ABSTRACT
In this paper we present RAMA (Relational Artist MAps), a simple yet efficient interface to navigate through networks of music artists. RAMA is built upon a dataset consisting about artist similarity and user-defined tags regarding 583,000 artists gathered from Last.fm website, and it provides two simultaneous layers of information: (i) a graph built from artist similarity data, and (ii) overlaid labels containing user-defined tags. Differing from existing artist network visualization tools, the proposed prototype emphasizes commonalities as well as main differences between artist categorizations derived from user-defined tags, hence providing enhanced browsing experiences to users.

Categories and Subject Descriptors
H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical user interfaces (GUI); H.5.5 [Information Interfaces and Presentation]: Sound and Music Computing—Systems; J.5 [Computer Applications]: Arts and Humanities—Music

General Terms
Design, Human Factors

Keywords
Information visualization, Search interfaces, Content ranking using social media, User interfaces for search interaction

1. INTRODUCTION
One of the fastest growing media on the web is web-radio. There are now many web-radios available where millions of users spend a very significant amount of their time. Users can customize a million-track collection to very specific music tastes. Web-radios usually allow users to type in a tag that describes the music they want to hear (e.g. “acid jazz”, “wake up”, etc.), and music items with that tag will make up the personalized radio feed recommended to that user. Artist similarities (computed from user tags, or any of the descriptive facets mentioned above) are also used to generate playlists. Albeit very useful, tag-based or similarity-based playlists are sometimes difficult for users to understand. The reason why a given artist or music was “selected” by the web radio is not always obvious and it can easily become confusing or frustrating for less experienced users that are unable to clearly express their queries. For example, very famous artists (e.g. “U2”) can sometimes be considered “similar” to other—otherwise quite different—popular artists (e.g. “Queen”, “Sting”, “Coldplay”, “Counting Crows”), just because they are also very popular or end up receiving the same relatively uninformative tags (e.g. “pop”).

In this paper, we explore the idea that networks of music artists contain rich and multifaceted information (music artist similarities, user tags, etc.) that can be useful for recommendations that go beyond the creation of playlists. We present RAMA, Relational Artist MAps, available through http://anonymised_for_reviewing_purposes. RAMA is a visualization tool that allows the user to navigate inside the network of artists of the Last.fm web-radio. Our tool uses information about artists similarity and artists tags provided by Last.fm to produce a visualization of artists relations and corresponding user-defined tags in a graph, hence fostering the visualisation of relational information. One of the original contributions of our work is to allow users not only to see which tags are common to a set of artists, but also those which are specific to a given artist when compared to similar ones. All this information is not directly visible in Last.fm’s web interface and hence our system allows users to obtain a clearer perception of both the commonalities and differences between artists, providing relevant directions for personalized browsing, and for gaining more insights regarding artists suggestions given by Last.fm recommendation algorithm. Navigating through the networks of similarities and tags is essential for understanding the media content and its relationships.

2. RELATED WORK
A great deal of work has been dedicated to the design of music recommendation systems. There are different approaches to music recommendation making use of the different facets of music. Recommending music items is essentially based on the definition of a similarity metric between items, possibly tuned for a particular user. As developed in greater details in [4] and [5], this can be achieved by: (i) automatic analysis of contents (i.e. via algorithms computing e.g. rhythmic and tonal descriptions of audio files), (ii) expert analysis (as is the case of Pandora, or Tapestry\(^1\)), (iii) collaborative filtering (i.e. exploiting a user-item relational matrix [7]), (iv) co-occurrence analysis (using e.g. crawling techniques to fetch from the web text related to music artists

\(^1\)http://www.amgtapestry.com/
and seek co-occurrences of artists names, terms, etc. [13], (v) analysis of meta-information provided by users (i.e. recommending music clips with a specific tag –e.g. “alternative rock”). Hybrid alternatives to these techniques also exist.

With regards to the means used for recommending music to users, this can be done for instance by the presentation of simple lists of potentially relevant music items (artist names, songs, etc.). This is the main artist recommendation means used e.g. on the Last.fm website.

Another way is to propose to users different visualizations of similarities between music items [12, 11]. For instance, a lot of attention has recently been given to the visualization of music artist networks. In artist networks, two different types of data are usually presented. On the one hand, there is encyclopedic data about individual artists. This includes biographic data such as names of albums, names of music tracks, photos and other images, etc. On the other hand, there is relational data, regarding e.g. artist similarities, connections between them, etc. A popular metaphor for visualizing these two types of data is that of connected graphs, where the data is presented through nodes and edges connecting them. Connected graphs offer a number of “containers” to represent information [14]: (i) Node Labels, (ii) Node Attributes, (iii) Edge Labels, (iv) Edge Attributes (v) Edge Directions (either directed or undirected links). Given specific types of artist-related data one wants to visualize, a specific mapping must be done onto these data “containers” to represent information [14]: (i) Node Labels, (ii) Nodes (or vertices) and edges are also central to the science of complex networks [1], and a number of works in this field have recently brought some light onto the manifold intertwinenes of musical artists networks [4, 8, 15]. A number of applications propose to visualize artist networks as two-dimensional connected graphs. For instance Musicovery2, TuneGlue’s “music map”3, Gnod’s “music map”4, Dimvision’s “music map”5, Kyle Scholz’s music recommendation tool6, or the “SimilarArtistGraph”7 by Last.fm user Shoxrocks. These applications make use of both individual and relational data.

The most common individual artist data is its name, presented as node label, sometimes together with a picture as well as an album or a track name. In some applications, nodes can be expanded to reveal attributes such as label names, dates of album release, biographies and link to artist websites.

But the real advantage of connected graphs over simple lists lies in the representation of relational information, such as artist similarity. In the above-mentioned applications, data regarding the similarity between artists is gathered from third parties such as Last.fm or Amazon. Artists (i.e. nodes) that are somehow similar (i.e. that Last.fm consider similar, or that Amazon recommend together) share a link (i.e. are connected via an edge). Links help users to easily identify interesting properties that would otherwise be very difficult to apprehend. More importantly, because links allow to visualize artist connections at a distance higher than 1 (further away than direct links), users can embrace in one sight artists that are similar to their query as well as those similar to the answers, and so on. A bird eye’s view upon the topology of artist graphs proposed in these applications may reveal clusters of artists, artists who are related to a large set of other artists (probably a sign of being influential), artists who connect several otherwise separated clusters (probably representative of a very specific and “hybrid” style of music), artists who are many links away of more popular artists (probably more obscure), etc. However, in the above-mentioned applications, similarities are binary, artists are either “completely” similar (all with the same degree of similarity), or not at all. Further, edges are not directed, and do not have labels nor attributes.

Similarly to these applications, our prototype uses third-party (i.e. Last.fm) similarity data to display music artists in 2D connected graphs, where edges represent similarities. In our prototype, node labels also convey similar data as they do. But an original aspect is that we intend to make further use of node attributes, as well as edge labels, edge attributes and directions, in order to convey more relational information (i.e. artist commonalities and specificities) computed from Last.fm data. Making use of more data, in our prototype, we sought a good balance between readability (avoiding cluttered use of space) and richness of the data presented to the user. Hence the special focus, in the design phase, on a proper use of graphical features (e.g. colors and transparencies) as well as interactivity between the user and the prototype (some information is shown by default, some other only as results of users’ interactions).

3. LAST.FM

Last.fm (http://www.last.fm) is an internet-based social music platform, where users can listen to music, find information about artists they like, or discover artists they might not know. Following Web 2.0 concepts, users can also set up their own profile, facilitating targeted automatic recommendations, among other things they can also get information about users with similar tastes, gis in their local area, videos, etc. Last.fm provides an interface for users to collaboratively edit encyclopedic information about artists. For some of the more popular artists there is extensive biographic information already available (see Figure 1 for an example).

User profiles –hence recommendations– are constantly updated via a software (free of use) which gathers (“scrobbles” in the Last.fm vernacular) statistics about the music listened to by users. User listening patterns are recorded and analyzed by Last.fm in order to better organize and recommend music.

Users are also encouraged to organize the music they listen to by assigning tags to artists, or even to specific albums or tracks. The definition of tags is up to the users and can describe any aspect users believe are relevant, as music genres (e.g. “rock”, “Viking metal”), locations (e.g. “Berlin”), mood (e.g. “chill”), opinions (e.g. “songs my mother would like”), contexts (e.g. “love”) or just about anything that cross users’ minds (see Figure 2 for examples of tags assigned to the band “Radiohead”). Tagging music helps users to browse Last.fm contents. But the real power of tags becomes clear when considering that tags of hundreds of thousands of users are combined, providing an emerging “bottom-up” categorization of music.

A cornerstone of Last.fm functionalities resides in links

of similarity between artists (which can be seen on the left column of Figure 1 and on Figure 3, and which is central to automatic recommendations made to users). The algorithm used for computing similarities between artists is unknown to the authors of this paper but is probably based on (i) the analysis of user-added tags, (ii) user listening patterns such as co-occurrence statistics (users that listen to artist X also listen artist Y), and (iii) user profiles information (“similar” users should like “similar” artists).

### 3.1 Some Statistics about Last.fm Data

For the purpose of better understanding the richness of this Web 2.0 medium we will present some descriptive statistics regarding user-defined tags, derived from the data that we obtained from Last.fm site in February 2008 (details about how this data was obtained are explained in Section 4.2).

By inspecting the tags given to an universe of about 583,000 artists we found 2,774,068 tag attributions, averaging 4.76 tags per artist. There are 208,565 distinct tags, which is a surprisingly high number. Table 1 shows the top 20 most common tags and the number of artists that were tagged with it. These very frequently used tags describe common music genres. Interestingly, the most used tag is “seen live”, which reflects a previous experience of the listener with that artist. We found that tag usage follows a typical Zipfian-like distribution. For example, 126,970 tags were used only once while 25,114 were used twice. Figure 4 shows tag usage distribution, ranging from tags used only once (125,970 tags), to a few tags being used tens of thousand of times (e.g., the ones shown in Table 1).

On low frequency part of the tag spectrum there are tags that express personal views about artists and very specific music genres, and tags that results from simple spelling mistakes. For example, we found tags such as “911 blues”, “a bit childish but still cute”, “a bit like enter shikari but without the whole emo aspect”, “absolutely kick ass”, “game musics”, “maria rock”, “married to a genius”, “marx-
Table 1: Top 20 most common tags

<table>
<thead>
<tr>
<th>Tag</th>
<th># Artists Tagged</th>
</tr>
</thead>
<tbody>
<tr>
<td>seen live</td>
<td>54,660</td>
</tr>
<tr>
<td>rock</td>
<td>41,854</td>
</tr>
<tr>
<td>electronic</td>
<td>33,108</td>
</tr>
<tr>
<td>indie</td>
<td>27,913</td>
</tr>
<tr>
<td>alternative</td>
<td>25,401</td>
</tr>
<tr>
<td>pop</td>
<td>24,010</td>
</tr>
<tr>
<td>punk</td>
<td>20,555</td>
</tr>
<tr>
<td>electronica</td>
<td>18,781</td>
</tr>
<tr>
<td>metal</td>
<td>17,419</td>
</tr>
<tr>
<td>experimental</td>
<td>16,680</td>
</tr>
<tr>
<td>jazz</td>
<td>15,927</td>
</tr>
<tr>
<td>folk</td>
<td>15,843</td>
</tr>
<tr>
<td>Hip-Hop</td>
<td>15,808</td>
</tr>
<tr>
<td>hardcore</td>
<td>14,572</td>
</tr>
<tr>
<td>female vocalists</td>
<td>14,906</td>
</tr>
<tr>
<td>ambient</td>
<td>14,807</td>
</tr>
<tr>
<td>dance</td>
<td>13,508</td>
</tr>
<tr>
<td>indie rock</td>
<td>13,064</td>
</tr>
<tr>
<td>rap</td>
<td>12,625</td>
</tr>
<tr>
<td>japanese</td>
<td>11,477</td>
</tr>
</tbody>
</table>

We also found tags that are actually very long user statements instead of short specific labels, such as:

- “a wonderful artist who makes loads of wonderful pieces of wonderful art and who really deserves such a wonderfully long wonderful tag which may wonderfully piss off everybody who has so wonderfully much time to read this wonderfully long wonderful tag”
- “ive seen these folks live and the show was really terrific i will cherish the memories always and i may have danced frenetically at these shows or been really drink or too sober but regardless these folks put on an amazing show”.

4. RAMA: RELATIONAL ARTIST MAPS

4.1 System Overview

RAMA is built on top of a client-server architecture. The visualization is performed on the client side (the user application) using information obtained from the server via an HTTP request. The server manages the data that has been extracted from Last.fm site and performs all the necessary pre-processing operations to provide the client with the information needed for visualization.

Given an initial query (i.e. an artist name) submitted by the user through a text-box in the client application, a request is sent to the server which provides all the information that the client application requires to draw the corresponding artist network. The network contains the artists that are found to be more similar to the queried artists, according to Last.fm’s artist similarity index. It also contains the artists that are found by propagating the similarity query to the previously found ones so that we obtain artists several levels away from the initial artist. This allows the user to see the artist position in wider context (see e.g. Figure 7).

The server provides the client with a list containing information regarding each node in the graph (i.e. each artist in the network), namely:

- the 2D coordinates of the corresponding node in the graph, computed by our graph layering algorithm (see Section 4.3);
- the list of user-defined tags for each artist as taken from Last.fm site;
- explicit similarity relationships with other artists in the network (for drawing the necessary edges) as taken from Last.fm site.

Once the visualization is rendered by the client application, users can either insert another query in the text-box or, more interestingly, they can browse through the network, interactively consult tag information, and perform additional queries by clicking on the nodes. Interaction continues in this fashion until the user quits the application.
4.2 Data Pre-processing

Our system uses data that is freely available through Last.fm web-services API\(^8\). Through this web-service developers can access different data categories related to Last.fm users and media. Available data includes: (i) the profile of users of Last.fm, (ii) information and statistics related to artists, their albums and corresponding music tracks, (iii) tag information for each of the previous items, (iv) information about user created groups (e.g. fan groups), (v) information about message forums that users can create/participate, and (vi) geo-aware statistics about users and music preferences (more details can be found in the web service site).

From all the data available, our system uses currently only data specially concerning artists, namely:

- artist data and basic statistics: name, URL of image, “popularity” index within Last.fm community,
- the list of the most similar artists for each artist. Last.fm assigns each similar artist a weight ranging from 100 (full similarity) to 0 (almost no similarity).
- information about the user-defined tags for each artist. Again, Last.fm assigns weights to quantify the association level for that tag: 100 means full association, while 0 indicates loose association.

Because each individual access to the Last.fm web service involves a considerable overhead due to network latency and server load, we created a local copy of the data we needed by crawling the web-service systematically. For a period of about a week (from January 30 to February 4 2008) we consulted the Last.fm’s web-service and obtained the previously described data for a total number of 583,000 artists. Local access to this data allowed to speed-up experimentation and to improve the global performance of our system.

Although the Last.fm user community is constantly contributing and changing this information, we can assume that the relationships between artists, which are the focus of our work, are more or less stable, at least for the reasonably popular artists. In any case, the crawling procedure can be repeated at any time to update data.

4.3 Layering the Graph

For layering the graph corresponding network of artists in a 2D plane, we implemented our own graph drawing system based on a force-directed placement strategy [6]. Our implementation, in Perl, is completely integrated in the overall server-side framework which was also implemented in Perl.

There are several parameters related to the graph layering operation that need to be set in order to create an appealing and useful visualization. The first parameter concerns the number of nodes (i.e. artists) that should be included in the graph. We wish to convey as much information as possible but we are limited to a given frame size. Layering graphs with many labeled nodes – which may have other attributes to be drawn – in a regular computer screen may generate rather confusing visualizations. Therefore we need to limit the number of nodes to a reasonably low value, for example between 15-50 nodes.

The second parameter has to do with the size of the context we wish to convey. Since we are limiting the number of nodes to a given maximum, there is a commitment between

\(^8\)http://www.audioscrobbler.net

Figure 6: Two choices of 6 related artists for laying out a network with different context perspectives.

Option 1 shows a network around around artist A consisting of the top six directly related artists (a1 ... a6). Option 2 display the same number of nodes but shows not only nodes directly related to the artist, a1 and a2, as well as those that are directly related to these but indirectly to artist A (a11, a12, a21 and a22). The number of nodes to be displayed is the same but Option 1 promotes local detail while Option 2 sacrifices local detail to widen the perspective of the visualization.

Our goal is to combine a good level of local detail with a wide enough perspective, and achieve a good balance between readability and richness of data presented to the user. We have thus defined 3 parameters to configure the properties of the network to be drawn around a given artist:

- level-1 branching factor: this parameter controls how many of directly connected nodes are to be visualized. The nodes chosen are the top similar ones. For example, in Figure 6, Option 1 has a level-1 branching factor of 6 while Option 2 has a factor of 2.
- level-n branching factor: this parameter controls how every node in the graph except the initial one will branch. For example in Figure 6 Option 2 as a level-n branching factor of 2, because nodes a1 and a2 are branched in two other nodes.
- maximum branching distance: this parameters imposes a threshold on the maximum branching distance, i.e. how many links away can nodes be from the initial artist node. Again in Figure 6, Option 1 has a maximum branching distance of 1 while in Option 2 that parameter is 2.

With these parameters we perform an iterative expansion of the initial artist node, branching each node according to the corresponding factor. Increasing level-1 branching will promote local detail, while increasing level-n-branching and maximum branching distance will widen the context. These last two parameters have to be configured with some care because they involve an explosive growth in the number of nodes. In any case the number of the nodes can never be more than a global maximum of nodes, so expansion stops when this pre-defined threshold is reached.
4.4 User Interface

The user interface was developed in Processing\(^9\) and is responsible for generating the visualization and providing interactivity.

Initially, the user can enter a name of an artist in a textbox. A query is then sent via HTTP to the server, which replies with the artist network data. For each artist (i.e. each node) in the network, the server sends back to the client the following data: (i) the name, (ii) a popularity index, (iii) the url artist photo in Last.fm server, (iv) the top 20 user-defined tags, (v) information about similarity with other artists (i.e. the node edges and their weights), and (vi) the coordinates for placing the corresponding node of the graph in a 2D plane. All this information is sent to the visualization interface in text format to allow a simple parsing procedure.

The interface application uses additional data that is directly fetched from Last.fm site at runtime, namely artist pictures and a short biographic description. For these media items performance constraints are not so severe (only two extra accesses for each artist in the network), so the interface can access that information directly as needed without requiring any intervention from the server.

On top of the basic network structure we place information related to user-defined tags. However, instead of presenting all the tags assigned to each artist over its corresponding node, we try and place the most common tags in the network in such a position that they will simultaneously describe all artists contained in a certain part of the network (simple animations of these tags permit to avoid readability problems). Such tag information is shown by default, no action being required from the user, allowing the user to easily identify the attributes that explain why certain artists are clustered in a specific part of the network.

For instance, in the network of “Radiohead” (see Figure 7), one can identify a region of artists tagged as “alternative” in the center because such a tag is common to most artists in the whole network. On the other hand, the region on the right branch is tagged “electronica”, the common attribute of artists in that part of the network. Tag sizes are proportional to the number of artists for which they are relevant.

Moving the mouse cursor on top of a specific artist node results in the presentation of diverse data: the artist picture, a short description and bio (gathered at run-time from the Last.fm site), and a link to its Last.fm webpage. Left-clicking on any artist name has the same effect as entering its name in the query box, i.e. sending a new query and refreshing the map with this query as seed. This allows simple user navigation through the artist network. To ensure that visualization always exhibits some degree of novelty, a new graph layout is computed in real time for each query, even if the query has been processed before.

(MAYBE REMOVE SUBSECTION TITLE)

4.4.1 Visualizing Specific Artist Tags

An original feature of our user interface resides in the possibility to visualize tags that are specific to an artist in relation to another, i.e. those tags that are relevant to a artist but not to its neighbors. This information is shown only when the user crosses the mouse cursor over an edge.

\(^9\)http://www.processing.org

This action will result in the rendering of the tags that are specific to each of the two artists at their respective sides of the edge.

In Figure 8 one can see an example of such a behavior. “Bjork” and “Massive Attack” are both part of the “Radiohead” network. Although they are both on a region characterized by the “electronic” tag, the tags “icelandic”, “pop”, “singer-songwriter” and “avant-garde” are only assigned to “Bjork”, while the tags “trip-hop”, “british”, “dub” and “chill” are assigned only to “Massive Attack”. This provides a very clear description of the unique features of each artist in the network.

We believe this feature is a very powerful tool in increasing user understanding about the network data. Table 3 shows

Figure 7: Example graph for artist “Radiohead”.

Figure 8: Examples of specific tags for artists “Bjork” and “Massive Attack”, both part of the “Radiohead” network.
a few more examples of tags that are found to be specific to a given artist (“Radiohead”) when compared to other artists in its network.

<table>
<thead>
<tr>
<th>Artist</th>
<th>Sample Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radiohead</td>
<td>Sigur Ros</td>
</tr>
<tr>
<td>UK, pop, britpop, art rock</td>
<td>shoegaze, chillout, icelandic, ethereal</td>
</tr>
<tr>
<td>Thom Yorke</td>
<td>Muse</td>
</tr>
<tr>
<td>progressive rock, 90s, post-rock, experimental rock</td>
<td>radiohead, triphop, electro, chillout, singer-songwriter, male vocalists</td>
</tr>
<tr>
<td>Placebo</td>
<td>Muse</td>
</tr>
<tr>
<td>emo, glam rock, metal, punk</td>
<td></td>
</tr>
<tr>
<td>Beck</td>
<td>Coldplay</td>
</tr>
<tr>
<td>progressive, emo, metal</td>
<td></td>
</tr>
<tr>
<td>british, art rock, progressive rock, UK</td>
<td>lo-fi, american, indie pop, folk, singer-songwriter, hip-hop</td>
</tr>
<tr>
<td>experimental, 90s, electronica, post-rock</td>
<td>emo, chillout, mellow, pi ano rock, punk</td>
</tr>
</tbody>
</table>

Table 2: Pairwise discriminative tags between “Radiohead” and other similar artists.

5. FUTURE WORK

Future work includes enhancing user experience by adding song snippets for each artist, so that the user can play them on demand while navigating across the network. Also, we plan to improve interactivity by allowing the user to optionally navigate through user-defined tags, and not just artists. We will also focus on allowing the user to manipulate the graph (zooming, rotating, etc) and to edit it. Editing capabilities will enable the user to remove nodes (artists) from the graph, expand only some, and thus generate a personalized graph, which could then be saved e.g. in the form of a playlist.

Future work will also include dealing with problems that we found with Last.fm’s data. One of the problems has to do with the multiple variations of names. It is difficult to enforce standards and absolute correctness on community-edited data so we found many cases where the correct artist names and several misspelled or non-standard variations coexist. For example, we found “Kaiser Chiefs”, “Kaisers Chiefs”, “Kaiser Chiefs”, “Keiser Chiefs”, and “Kaiser Chiefs”. There are more complex variations for cases involving collaboration between artists such as for example “Dave Matthews & Rolling Stones”, “Dave Matthews Band & Rolling Stones” and “Dave Matthews Band ft. The Rolling Stones”. There are also similar issues with user-defined tags. For example, the tags “post rock” ”postrock”, “postrock” and “postrock” are probably equivalent. Another problem regarding naming that we found is that Last.fm information system assumes unique names for artists. If two artists have the same names there is no additional key that allows us to differentiate the corresponding artists. Thus, tag and similarity information of different artists get merged under a single polysemic name. Although this is not a problem for very popular artists, whose names are usually very carefully chosen, it occurs not so rarely for other less popular artists for which meta-information may become inconsistent. All these issues are instances of a problem known as named entity resolution (NER). Similar ambiguity issues occur for example in social networks [10], bibliographic databases [2] and information extraction from text [9]. We plan to address this in future work.

6. CONCLUSIONS

RAMA provides a simple yet efficient interface to navigate through the network of similar artists, allowing users to obtain a wider view about the artists they know, and to easily discover new bands and artists that they might like. It provides two simultaneous layers of information: (i) a graph built from artists and their connections, and (ii) overlaid labels containing user-defined tags that express the classification made by Last.fm community for each of the artists. From experimentation we have observed that the system effectively allows to identify clusters of tightly connected bands and artists (such as for example former members of a band that pursued a solo career). Additionally, the visualization procedure in RAMA also emphasizes the main differences between artists, allowing the user also to visualize which are artists most distinctive attributes.

7. ACKNOWLEDGMENTS

This work was partially supported by grant SFRH/BD/23590/2005 from FCT (Portugal), co-financed by POSI.

8. ADDITIONAL AUTHORS

Additional authors: Eugénio Oliveira (LIACC / FEUP, Univ. do Porto, email: eco@fe.up.pt).

9. REFERENCES


