MUSIC DISCOVERY IN SPOTIFY WITH RAMA

José Bateira[1,2], **Fabien Gouyon**[1,2], **Matthew Davies**[2], **Marcelo Caetano**[2]
[1]Department of Informatics Engineering, University of Porto
[2]Sound and Music Computing Group, INESC TEC
ei10133@fe.up.pt {fgouyon,mdavies,mcaetano}@inesctec.pt

ABSTRACT

This demo presents a Spotify App [1] of the RAMA [2] system that aims to improve the way music recommendations and artist relations are shown in Spotify [3]. This new version includes the majority of the existing features from RAMA [4] including: editing the visualization parameters (to obtain more detailed graphs); adding and removing node artists (for users wishing to customize the graph); and visualization of the tags/genres that describe artists in the graph. The metadata used in the application is provided by the Spotify and Echonest APIs [5].

1. MUSIC DISCOVERY TOOLS IN SPOTIFY

Spotify provides several features to allow users to discover new music such as *Browse, Activity, Discovery, Radio* and *Spotify Apps*. These features commonly present the music recommendations as a list and/or grid, however by representing the information in this way, users cannot easily understand the reasons behind the recommendations. Spotify Apps offer different possibilities for users to discover new music because they rely on third-party applications. For example, Last.fm [6] developed an application using their own recommendation system. However, these recommendations are displayed in a manner which is not informative for users, by again using a grid and lists. The goal of creating a Spotify App for RAMA was thus to provide users with a visual representation that is more natural and intuitive than lists towards an enhanced experience of browsing music recommendations and artist relations.

2. RAMA IN SPOTIFY

RAMA uses a tree graph to display a network of music artists that share similarities. While there are other visual-based recommendation applications (e.g., `http://liveplasma.com` and `http://musicroamer.com`). A downside of the applications is the use of Youtube [7] to stream the music tracks. This is not ideal given the often poor sound quality of Youttube audio. Also, Youtube is not a music oriented service thus music-related queries do not always return expected content. For example, interviews or podcasts might appear in the search results among the music videos.

In contrast, Spotify is a music oriented service that has a large music catalogue and a high quality music content. It also has a very user-friendly environment that is ideal for the integration with RAMA. Spotify has made several tools available to develop applications [8] integrating their services. Spotify Apps is the ideal choice to integrate RAMA into Spotify thus improving user experience in a more natural and intuitive way.

3. MAIN FEATURES

3.1 Visualization of the Artists' Map

RAMA's Spotify App automatically draws the map with the current playing artist as the main node, as seen in Figure 1. The graph-like structure is created by recursively

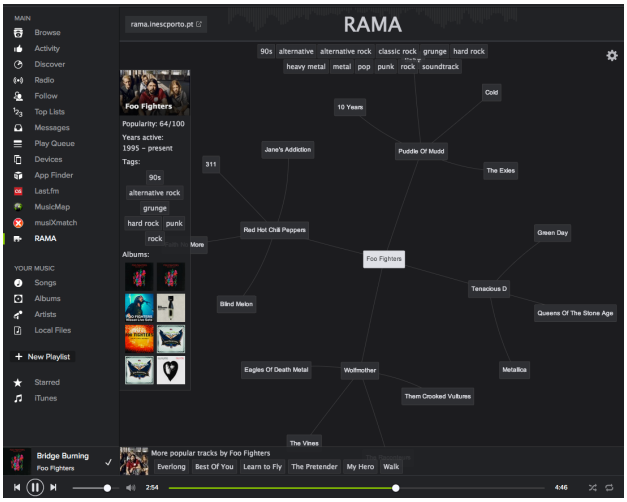

Figure 1. RAMA Spotify App opened inside the Spotify Desktop Client

[1] `https://developer.spotify.com/technologies/apps`
[2] `http://rama.inescporto.pt`
[3] `http://spotify.com`
[4] `http://rama.inescporto.pt/app`
[5] `http://developer.echonest.com`
[6] `http://last.fm`
[7] `http://youtube.com`
[8] `http://developer.spotify.com`

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.
© 2014 International Society for Music Information Retrieval.

fetching a list of related artists from each artist using the Spotify API [9] .

3.2 Visualization Parameters

The visualization parameters can be personalized in the settings menu in order to create richer graphs that better match users' preferences. The *depth* value of a graph determines how far the recursive algorithm goes when constructing the graph, such that the depth is the maximum distance between the root node and any other node in the graph. The *branching* value of a tree graph determines the maximum number of child nodes a node can have. The *tree mode* option indicates whether the graph to be built is a tree graph or not. This means that when the tree mode is on, the graph has fewer edges (which leaves the graph more clean) and so it becomes a tree graph.

3.3 Graph Edition

The available features to edit the graph are: *expand node*, *delete node* and *create a new map*. These interactions are available in the Artist Menu (Figure 2). The *expand node* action allows users to expand a node further (ignoring the graph's branching value). The *delete node* action allows users to delete a node from the graph. These two actions are useful for when users want to construct the graph to their needs. The *new map* action allows users create a completely new graph from a another node. This way the root node will be the selected node of the new graph.

3.4 Artist Menu

Users are able to see additional information about artists in the Artist Menu (Figure 2) such as their popularity value, albums and tags, as well as performing the expand and new map functions described in 3.3. When users select a node by clicking on it, the artist menu updates the displayed information. The popularity, the albums and the tags are metadata information retrieved from the Spotify API, although the Echonest API is also used as a fallback source for the tags as sometimes the Spotify API only returns a small number of results.

3.5 Tags Overlay

The tags overlay menu (Figure 3) can enhance users' understanding of the displayed artists' nodes in terms of associated tags. These tags are the same as those used in the Artist Menu (Figure 2). They are selectable, and so, when clicked, the respective artist nodes that are described by those tags, are highlighted (as seen in Figure 3). The tags shown in this menu, are just a small sample of the artists tags of the whole graph.

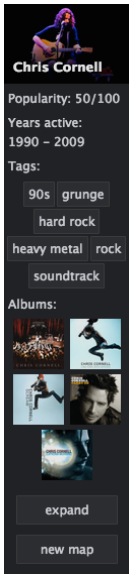

Figure 2. Artist Menu with information about Chris Cornell

Figure 3. Visualization of artists associated with a user-selected tag.

[9] https://developer.spotify.com/docs/apps/api/1.0/api-models-artist.html