which we invite other radio stations to test these interfaces, and evaluate how complete and modular they are. This phase will allow us find features we might not have identified and help finalize the MARCONI platform.

Acknowledgements

THE RESEARCH LEADING TO THESE RESULTS HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON 2020 RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NO. 732461 MARCONI ("MULTIMEDIA AND AUGMENTED RADIO CREATION: ONLINE, INTERACTIVE, INDIVIDUAL").

References

[1] The MARCONI project (acronym for Multimedia and Augmented Radio Creation: Online, iNteractive, Individual) aims to bring radio experiences to the next level by enabling fully interactive and personalised radio solutions, integrating broadcast radio with digital and social media. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 761802.

[2] VRT has a dedicated app for each of its radio stations: https://itunes.apple.com/be/developer/vrt/id337708002 (iOS) https://play.google.com/store/apps/developer?id=VRT (Android)

[3] Chamberlain, A., Crabtree, A., Rodden, T., Jones, M., & Rogers, Y. (2012, June). Research in the wild: understanding'in the wild'approaches to design and development. In Proceedings of the Designing Interactive Systems Conference (pp. 795-796). ACM.

[4] Claes, S., Bauwens, R., & Matton, M. (2018, June). Augmenting the Radio Experience by Enhancing Interactions between Radio Editors and Listeners. In Proceedings of the 2018 ACM International Conference on Interactive Experiences for TV and Online Video (pp. 215-220). ACM.

[5] Chatlayer is a technology stack developed by Faktion for creating custom chatbots.

[6] Winter, Martin, and Werner Bailer. "Incremental Training for Face Recognition." Proc. of International Conference on Multimedia Modeling, Thessaloniki, GR, Jan. 2019.

[7] Amir Saari, Christian Leistner, Jakob Santner, Martin Godec, and Horst Bischof. On-line random forests. In Computer Vision Workshops (ICCV Workshops), IEEE 12th International Conference on Computer Vision, 2009.

[8] Redmon, Joseph, and Ali Farhadi. "YOLOv3: An Incremental Improvement." arXiv preprint arXiv:1804.02767 (2018).

[9] Werner Bailer, Stefanie Wechtitsch and Marcus Thaler, "Compressing Visual Descriptors of Image Sequences," in Proceedings of the 23rd International Conference MultiMedia Modeling, Reykjavik, IS, Jan. 2017.