Map Visualization of Artists' Tour Stops and Microblog Data

Martin Hellwagner, 0956048
Johannes Kepler University, Linz

Abstract. In times of decreasing quantities of music sales, touring is becoming a more important aspect for artists to consider. Thus, keeping an eye on tour schedules becomes increasingly difficult for music fans as the number of concerts is growing continuously. Event directories such as `Last.fm` and `Songkick` offer text-based listings of tour stops around the world in order to facilitate their discovery by potential audiences. The paper at hand extends these listings in favor of a novel event visualization approach using `Google Maps`. The rationale is that fans are rather interested in being able to locate shows in spatial vicinity by consulting a map than scanning through extensive lists of events. Furthermore, the presented application is augmented with microblog data, effectively displaying `Twitter` messages in order to discover the locations of artists' dedicated fan bases and thereby assisting musicians and their management in the planning of tours. First, the application is described as well as its functionality demonstrated. Subsequent evaluation of the results reveals good accuracy of the tool in comparison to a ground truth acquired through the respective artists' websites.

Table of Contents

1 Introduction

In times of decreasing quantities of music sales (especially on physical discs), touring is becoming a more important aspect for artists to consider. Either as a regular show or as part of a summer festival, live concerts reach large audiences while generating veritable profits for well-known musicians. Although exhausting and stressful at times, it is more often than not in the artists' own interest to cover large parts of the world with different "legs" of a tour supporting their latest studio effort.

For instance, one of the most successful tours so far was undertaken by the English rock band The Rolling Stones following the release of their album A Bigger Bang [5]. Comprising 144 shows in front of nearly 4.7 million people around the globe between the years of 2005 and 2007, it grossed more than 550 million US dollars at the time. Additionally accounting for inflation, the 2013 equivalent of this sum would be 618 million US dollars, a figure that even rivals some smaller countries' GDPs [6].

However, while concerts around the world are becoming more numerous, music fans and potential audiences are not always willing and able to keep an eye on tour schedules in their own country and - depending on their dedication to an artist - also abroad. The paper at hand therefore presents a tool that extends the text-based listings of event directories such as `Last.fm` [3] and `Songkick` [4] in favor of a novel visualization approach using `Google Maps` [1]. The rationale is that fans are rather interested in being able to locate shows in spatial vicinity by consulting a map than scanning through extensive lists of events in the hope of finding a tour stop nearby. Besides, many ticket sales companies (e.g. Eventim or Ticketmaster) solely feature events in one particular country, ignoring those that take place elsewhere. The proposed tool avoids these restrictions by displaying all the shows of an artist, giving fans a useful tool to support them in deciding which concerts they are going to attend.

While the geo-localization of events is particularly aimed at potential audiences, the augmentation of the presented tool with microblog data is especially interesting for artists undergoing tours. One of the most popular microblogs is `Twitter` [2], currently home to an online community of over 500 million registered users [7]. At the core of `Twitter` are messages limited to 140 characters, which are colloquially known as tweets. With these messages, users can employ keywords labeled with hash tags in order to contribute to trending topics as well as communicate with other users by mentioning their user names. Mining the quantity of mentions regarding various artists, this paper aims at discovering the locations of their dedicated fan bases, ultimately assisting musicians and their management in the planning of tours. It is assumed that by taking into account public demand, tours can be more accurately scheduled. As a result, the fraction of sold tickets can be increased and the overall revenue maximized.

The remainder of the paper is organized as follows: Section 2 presents related work on event data mining, microblog data analysis and maps visualization. In Section 3 used

technologies are presented and necessary terms are described. Section 4 explains and demonstrates the functionality of the visualization tool, and Section 5 identifies problems of the implementation. Finally, in Section 6 the data is exemplarily evaluated against a ground truth obtained from human inspection of artists' websites, and Section 7 concludes the paper while discussing future work.

2 Related Work

As far as the author is aware of, no scientific work has been published on the specific topic of the paper at hand. However, a fair amount of research has been conducted in the fields of event data deployment, microblog data analysis and maps visualization, respectively.

2.1 Event Data Deployment

Troncy et al. [21] analyze ways of linking events with media items in order to present users with a familiar, yet enriched event browsing experience. The authors incorporate background information, pictures, videos as well as user opinions and comments into the text-based event descriptions provided by services such as `Last.fm`, eventually creating an application that displays events using three different perspectives (media-centric, location-centric and chronological). For data standardization purposes, the authors additionally employ two ontologies, both providing minimal models that encapsulate the most useful properties for the description of event data and associated media respectively. While the `LODE Ontology` describes an event's kind, place, time and its participants, the `W3C Ontology for Media Resources` specifies properties such as various renditions of a photo or duration of a video, its target audience, genre and rating.

Liu et al. [22] extend this approach by arguing that events can automatically be detected and identified through the analysis of media items shared on social networks. Indeed they are able not only to detect events with high accuracy but also to identify events that have not been published in event directories. Also employing the ontologies presented in [21], the authors first collect plenty of events for a number of selected venues around the world. Then, they model the venues' locations using a bounding box approach which relies on GPS coordinates of photos taken around the venues and shared on social networks. Finally, they select the dates with peaks in the quantity of pictures and consider them as candidate event dates. Their approach performs well in comparison to a ground truth, emphasizing its potential.

2.2 Microblog Data Analysis

Given the fact that in social networks and microblogs, virtually every user is able to create any kind of information without even having to specify their real name, verification of the

posted data is crucial for scientific research. Work dealing with information credibility on `Twitter` is presented by Castillo et al. [24]. In their paper, the authors present a classifier which first partitions random tweets into chat and news categories and subsequently assesses the trustworthiness of information in the latter. Features such as the length of the tweets, the quantity of emoticons, the number of statuses by a particular user and the structure of the tweet propagation tree are used to train the partitioning classifier. Comparing against human assessment of the same set of tweets, the methods presented in [24] show reasonable performance, giving the authors confidence in the practical applicability of their work.

Cha et al. [25], on the other hand, aim to determine the influence certain `Twitter` users exert in the network. The authors define influence by means of three measures: indegree (i.e. number of followers of a user), retweets (i.e. quantity of a user's tweets being forwarded) and mentions (i.e. number of messages referring to a user). The analysis of the top influences in each of these categories reveals that influential users are not only able to retain the effect across various tweet topic genres, but also over time. Moreover, the authors show that average users can suddenly gain a lot of influence by being active in the advent of a certain topic.

Dealing with microblog data analysis in a slightly different way is the work presented by Hiruta et al. [27]. The authors' aim is to deduce real-world events by considering tweets containing both geo-tags (e.g. GPS coordinates) as well as content relating to them. In order to classify the collected information, the authors introduce five significant categories for the tweets to be partitioned into. They subsequently implement their approach using both an animation visualizer and a web-based interactive interface. While the former allows to quickly identify patterns of tweet propagation (e.g. following an earthquake), the latter allows browsing tweets grouped by topic, area and time. Again, human evaluation of the tweet categories confirms the potential of the proposed classification algorithm.

Conversely, Cheng et al. [28] propose a method of deriving the location of `Twitter` users based solely on the content of their tweets. The authors argue that web users employ different unique keywords in their messages depending on where they are located, identifying the city with the highest quantity of these keywords as a possible candidate for the users' position. Indeed they are able to show that by employing the appropriate models and filtering methods, more than half of the set of mined `Twitter` users can be placed within 100 miles (160 kilometers) of their actual location. Although their experiments are conducted only within the continental US, the proposed approach yields results promising enough to be expanded further. Because of the general sparsity of geo-spatial information in microblog data (cf. Section 5.2 for more detail), their work is particularly interesting for any application dealing with location data in tweets.

2.3 Map Visualization

On the intersection of microblog data analysis and maps visualization is work presented by Hauger and Schedl [26]. The authors develop a visualization framework for interactive browsing and dynamic exploration of tweeted music listening events, thus giving users an overview of where in the world `Twitter` users listen to which kind of music. A high quantity of tweets has been acquired prior to development that contain both a music-related hash tag (e.g. `#nowplaying` or `#itunes`) and geo-spatial information. After partitioning different types of music into a number of clusters, coordinates of the tweets are attached to a `Google Maps` representation of the world using different colors. Similar to the tool presented in this paper, interactions with the visualized tweets are possible by hovering with the mouse over a marker in order to open an information window. Short audio snippets of the music items referred to by the tweets can be played if available.

In addition to the visual presentation of the tweets, the authors implement a set of analysis tools with which the users can query play counts of certain artists and apply geographic and temporal restrictions if desired. Furthermore, genre similarity between two artists can be computed based on the number of occurrences occ_i and occ_j of each of the artists' names in a tweet as well as the number of their co-occurrences $cooc_{i,j}$ (e.g. the number of `Twitter` users writing about artist i as well as about artist j). The authors introduce the following formula in order to calculate similarity:

$$sim(i,j) = \frac{cooc_{i,j}}{\sqrt{occ_i * occ_j}} \tag{1}$$

Albeit used for different purposes, the same notion of map visualization makes the tool presented in [25] an important resource for the implementation of the microblog data visualization component of the work at hand.

Bassoli and Brewer [23] present in their paper a tool that is at the same time at the core of their company. A service for web and phone, `frestyl` addresses the problem of promoting and discovering live music and is especially aimed at emerging musicians, local promoters as well as small and medium venue owners. Displaying a map centered on the user's location (as determined through geo-localization), all the events happening in spatial and temporal vicinity to the user are shown as black markers. Not unlike the work at hand and the tool presented in the next subsection, a click on a specific marker reveals more detailed information about the associated show. The authors state that due to the design of `frestyl`'s interface, rich and complex local music scenes of bigger cities can be represented while also being accessible globally. Moreover, they argue that the focus of their tool lies more on giving users an overview of where and when to attend small non-mainstream shows and less on discovering large concerts by popular artists that are already well-advertised.

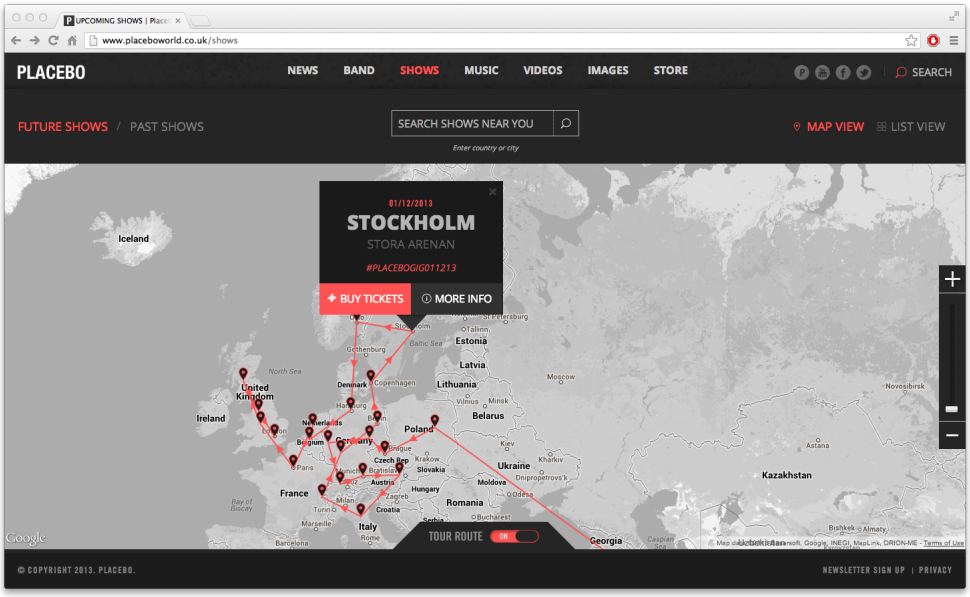


Fig. 1. Map view of Placebo's 2013 European tour, with focus on the Stockholm show.

2.4 Additional Related Work

Not a scientific publication per se, but nevertheless notable is the newly designed website of the English rock band Placebo, specifically the section "Shows" [8]. Of particular interest is the interactive map featuring all the band's upcoming shows as well as concerts given in the past. Figure 1 depicts the upcoming 2013 fall/winter European tour of the band with focus on the Stockholm tour stop. When hovering over a red marker with the mouse, the city name and date of the corresponding tour stop are displayed. Clicking on a specific date reveals more detailed information about the selected show such as the name of the venue, the `Twitter` hash tag and a link to the responsible ticket sales company. Note that the red arrows between the markers illustrate the tour route, and can be deactivated using the corresponding switch. As displayed in the bottom corners of the map, respective data is supplied by `Google Maps`, much like in the tool presented in this paper. Information about the shows is provided by the band itself, a fact that becomes more apparent when choosing the "List view" option in the upper right corner of the website.

While the approach is akin to the one presented in this paper, the announcement of the website's update [9] has been published about a month after development of the author's work had started. Furthermore, the tool at hand rather aims at providing users with a possibility to visualize any band's tour without being developed for or limited to any specific artist. Data about events is retrieved using the public interfaces of event

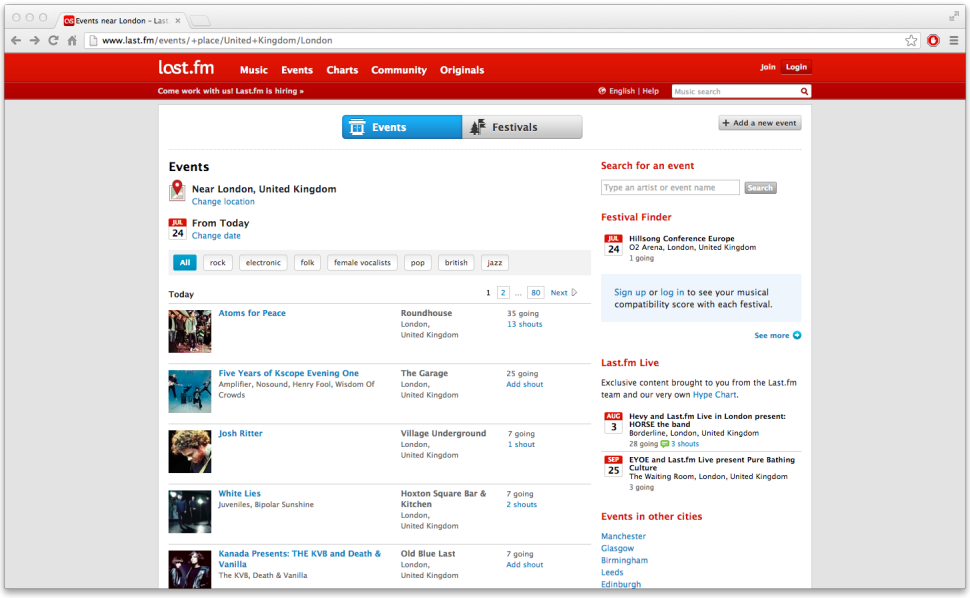


Fig. 2. Website of the `Last.fm` [2] event directory while browsing concerts in and around London.

directories such as `Last.fm` and `Songkick`. These technologies are subsequently presented in the next section and necessary terms are described.

3 Methodology

This section presents used technologies and describes necessary terms. Similarly to the previous section, it is distinguished between the acquisition of event and microblog data as well as the development of the backend and frontend of the web application. The subsequently presented technologies are, however, not considered related work but rather means with which the implementation was realized. Most of them are well-established de facto standards in the industry, thus enabling adaptability and expandability of the tool presented by the paper at hand.

3.1 Event Data Acquisition

The first notion upon having been presented with the problem of event data visualization was identifying suitable sources for this type of information. Mining all artists' separate websites would have been too complex a challenge to accomplish since they are more often than not too diversely built and seldom offer a proper interface. Even though the ground truth is embedded in these websites (cf. Section 6 for an evaluation thereof), a

different approach was necessary to tackle the problem of even data acquisition. Luckily, event directories cumulating all the concerts for a certain artist and location have been becoming more popular in the last few years, featuring an extensive list-based view of small and large concerts virtually everywhere in the world. Therefore, the author chose two well-known event directories to be at the core of the presented application, namely `Last.fm` and `Songkick`. While the latter solely features a list of events comprised of the intersection of users' favorite artists and cities (thus creating a personalized event list), the former additionally offers a music discovery component as well as a large database containing artist information, pictures and videos. Filtering possibilities include, amongst others, spatial, temporal and genre-based settings. Figure 2 shows the homepage of the "Events" tab of the `Last.fm` website while browsing concerts happening in and around London.

`Last.fm` and `Songkick` both provide an API (application programming interface) which can be used to query detailed information for a certain artist as well as their upcoming and past events. The only premise for using these services is the registration as a developer, which was done prior to development of the presented tool. Access to the API is granted upon specifying a unique identification number assigned to every developer. Along with this API key, the query string needs to include information about which artist is selected. Although it is possible to simply specify the artist's name, the author decided against doing so because of the ambiguity implied by this approach. For instance, a user who searches for concerts by the "Red Hot Chili Pepers" might end up with zero results because of the incorrect spelling of the word "pepper".

Instead, the approach favored in this paper makes use of a 36 character identifier issued by the `MusicBrainz` [10] database. Publicly available, a unique MBID is assigned to every artist, release, recording and so forth. Conveniently enough, `Last.fm` offers a service which cross-references an artist's name with the `MusicBrainz` [10] database and returns the correctly spelled name as well as its unique MBID and various other parameters (e.g. the artist's music genre), ruling out potential ambiguity. This service, which is called $artist.getInfo$ in the `Last.fm` API, is employed by the work at hand prior to the acquisition of event data. Both the request and the reply are processed using JSON (JavaScript Object Notation) and AJAX (Asynchronous JavaScript and XML). These technologies are explained in Section 3.3 in more detail.

After having mined an artist's information using the aforementioned service offered by the `Last.fm` API, their upcoming and past events are fetched employing the $artist.getEvents$ and $artist.getPastEvents$ services respectively. Acquisition of events with the `Songkick` API is carried out analogously by employing the corresponding *artists.calendar* and *artists.gigography* services. Note that while the `Last.fm` API is used for both the acquisition of an artist's information and their events, the `Songkick` API is used solely for the latter. The reason for this is that a higher quantity of events can be fetched when

mining data from two independent sources, whereas artist information has to be mined only once.

While the APIs of both event directories provide the developer with plenty of information about a single event, only the following parameters are used in the web application: date of the event, name and - if available - coordinates of the venue as well as names of the city and country. Non-existence of events (e.g. because of an artist not being on tour) is marked accordingly. In order to avoid unacceptably slow reaction time of the application, mined events are constrained to those lying ahead and back a maximum of 365 days respectively.

3.2 Microblog Data Acquisition

Having acquired the necessary event data, the next logical step was to mine microblog data. As mentioned in Section 1, one of the most popular microblogging services is `Twitter`. It therefore stood to reason to use this service in order to fetch the required data. Not unlike the event directories mentioned before, the service provides multiple extensive APIs for operations on tweets, users, entities (e.g. hash tags and media items) and places. An initial attempt of mining `Twitter` data was following the approach taken by Hauger and Schedl [26]. In their work, the authors used the `Twitter` Streaming API to collect large amounts of tweets over a period of nearly a year. This approach was, however, deemed inappropriate for the work at hand. For once, the tool presented in this paper is intended to be developed in a much shorter time frame than the application introduced in [26]. Moreover, upon closer inspection the `Twitter` Streaming API didn't prove a valuable option to the author due to the different nature of the tool at hand. This is because its focus lies less on analyzing pre-collected information, but more on dynamic just-in-time data collection. Therefore, the author eventually chose to employ a different approach involving the `Twitter` Search API.

While accessing the APIs of `Last.fm` and `Songkick` proved to be a fairly straightforward endeavour, the `Twitter` Search API required a more complex authentification procedure. Before being able to acquire any data, an application has to authenticate itself using the OAuth standard which requires information such as a consumer key, a consumer secret, an access token as well as an access token secret. When access to the API is granted, the developer can search for any of the publicly available tweets by specifying a set of parameters. While many of them are optional, the search query q is mandatory at all times. The application at hand employs as search query the term @*artist*, with "artist" being a place holder for the desired artist's name and the "@" sign denoting the process in `Twitter` of mentioning a user name (i.e. communicating with the respective user). Other parameters used in the application query include *count*, which specifies the maximum number of tweets to be fetched, and *result_type*, which describes the type of results returned by the API (e.g. popular, recent, mixed).

Again, not all the information about a single tweet is used in the web application. Instead, only the following parameters are needed: date of the tweet as well as location and timezone of the user creating the tweet. Unfortuantely, very little information is available through the `Twitter` Search API about the exact coordinates of a tweet, hence this kind of information is of no use when trying to assign tweet locations (cf. Section 5.2 for more detail on this problem).

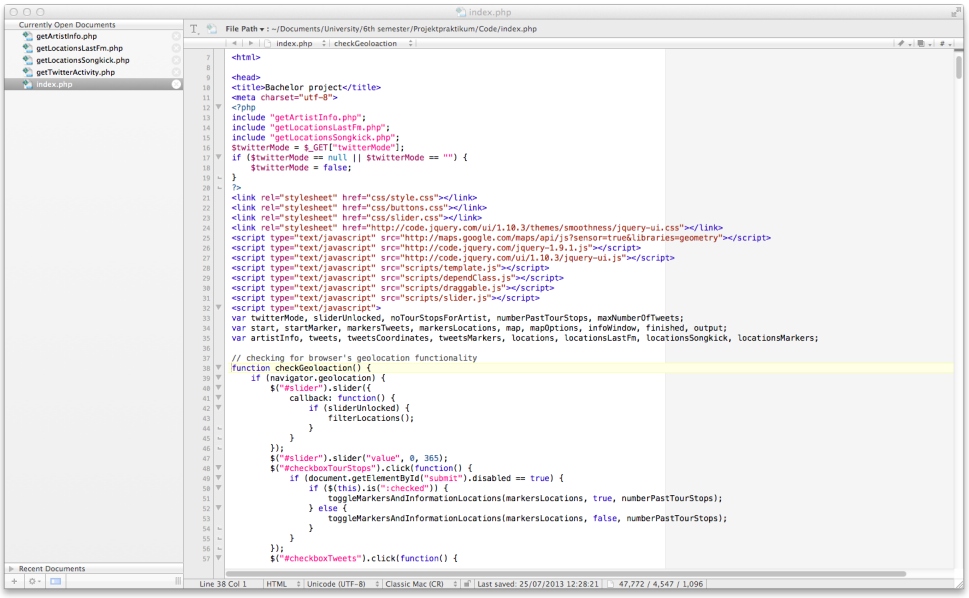
In order to make sure that only tweets mentioning actual artists are mined, the selected artist account is checked to be verified prior to the acquisition of tweets. Verification is a service offered by `Twitter` to make sure that accounts belonging to popular personalities such as celebrities, musicians and politicians are truly in their possession and not "hijacked" by fans or other users trying to impersonate these people. Thus, the search query with which the tweet data is mined will only be issued if the recipient account is verified. Therefore, accuracy of the popularity estimation approach can be guaranteed.

3.3 Backend Web Application Development

The backend of the presented web application largely implements the acquisition of data as presented in Sections 3.1 and 3.2. Moreover, raw information mined via the APIs is processed and merged before being visualized in the frontend. Four technologies are used and intertwined in order to make the application reasonably fast and useable: PHP, JavaScript, JSON and AJAX. The rest of this subsection covers the use of these technologies and explains why the author chose certain programming languages for specific application tasks.

PHP [12] is a server-side scripting language used for web development. It can be embedded directly into HTML (HyperText Markup Language) documents, thus adding dynamic flexibility to the approach. For the implementation of the tool at hand, it was mostly used as a "container" language, providing a *.php* ending to the five main files of the application. While the *index.php* file is largely written in JavaScript, the functionality of the application's data acquisition part is implemented in PHP. Each of the four different API operations is processed in its own file, namely in $getArtistInfo.php$, $getLocationsLastFm.php$, $getLocationsSongkick.php$ and $getTwitterActivity.php$. These files are all included by the *index.php* master file and called when necessary.

Essentially, all API service queries follow the same principle. First, a connection to the respective service is established either by specifying the API key (`Last.fm` and `Songkick` API) or by employing the OAuth authentification tool (`Twitter` API). Then, the necessary data is fetched using JSON and AJAX. While the former is a JavaScript-derived data interchange standard representing arrays, the latter denotes client-side technologies with which it is possible to asynchronously transfer data to and from a server (e.g. independent from other operations). After the request is processed, the API returns an object which can be fragmented in a way specified by the respective documentation in order to parse




```
<html>

<head>
<title>Bachelor project</title>
<meta charset="utf-8">
<?php
include "getArtistInfo.php";
include "getLocationsLastFm.php";
include "getLocationsSongkick.php";
$twitterMode = $_GET["twitterMode"];
if ($twitterMode == null || $twitterMode == "") {
    $twitterMode = false;
}
?>
<link rel="stylesheet" href="css/style.css"></link>
<link rel="stylesheet" href="css/buttons.css"></link>
<link rel="stylesheet" href="css/slider.css"></link>
<link rel="stylesheet" href="http://code.jquery.com/ui/1.10.3/themes/smoothness/jquery-ui.css"></link>
<script type="text/javascript" src="http://maps.google.com/maps/api/js?sensor=true&libraries=geometry"></script>
<script type="text/javascript" src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script type="text/javascript" src="http://code.jquery.com/ui/1.10.3/jquery-ui.js"></script>
<script type="text/javascript" src="scripts/template.js"></script>
<script type="text/javascript" src="scripts/dependClass.js"></script>
<script type="text/javascript" src="scripts/draggable.js"></script>
<script type="text/javascript" src="scripts/slider.js"></script>
<script type="text/javascript">
var twitterMode, sliderUnlocked, noTourStopsForArtist, numberPastTourStops, maxNumberOfTweets;
var start, startMarker, markersTweets, markersLocations, map, mapOptions, infoWindow, finished, output;
var artistInfo, tweets, tweetsCoordinates, tweetsMarkers, locations, locationsLastFm, locationsSongkick, locationsMarkers;

// checking for browser's geolocation functionality
function checkGeoloaction() {
    if (navigator.geolocation) {
        $("#slider").slider({
            callback: function() {
                if (sliderUnlocked) {
                    filterLocations();
                }
            }
        });
        $("#slider").slider("value", 0, 365);
        $("#checkboxTourStops").click(function() {
            if (document.getElementById("submit").disabled == true) {
                if ($(this).is(":checked")) {
                    toggleMarkersAndInformationLocations(markersLocations, true, numberPastTourStops);
                } else {
                    toggleMarkersAndInformationLocations(markersLocations, false, numberPastTourStops);
                }
            }
        });
        $("#checkboxTweets").click(function() {
```

Fig. 3. Interaction between the different programming languages used in the web application's backend.

the desired data items. For instance, the unique MBID of an artist can be stored using the *data.artist.mbid* variable of the returned object. The AJAX technology's asynchronism initially made it difficult for the author to correctly implement proper synchronization of the API requests. To avoid any problems with synchronization, a procedure call system was developed to make sure each operation is finished before handing over control to the supervising method.

Already mentioned before, the *index.php* master file is used to control the functionality of the application while temporally transferring control to the files used for data acquisition and subsequently collecting and merging the resulting data arrays. It is largely written in JavaScript [13], which ultimately is a computer programming language used as part of web browsers. While PHP operates server-side, JavaScript is used for client-side interaction with the user. Therefore, cooperation between these two technologies is more often than not rather cumbersome. However, the author deemed it best to implement the application's main functionality in JavaScript for the reason of being able to use certain constructs solely available in this programming language. Figure 3 shows the interaction between the HTML, PHP and JavaScript programming languages in the *index.php* file: while the framework is implemented in HTML (note the opening *html* and *head* tags), a short snippet including data acquisition files into the application and setting a parameter is written in PHP. However, checking the browser's geo-location capability (cf. line

marked in yellow) is programmed in JavaScript, as are several statements specifying the inclusion of external files.

The modus operandi of the functionality implemented in *index.php* is explained in the following paragraphs. Note that most implementation decisions of the backend are not visible in the frontend, but rather manifest in the way the application processes the data. Upon loading the website, all data variables are reset and initialized anew to avoid inconsistencies. Subsequently, user input is checked and validated. Should the inserted artist name not exist in the `MusicBrainz` database (even after being auto-corrected by the `Last.fm` API), a corresponding error is shown. If an artist's name is valid, however, their data is fetched and event information is loaded as explained at the beginning of this section.

Tweet messages mentioning the selected artist are thereafter mined using the respective API service. Because of the sparsity of coordinates attached to tweets, exact locations have to be calculated before being visualized on a map. Fortunately, the `Google Maps` API (cf. Section 3.4 for more detail) provides an excellent geo-coding tool, which the application employs in order to derive necessary location data. Unfortunately, due to the low per-second query limit of the geo-coding tool, the application's reaction time would become intolerably long when including more than 50-100 tweets (cf. Section 5.3 for more details on this problem), explaining the small number of messages the web application is requested to acquire.

After all the data is mined and stored in its respective arrays, a JavaScript function is called that filters the event locations in order to avoid duplicates (e.g. data items contained in both arrays). Since the application's two event data sources `Last.fm` and `Songkick` are independent from each other, no guarantee can be given for the mutual conformity of their contained information. After due consideration, the author nevertheless decided to put enough trust in each of these sources to use all of their separate events, not only those that are congruent in both. Therefore, the event filter mechanism simply merges the two resulting information arrays into one.

Before being visualized on a map, all the event data items are reviewed regarding the existence of geo-coordinates. Fortunately, most of the locations fetched from `Last.fm` and `Songkick` already feature exact longitude/latitude information, leaving the geo-coding tool with only a small fraction of events lacking geo-data. Since the aforementioned geo-coding tool also works vice versa (e.g. computing city and country names for a set of coordinates), every tour stop will be supplied with both geo-data and human-readable location information after its employment. Problems with certain events that feature a double zero coordinate pair and thus are placed in the center of the equator have been solved in the course of development.

3.4 Frontend Web Application Development

While the preparation of event data is explained in the previous subection, the development of the web application's frontend (i.e. the part of the application actually visible to the users) is covered in the subsequent paragraphs. Specifically, the use of `Google Maps` as core of the frontend is highlighted and several design decisions are justified. It is assumed that the status quo involves two arrays of ready-to-be-visualized event and microblog data respectively. Prior to the visualization process, however, the GUI (Graphical User Interface) of the application is introduced. Figure 4 shows the initial state of the GUI, with which users are presented before entering any information. The yellow marker, which the map is centered around, denotes the user's location (as determined through geo-localization). Note that in addition to the "Submit" and "Reset" buttons, a slider is included as filtering option in order to select an arbitrary time slot for the shows to be displayed. The range of this slider is a maximum of 365 days, both back and forth in time. Moreover, two radio boxes offer users the possibility of showing and hiding tour stops and `Twitter` activity on the map respectively.

Users can employ the GUI to search for desired artist names. Any error in the input will be indicated accordingly. Since the data has already been processed when the execution of *index.php* reaches the visualization mechanism, the `Google Maps` API can be used to place location markers on the map. Allocating red color to markers which denote future events, and coloring those which denote past events green, the markers are subsequently placed on the map by creating for each marker a new object and assigning as source parameter the respective map object. Each of the differently colored sets of markers is chronologically numbered in order to give the user an overview which events occur first. Analogously to creating event data markers, a set of blue markers is placed on the map denoting tweets mentioning the selected artist. However, these markers are not numbered because of the temporal vicinity of tweets. By calling the *bound.extend* and $map.fitBounds$ methods supplied by the `Google Maps` API, zoom level of the map is adjusted to fit the scattered markers.

In addition to placing markers on the map and thus visualizing both event and microblog data, helpful information is shown on the right side of the application's website. Although this approach is explained in the next section in more detail using screenshots of the GUI, the basic notion is nonetheless drafted in this paragraph. In order to give users an overview which show will be happening next, or has happened previously, the first item of future event dates as well as the last item of past events dates are selected. The implementation of this process indexes a date i which happens in the future (i.e. today or any other day thereafter) and for which the directly preceding date $i-1$ has happened in the past. Both i and $i-1$ are then displayed as next and previous tour dates respectively.

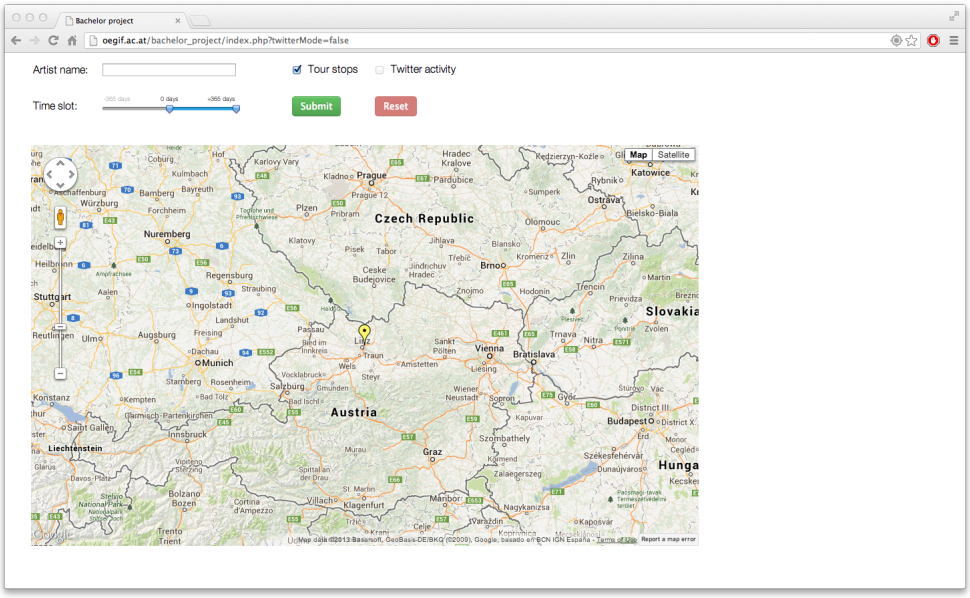


Fig. 4. The web application's graphical user interface in its initial state.

Naturally, the choice of which piece of information to display is dependent on the data supplied by the time slot slider. If users decide to only select past event dates, then date i is obviously not shown on the right side of the website (and vice versa). Information about the nearest tour stop, however, is displayed at all times. It is calculated by the `Google Maps` API too, returning the linear distance between two locations when being supplied with the respective sets of coordinates. Unfortunately, computing driving distances between arbitrary locations requires use of a different API service with a per-second query limit too low to be reasonably adopted by the application (cf. Section 5.3 for more information). Hence, it might be possible that users find themselves being suggested locations that are closer as the crow flies while nevertheless requiring a longer drive.

4 Demonstration of Functionality

In this section, functionality of the application is demonstrated and exemplary use cases are presented. A two-fold approach is taken, describing the application's functionality without and with visualization of `Twitter` activity respectively. This distinction is necessary because of the different reaction times of the tool. While the version renouncing `Twitter` activity is sufficiently fast to be deployed in real-life scenarios, additionally including tweet visualization results in an intolerably long response time of the application, making this use case applicable for scientific purposes only (cf. Section 5.3 for more detail).

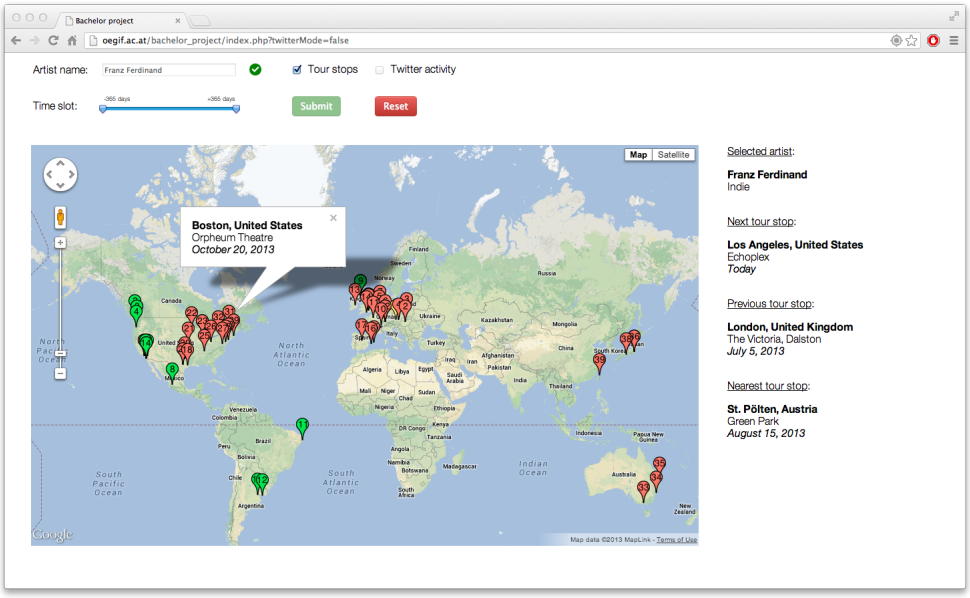


Fig. 5. The web application's graphical user interface handling a query for Franz Ferdinand issued on July 30, 2013, with focus on the Boston show.

4.1 Tour Stop Visualization

As shown in Figure 4, users are initially presented with a configuration in which every option is reset, giving them the chance to freely choose an artist's name as well as a time slot for shows to be displayed. The checkbox triggering the depiction of tour stops is selected by default, whereas the checkbox controlling the display of `Twitter` activity is deactivated. Upon specifying a musician's name, event data is mined as explained in Section 3. In order to indicate the process of information mining, two spinning arrows are displayed as long as the application is busy. When being finished with the task, a green check mark indicates success or, alternatively, a red cross indicates failure. While the latter use case is examined in Section 4.3, the former is discussed in this subsection.

Depending on the setting of the time slot slider, a multitude of markers indicating shows is displayed on the map following the selection of an artist. Already explained in Section 3.4, future events are displayed using red markers, whereas past events are depicted by green markers. Figure 5 shows the event visualization result for a query placed on July 30, 2013 specifying the Scottish indie rock band Franz Ferdinand. The mouse pointer is focused on the Boston show, thus displaying further information about the corresponding concert. Note that four pieces of information are displayed on the right side of the website: selected artist (including genre), next tour stop, previous tour stop and nearest tour stop. While the first data item is essentially comprised of the artist's

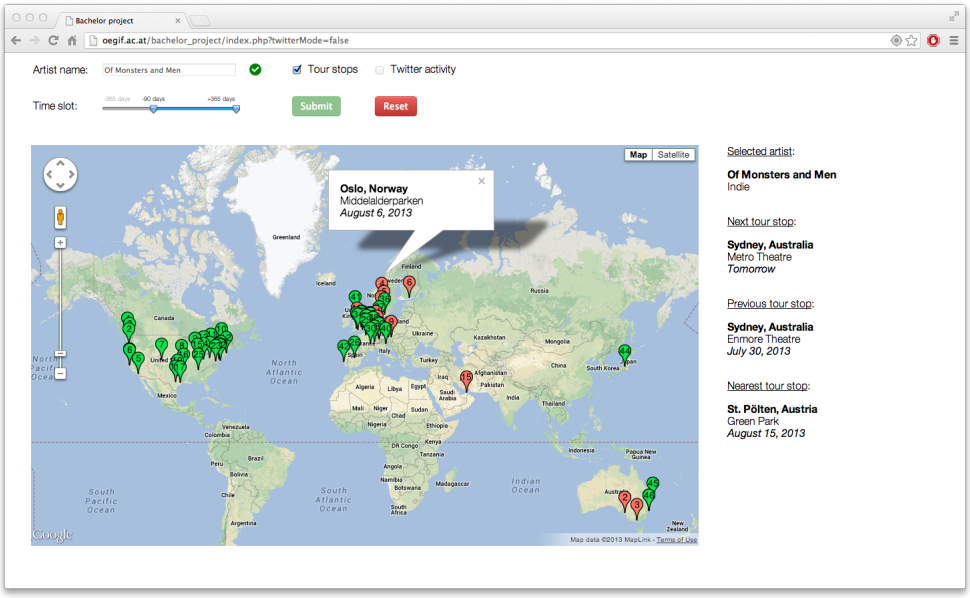
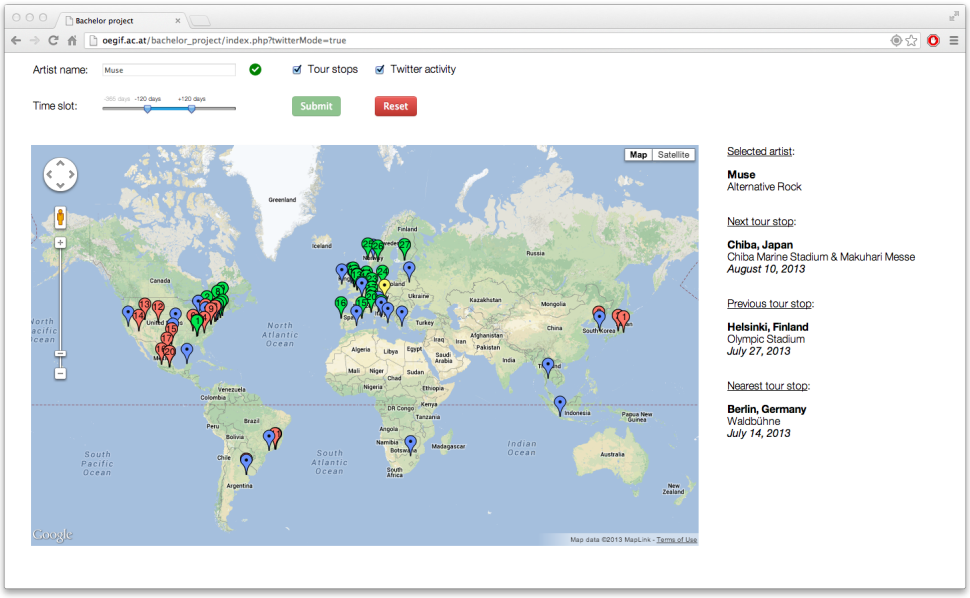


Fig. 6. The web application's graphical user interface handling a query for Of Monsters and Men issued on July 30, 2013, with focus on the Oslo show.

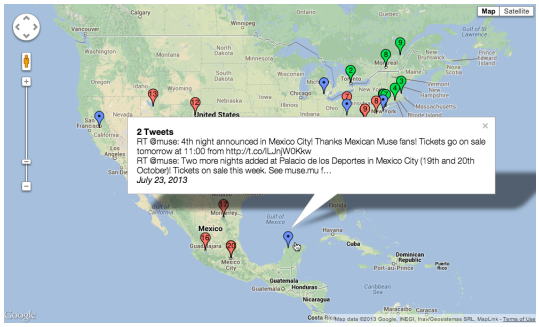
details, the last three are calculated dynamically as described in Section 3.4. Modifying the settings of the time slot slider also changes the respective information snippets in order to be consistent with the shows displayed on the map.

Only the most important properties are displayed for each of the three pieces of information denoting tour stops: city and country, venue as well as date of the show. Concerts happening the day before the query, the day of the query or the day after the query are not described by means of their actual dates but rather by using "Yesterday", "Today" and "Tomorrow" respectively. While the next and previous show items identify temporal aspects of the application, the nearest show item describes its spatial properties. Already explained in Section 3.4, the latter is calculated using the linear distance between a user's location and the nearest tour stop.

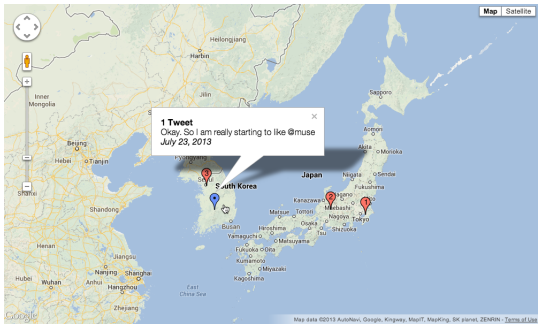
Figure 6 shows another use case of the tour stop visualzation tool, displaying shows by the Icelandic indie rock band Of Monsters and Men as queried on July 30, 2013. Focus is set on the Oslo show by hovering over the respective marker with the mouse. While the query in Figure 5 aims at locating every show played by the band in the last and next year respectively, the settings in Figure 6 limit the time slot of past shows to 90 days while leaving the time frame for future shows in its initial state. Still, it is clear upon inspection of both use cases that a query involving Franz Ferdinand yields a majority of red markers, while a search for Of Monsters and Men results in a multitude of green



(a) overview of the query



(b) detailed inspection of a tweet with spatial reference to a nearby show



(c) detailed inspection of a tweet without spatial reference to a nearby show

Fig. 7. The web application's graphical user interface handling a query for Muse issued on July 31, 2013, with focus on two markers in Mexico and South Korea respectively.

markers (even though the time slot for past concerts is restricted). It can hence be inferred that the latter band is close to finishing their tour while the former has just started. This theory is backed by consulting the international release dates of the artists' latest studio albums, which are April 2012 (Of Monsters and Men) and August 2013 (Franz Ferdinand) respectively.

Additionally, the information on the right side of the websites displayed in the two figures reveals an overlap in music genre and nearest tour stop of both bands. This coincidence can be explained by consulting the line-up of the event allegedly happening in Austria's St. Pölten on August 15, 2013. Indeed, both Franz Ferdinand and Of Monsters and Men are billed to play the first day of the respective rock music festival called Frequency [14], validating the information specified by the visualization tool.

However, wrong information is given about Of Monsters and Men's past and future shows in Sydney, specified in the respective sections. Inspection of the band's [15] as well as the venue's [16] official websites reveals three shows booked in the local Enmore Theatre, the last of which having been upgraded from the smaller Metro Theatre. Apparently though, the event directories employed by the application fail to provide this new venue information, resulting in the July 31, 2013 show still to be displayed to take place in Metro Theatre. Furthermore, while the date of the aforementioned show has correctly been changed into "Tomorrow" (for a query issued on July 30, 2013), the show happening on the date of the query hasn't been converted into "Today" and, in addition, is already denoted as a previous tour stop. This problem arises from the discrepancy between event data contained in `Last.fm` and `Songkick`, which the application at hand attempts but not always manages to compensate. For more information regarding this problem, refer to Section 5.1.

4.2 Twitter Activity Visualization

In addition to map representation of events, tweet visualization has been incorportated into the application too. The basic notion is that, upon being presented with a spatial distribution of a band's fan bases (as measured by the quantity of the respective artist's mentions on `Twitter`), their management might be able to schedule a tour more accurately, both reaching a bigger audience and improving the percentage of sold tickets for shows. Figure 7 shows the past and upcoming tour dates during a period of 120 days respectively for the English rock band Muse. Moreover, tweets mentioning the band's official `Twitter` account by including at the beginning of the text *@muse* are displayed on the map using blue markers. Note that the checkbox controlling the display of `Twitter` activity is not deactivated in this use case but selected. Although the query was placed on July 31, 2013, most of the tweets originate from July 23, 2013, the reason of which being the `Twitter` Search API's nature of making public a tiny fraction of all posted tweets only, more often than not in a temporally random manner.

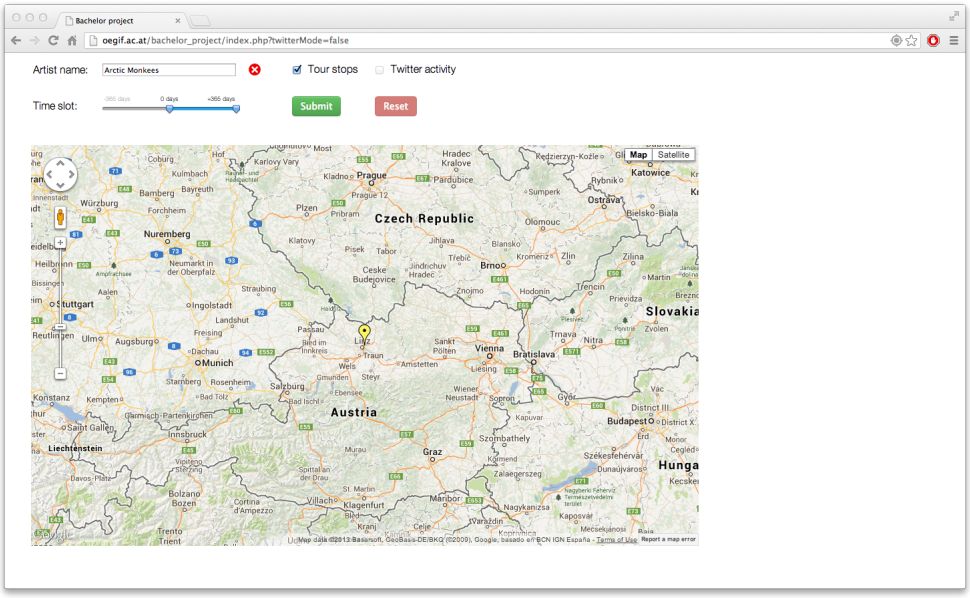


Fig. 8. The web application's graphical user interface displaying an error when handling a query for "Arctic Monkees" issued on July 31, 2013.

It is clearly visible that most of the band's concerts have been or will be played in Europe and North America, with only a couple of shows scheduled in South America and Asia. This trend is observable for just about every query that has exemplarily been placed during the development of the tool (with the exception of queries specifying locally famous bands). The reason for this is the disproprotionally high wealth that people living in Europe and North America enjoy. Since attending a concert is essentially not a necessity but rather a recreational activity, more often than not potential audiences living in developing countries are not able or willing to pay large amounts of money in order to attend a show.

However, since `Twitter` is free to use for everyone around the globe, the possibility of issuing tweets mentioning a specific band is not limited to or dependent on the wealth of a country. This is reflected in Figure 7 (a), where a fraction of the exemplary set of tweets is located in countries where no tour dates are scheduled (e.g. Belarus, Greece, Indonesia, South Africa and Thailand). However, the majority of tweets are roughly placed around the areas where most of the concerts are played. Although too small a data set to draw universal conclusions from, the experiment conducted in Figure 7 (a) nevertheless offers an initial attempt to quantitatively determine fan dedication around the world.

While the aforementioned subfigure provides an overview of the query, Figures 7 (b) and 7 (c) aim to illustrate the difference in tweets generated by users. Although the

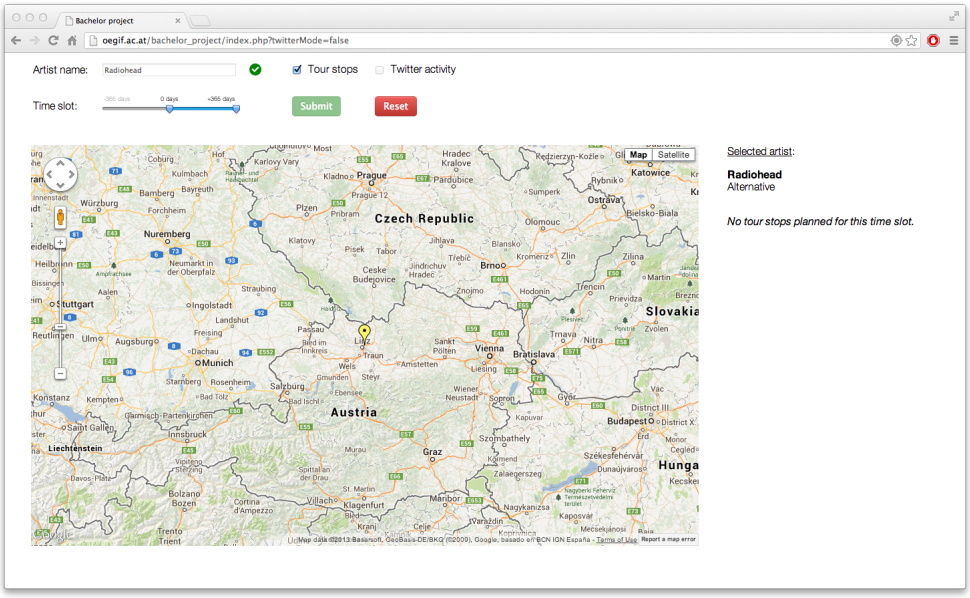


Fig. 9. The web application's graphical user interface displaying an error when handling a query for Radiohead issued on July 31, 2013.

author expects the fraction of concert-related tweets to grow when a tour stop in spatial vicinity is either announced or about to happen, this phenomenon seems not always to be the case. While the tweet shown in Figure 7 (b) effectively deals with an increase in the quantity of shows given by Muse in Mexico City (thereby being issued somewhere near the Mexican city of Merida), the tweet displayed in Figure 7 (c) solely denotes the user's recently found pleasure in listening to the band's music, ignoring the fact that a concert is scheduled nearby in South Korea's capital city Seoul. Although the majority of tweets mentioning an artist is of former nature (i.e. referencing a show happening in spatial vicinity), a couple of users issue tweets of the latter form too. However, since the work at hand not only aims at measuring the excitement about a show but also about a band in general, both categories of tweets are crucial for determining the spatial distribution of an artist's fan base.

4.3 Error Cases

Two error cases are considered by the application, accounting for the non-existence of a specific artist as well as for a band not being on tour at the moment. As indicated at the beginning of this section, failure to find an artist's name in the `Last.fm` and `Songkick` databases is indicated by displaying a red cross on the right side of the text input field. Note that the auto-correction offered by the `Last.fm` API is able to avoid a couple of

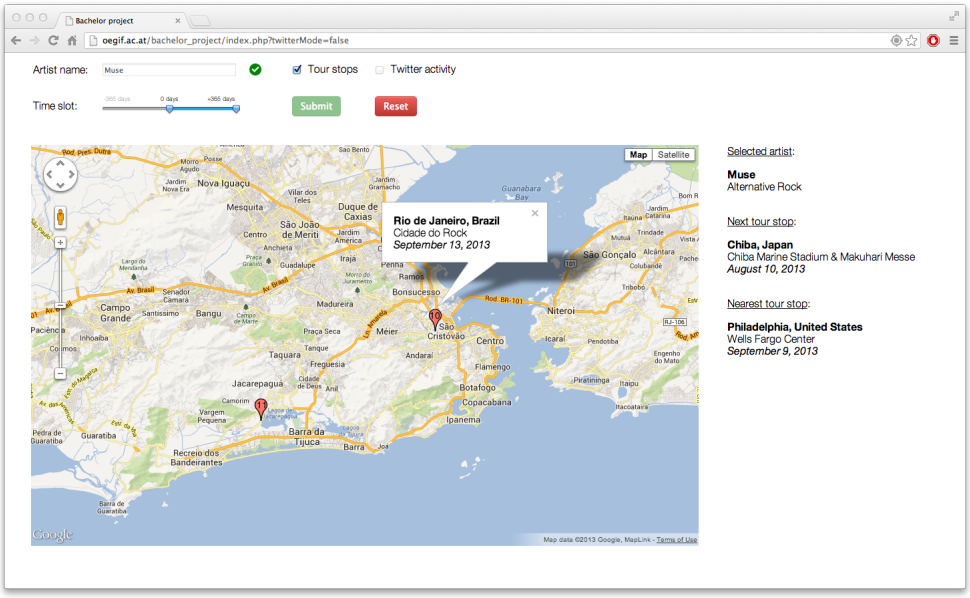


Fig. 10. The web application's graphical user interface displaying an example for data discrepancy when handling a query for Muse issued on August 2, 2013.

errors due to incorrectly spelled artist names prior to displaying a result. For instance, the English indie rock band Arctic Monkeys is detected by the tool even if a user wrongly uses the term "Artic Monkeys" in the query. However, an error is displayed when searching for "Arctic Monkees", as shown in Figure 8.

While the aforementioned error case is not likely to change over time (except for the very rare occasion when a band with exactly the "incorrectly" spelled name of another artist is suddenly formed), the second error case of an artist currently not being on tour is more likely to vary dynamically according to temporal parameters and slider properties. Figure 9 depicts this error case for the English alternative rock band Radiohead, using a slider position covering a time frame from 0 to 365 days into the future. The error message indicates that no tour stops are planned by the band for the specified time slot. However, modification of the slider to additionally accomodate past events would get rid of the error by showing a selection of green markers denoting shows previously played by the band. Obviously, the error would persist for music groups that are disbanded for a long time already (like the English rock band The Beatles), due to the artists not having played or scheduled a show in the past and upcoming year respectively.

5 Problems and Drawbacks of the Implementation

During the implementation of the tool several problems arose, all with different complexity and severity. This section aims at giving an overview of these drawbacks while suggesting ways of solving them.

5.1 Discrepancy of Event Data Directories

Perhaps the most pressing problem exhibited by the application is the discrepancy prevalent in the process of mining data from the `Last.fm` and `Songkick` event data directories. While each of the services offers acceptable accuracy of the data compared to a ground truth (cf. Section 6 for an evaluation thereof), combining the information of both directories often yields data records which, although describing the same event, are nevertheless slightly different from each other. For instance, whereas the `Last.fm` API might not include a specific event's venue name, the `Songkick` might. This deviation sometimes leads to two chronogically numbered events that are subsequently placed on the map, one located in the respective city's center and the other located at the exact position of the venue. Moreover, these virtually redundant events sometimes exhibit different dates, especially when being part of a music festival. While `Last.fm` might define the date of an artist's show as the respective festival's starting date, `Songkick` might correctly identify the day the band is actually playing on (e.g. the second day of the festival).

Figure 10 shows a close-up of two markers in and around Rio de Janeiro for a query involving the English rock band Muse. Both markers denote shows that are part of a local festival (Rock in Rio) as well as the band's tour. The marker labeled "11" shows the event's correct data, identified through inspection of the band's official website [17]. In fact, the small `Google Maps` widget embedded into Muse's website confirms the location of the aforementioned marker to be in the western part of the city. Furthermore, the event date is attested to be September 14, 2013. However, the marker labeled "10" (on which the focus is set in Figure 10) claims the show to take place in downtown Rio de Janeiro on September 13, 2013. Although it is not clear which of the two event data directories provides the incorrect piece of information, the example at hand is one of the few times the application is unable to filter out and compensate such discrepancies.

Although some deviations persist, most can be avoided by employing the following notion: first of all, it is checked whether the array resulting from querying the `Songkick` API is empty. If this is the case, only the data array retrieved from `Last.fm` is considered for visualization - otherwise, both arrays are considered. However, before consolidating these two data sets, each of them is individually scanned for duplicate event items, which manifest in being scheduled on the same date and in roughly the same area (a difference in GPS coordinates lower than 1 is arbitrarily declared to describe spatial vicinity). After removing redundant events inherent to the arrays, a cross-check is performed, deleting

in the `Songkick` result array all data items that are already contained in the `Last.fm` result array. Finally, the remaining events are written into a newly created array object. Already mentioned in the previous paragraph, this two-fold approach is able to filter out most discrepancies, although at times the algorithm fails to recognize deviating event data denoting the same show, as exemplified in Figure 10.

Besides the discrepancy concerning event data, an implementation decision regarding location names entailed confusion in the first steps of development. Whereas `Last.fm` defines cities located in Australia, Canada and the United States to be of the form $city, country\_long$ (e.g. "San Francisco, United States"), `Songkick` includes the federal state as well, resulting in location names to look like $city, state, country\_short$ (e.g. "San Francisco, CA, USA"). In order to keep data items consistent with countries other than the aforementioned three, the author chose the former approach offered by `Last.fm` for naming event locations. This decision, however, led to confusion regarding the placing of markers on the map, both within and among countries. For instance, there are several dozen cities in the United States named Greenville, Franklin or Clinton [18]. Exemplarily trying to place an event marker in one of those cities, the application frequently ended up choosing the wrong location. Furthermore, some concerts scheduled to take place in Australia were suprisingly "moved" to Canada, due to the same acronym denoting the former's Northern Territory and the latter's Northwest Territories. In the course of programming, however, the author managed to solve these problems by further integrating coordinates into the application.

5.2 Geo-Location Sparsity

Already mentioned throughout the paper, a drawback that becomes noticeable when being presented with the task of visualizing tweets is the sparsity in geo-locations inherent to these data items. Hauger and Schedl [26] refer to the same problem in their work, which also aims at placing tweets on a map. The authors claim that geo-coordinates in tweets are still virtually unavailable as they require GPS-enabled devices, which came into being solely in the last couple of years. Moreover, `Twitter` only recently decided to stronger integrate coordinates into its users' tweets. While the work at hand is unable to employ the `Twitter` Streaming API for a long time due to its temporally limited development period, the application presented in [26] isn't, thus yielding a large enough set of tweet data featuring GPS coordinates to work with.

Since no geo-location information is available for the tweets mined through the `Twitter` Search API employed by the application at hand, users' location and timezone are used instead to determine their position. This approach, however, is more prone to yielding erroneous information due to the relative inaccuracy of the aforementioned parameters. For instance, some users who like to listen to the music of the English rock band Muse might register as their location "Cydonia", referring to the band's most popular song

Knights of Cydonia. Moreover, the author has encountered several `Twitter` profiles denoting as timezone "Greenland". Of course, only tweets for which the `Google Maps` API geo-coding tool is able to compute an exact location can be placed on the map. While the API service correctly spawns coordinates for a query specifying "Buenos Aires" as location and "Argentina" as timezone, it nevertheless is incapable of calculating geo-location information for a query specifying "Buenos Aires" and "Brazil" respectively. Because of this necessity of matching pieces of information, the number of visualized tweets might be lower than the quantity of 50-100 that is initially mined for each query, thus reducing the application's ability to spatially determine artists' fan bases. In Section 7, it is discussed how future work might circumvent this issue by mining more tweets as well as placing them on the map with higher accuracy.

5.3 Reaction Time Issues

Another problem faced by the application arises from the limit set by the `Google Maps` API geo-coder, incorporated into the service in order to avoid spam. Although the API documentation specifies as query limit 2,500 requests per day [19], it has been discovered during development of the application that roughly 20-25 data items can consecutively be calculated by the service (in the course of a couple of milliseconds) before resulting in an over query limit error. In general, this restriction is not a problem when dealing with the visualization of events. Out of approximately 50-100 tour dates played by an arbitrary artist (both past and upcoming), about 10 percent are lacking coordinates or the city and country names respectively, resulting in the manageable quantity of 5-10 event items that need to be supplied with additional information. When considering tweets, however, all 50-100 mined tweets are lacking coordinates due to geo-location sparsity described in the previous subsection. Therefore, the geo-coding tool has to be employed more often, each time being assigned a subset of tweets small enough not to cause an over query limit error. Since the API service needs to pause a couple of seconds after having processed a subset, the calculation of geo-information results in the application's long response time, making the inclusion of `Twitter` messages applicable for scientific purposes only.

Much like the geo-coding service, the API's directions service also features the aforementioned drawback by exhibiting too low of a per-second query limit. In the application at hand, the directions tool could be employed in order to calculate the piece of information denoting the nearest show, as displayed on the right side of the website (cf. Sections 3.4 and 4.1). However, doing so would also result in multiple requests to the API service together with its standby time, thus delaying the application's reaction time even more. Since the calculation of distances is used for every tour stop of each query, even the event data-only use case of the tool would become virtually unusable when additionally employing the directions service. Therefore, the author has decided at an early stage of development to resort to the `Google Maps` API distances service instead.

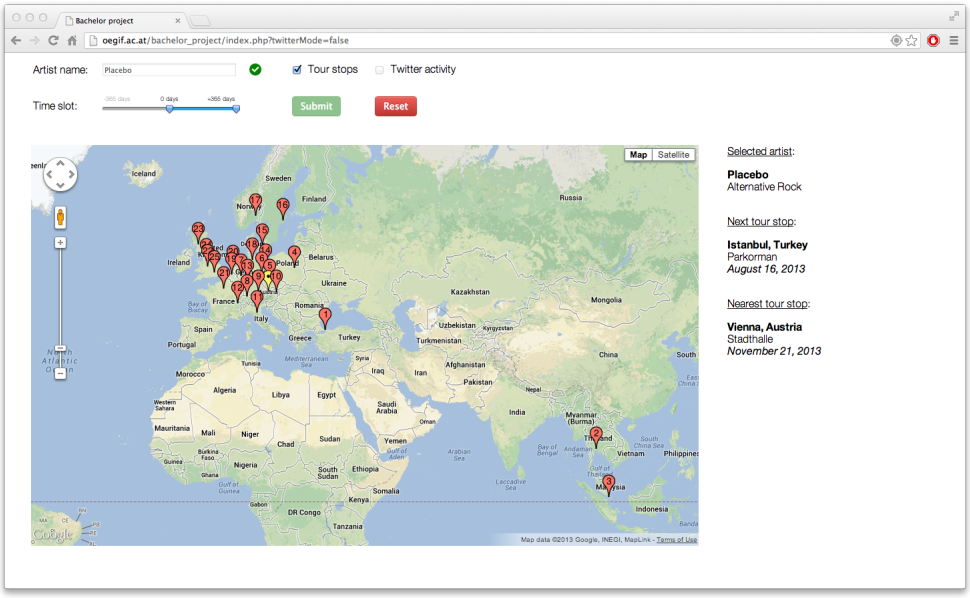


Fig. 11. The web application's graphical user interface handling a query for Placebo issued on August 4, 2013.

Another issue concerning reaction time results from the use of JSON and AJAX. As explained in Section 3.3, executive power is first handed from the *index.php* file to various API request procedures and subsequently back to the master controller. Since an AJAX request usually takes some time to finish, consecutive execution of all three queries (first mining the artist info array and afterwards the two event info arrays) might result in a mixup when pressing the "Submit" button too early or too frequently. As a result, data items cannot be read properly, spawning an error to that effect. Fortunately, tests of the finished tool reveal no such problem when solely considering the event data visualization mode. However, occasionally the error still occurs in the tweet visualization mode, emphasizing the conclusion that this use case be subject to application in scientific environments only.

6 Exploratory Evaluation

In order to determine the accuracy of the application at hand as well as to demonstrate its ability to correctly visualize an artist's tour stops, mined event data is evaluated against a ground truth obtained by human inspection of the respective artist's website. Two examples of the evaluation process are presented in more detail as follows. Evaluation is done for event data only, because of the lack of suitable ground truth regarding `Twitter` data.

Due to the same concept of visualizing tour stops employed by the tool at hand and the website of the English rock band Placebo presented in Figure 1, accuracy of the visualization approach is exemplarily measured for this artist first. Facilitating the evaluation process, only upcoming tour stops are considered in order to avoid the difficulty of locating past shows on the band's website. Evaluation classification is two-fold, featuring both general and detailed criteria used for determining accuracy (which is measured in percent). General criteria solely consist of quantitative correctness of tour stops, whereas detailed criteria consider every tour stop separately, denoting fidelity of date, city and country as well as venue respectively. The following formulae are introduced in order to calculate accuracy:

$$acc_g = 1 - \frac{|num\_stops_a - num\_stops_t|}{num\_stops_t} \tag{2}$$

$$acc_d = \frac{corr\_dates_a + corr\_cities_a + corr\_venues_a}{num\_stops_a * 3} \tag{3}$$

$$acc = \frac{acc_g + acc_d}{2} \tag{4}$$

In the first of the above equations, acc_g denotes general accuracy, calculated by dividing the absolute quantitative difference in tour stops featured in the application a and ground truth t respectively by the number of tour stops featured in the latter, subsequently substracting the resulting value from 1. The second equation, on the other hand, identifies the detailed accuracy measure acc_d by calculating the number of the application's dates, cities/countries and venues consistent with the ground truth respectively, before dividing their sum by the tripled number of tour stops. Finally, the average of acc_g and acc_d is denoted with acc, measuring the total accuracy of the application's retrieval approach.

Before being able to evaluate the tool's performance, an exemplary query specifying Placebo has to be issued in order to mine the artist's event data. To that effect, Figure 11 depicts the visualization of all of the band's upcoming concerts in the next 365 days. Corresponding ground truth is acquired through Placebo's website [8], exemplarily shown in Figure 1. In total, 25 markers are placed on the map by the application at hand, ranging from a show in Istanbul on August 16, 2013 to two concerts in London on December 16, 2013 and December 17, 2013 respectively. Since the application conveniently combines into one single marker multiple shows consecutively happening in the same venue, the number of markers doesn't necessarily correspond to the number of shows. However, upon closer inspection of the upcoming tour the only multiple dates to be played by the band seem to be the aforementioned London shows, raising the total number of Placebo's tour stops to 26. This value is confirmed by manual examination of the artist's website, resulting in perfect general accuracy of $acc_g = 100\%$.

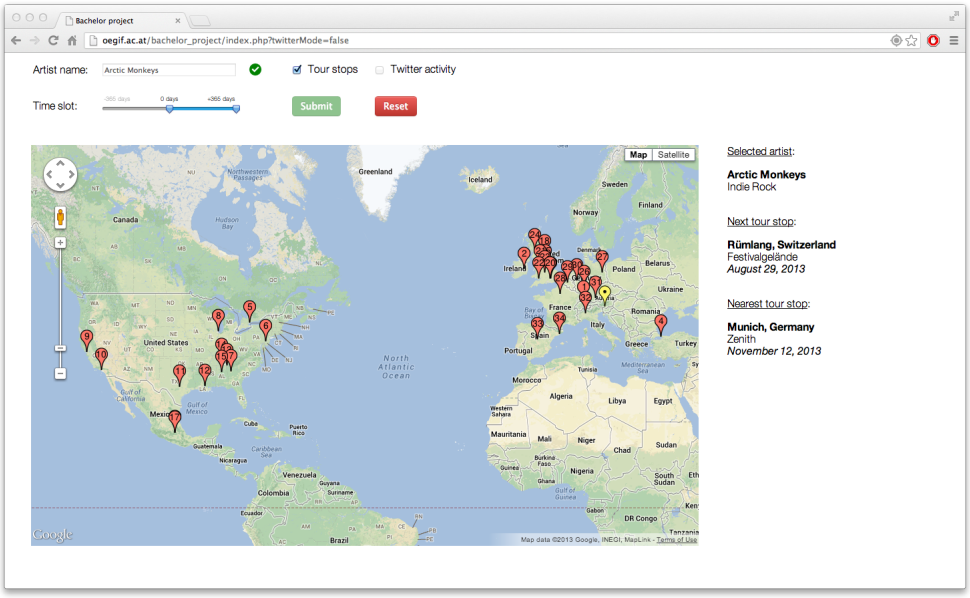


Fig. 12. The web application's graphical user interface handling a query for Arctic Monkeys issued on August 4, 2013.

Inspection of every show's detailed information yields the following results: of all 26 concerts played by the artist, every show's date is correctly identified by the application compared to the ground truth. Regarding city and country names, solely two names displayed by the tool are deviating from the information on the website, namely "Casalecchio di Reno" denoting the Bologna show and "Frederiksberg" indicating the concert in Copenhagen. Although in both cases, the diverging names respectively identify the city's suburb where the concert venue is located rather than the city itself, detailed accuracy acc_d is nevertheless reduced due to these mistakes.

Finally, venue names of the shows are evaluated against the ground truth. Interesting enough, inspection of the first two tour stops displayed on the website (happening in Istanbul and Bangkok respectively) reveals their venue information to denote festivals the band is booked to play rather than the actual names of the venues. Subsequently conducted research confirms the venues of these two festivals to be "Parkorman" and "Impact Arena", coninciding with the information entailed in the application at hand. Because of the tool's aim to display actual venue information instead of potential festival names, the author therefore decided to consider the aforementioned instances as accurate. However, two other venue names are truly deviating, namely Stockholm's Stora Arenan (displayed in the application as "Arenan, Fryshuset") and Paris' Bercy (shown as "Palais Omnisports"). As a result, detailed accuracy is reduced further, resulting in its value

of $acc_d = 94.87\%$. Combined with the general accuracy value computed before, total accuracy of the application regarding the example at hand is $acc = 97.44\%$, indicating good performance of the tool in comparison to the ground truth.

Another exemplary evaluation is conducted through the analysis of upcoming tour dates played by the English indie rock band Arctic Monkeys. Figure 12 shows the visualization result for all the concerts given by the band in the next 365 days. A total of 34 markers are numbered chronologically, denoting shows ranging from Zurich on August 29, 2013 to Barcelona on November 16, 2013. Again, the number of markers doesn't necessarily have to correspond to the number of shows. In fact, quite a few double and even triple dates are played by the band in the United States and England respectively, raising the total number of concerts to 42. However, ground truth acquired through examination of the Arctic Monkeys' website [20] only reveals information about 38 tour stops, resulting in general accuracy of $acc_g = 90.48\%$.

Subsequently, every concert's information is cross-checked in order to compute detailed accuracy. Concerning the shows' dates, the application is able to assign correct values to 34 of 42 tour stops. Errors are mostly made due to a problem involving festivals and spatial vicinity, as indicated in Section 5.1. Inspection of city and country names reveals three deviating pieces of information, namely "Rümlang" denoting the Zurich show, "Co Laois" referring to Ireland's Stradbally concert and "Badalona" indicating the tour stop in Barcelona. Once again, the respective cities' suburbs featuring concert venues are displayed in lieu of the city names themselves.

As shown before, the last part of the evaluation process consists of comparing the application's yielded venue names against a ground truth scraped from the band's website. Not unlike the preceding example, venue names denoting festival sites are considered to be correct. Thus, every venue name is accurately represented by the tool at hand, resulting in detailed accuracy of $acc_d = 91.27\%$. Subsequently, total accuracy of the application regarding the current example is $acc = 90.87\%$, indicating good performance compared to a ground truth too. Although acc changes dynamically depending on parameters such as artist name and slider settings (as seen in the foregoing example evaluations), it is estimated that both resulting values of more than 90 percent serve as reference points for further evaluation.

7 Summary and Future Work

The paper at hand presented a tool extending the text-based listings of event directories such as `Last.fm` and `Songkick` in favor of a novel visualization approach using `Google Maps`. The rationale is that fans are rather interested in being able to locate shows in spatial vicinity by consulting a map than scanning through extensive lists of events. Furthermore, the tool at hand was augmented with microblog data, effectively displaying

`Twitter` messages in order to discover the locations of artists' dedicated fan bases and thereby assisting musicians and their management in the planning of tours. First, related work was introduced and analogies were discussed. Thereafter, the application was described in more detail as well as its functionality demonstrated by means of different use cases. Furthermore, drawbacks of the implementation that limit the aforementioned functionality were explained and potential solutions outlined. Subsequently, visualization results were evaluated, thereby revealing good accuracy of the tool in comparison to a ground truth acquired through artists' websites.

Regarding future work, several improvements can be made in order to solve the application's problems and enhance its functionality. For instance, it has been discovered that geo-location sparsity in `Twitter` data more often than not leads to inaccurate positioning of tweets, thus degrading the application's ability to spatially determine artists' fan bases. Therefore, employment of an approach taken by Cheng et al. [28] is suggested, making use of the authors' algorithm which aims at deriving users' locations solely from the content of their tweets. Combined with the positioning approach using `Twitter` data's location and timezone parameters presented by the work at hand, greater accuracy of placing tweets on a map might be achieved.

Employment of the `Twitter` Streaming API in order to mine actual geo-coordinates, as presented by Hauger and Schedl [26], constitutes an achievable improvement of the application as well. Given an extended time frame for development, the author of the work at hand might be able to acquire more `Twitter` data including geo-coordinates which in turn will result in more accurate visualization. However, due to dynamic properties exhibited by the tool at hand, only tweets issued in the last couple of days should be used in order to keep information up to date. `Twitter` data collected through the Streaming API is instantly ready for visualization though, eliminating long response times inherent in the current tool at hand and therefore extending its area of application.

Another aspect of future work denotes the evaluation process conducted in the previous section. So far, only future tour stops have been considered for evaluation in the paper at hand, resulting in good accuracy of the application's approach compared to a ground truth. However, it might be of interest to witness modifications in the accuracy value when additionally incorporating past tour stops into the evaluation process. Although the author expects the tool's correctness value to change only slightly (if anything), consideration of past and upcoming tour stops is nevertheless the next logical and necessary step in the evaluation process.

In addition to backend and evaluation refinements of the application, one GUI enhancement concerning the frontend is proposed as well. The basic idea is already implemented in the visualization service found on the website of the English rock band Placebo [8] and correspondingly explained in the related work section. Instead of merely visualizing tour stops on a map, an artist's travel route between individual shows is plotted

too using red arrows pointing into corresponding directions, giving the user an idea of how the respective tour is laid out. Once again employing the `Google Maps` API service for the application at hand, routes and distances between visualized markers can be displayed, thereby enhancing user experience of the tool. Furthermore, musicians' managements are potentially given an even better understanding of the planning of tours, thus cutting additional costs inherent to long-distance transport of music instruments and stage equipment.

References

1. http://www.google.com/maps (access: July 16, 2013)
2. http://www.twitter.com (access: July 16, 2013)
3. http://www.last.fm (access: July 16, 2013)
4. http://www.songkick.com (access: July 16, 2013)
5. http://en.wikipedia.org/wiki/A_Bigger_Bang_Tour (access: July 18, 2013)
6. http://en.wikipedia.org/wiki/List_of_countries_by_GDP_(nominal) (access: July 18, 2013)
7. http://www.statisticbrain.com/twitter-statistics (access: July 18, 2013)
8. http://www.placeboworld.co.uk/shows (access: July 22, 2013)
9. http://www.placeboworld.co.uk/news/welcome-new-placeboworld (access: July 22, 2013)
10. http://www.musicbrainz.org/doc/MusicBrainz_Identifier (access: July 24, 2013)
11. http://www.barebones.com/products/textwrangler (access: July 25, 2013)
12. http://en.wikipedia.org/wiki/PHP (access: July 25, 2013)
13. http://en.wikipedia.org/wiki/JavaScript (access: July 25, 2013)
14. http://www.frequency.at/lineup/days (access: July 30, 2013)
15. http://www.ofmonstersandmen.com (access: July 30, 2013)
16. http://www.enmoretheatre.com.au (access: July 30, 2013)
17. http://muse.mu/tour-dates,rockinrioriodejaneirobrazil_1807.htm (access: August 2, 2013)
18. http://en.wikipedia.org/wiki/List_of_the_most_common_U.S._place_names (access: August 2, 2013)
19. https://developers.google.com/maps/documentation/geocoding/#Limits (access: August 3, 2013)
20. http://arcticmonkeys.com/gigs.php (access: August 4, 2013)
21. Troncy, R., Malocha, B., Fialho , A. T. S. (2010). Linking Events with Media. In *Proceedings of the 6th International Conference on Semantic Systems*, Article No. 42, Graz, Austria. New York, NY, USA: Association for Computing Machinery.
22. Liu, X., Troncy, R., Huet, B. (2011). Using Social Media to Identify Events. In *Proceedings of the 3rd ACM SIGMM International Workshop on Social Media*, 3-8, Scottsdale, AZ, USA. New York, NY, USA: Association for Computing Machinery.
23. Bassoli, A., Brewer, J. (2011). frstyl: simplifying the process of promoting and discovering local live music. In *Proceedings of the 9th ACM SIGCHI Italian Chapter International Conference on Computer-Human Interaction: Facing Complexity*, 167-170, Alghero, Italy. New York, NY, USA: Association for Computing Machinery.

24. Castillo, C., Mendoza, M., Poblete, B. (2011). Information Credibility on Twitter. In *Proceedings of the 20th International Conference on World Wide Web*, 675-684, Hyderabad, India. New York, NY, USA: Association for Computing Machinery.
25. Cha, M., Haddadi, H., Benevenuto, F., Gummadi, K. P. (2010). Measuring User Influence in Twitter: The Million Follower Fallacy. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, 10-17, Washington, DC, USA. Menlo Park, CA, USA: Association for the Advancement of Artificial Intelligence.
26. Hauger, D., Schedl, M. (2012). Exploring Geospatial Music Listening Patterns in Microblog Data. In *Proceedings of the 10th International Workshop on Adaptive Multimedia Retrieval*, Copenhagen, Denmark.
27. Hiruta, S., Yonezawa, T., Jurmu, M., Tokuda, H. (2012). Detection, Classification and Visualization of Place-triggered Geotagged Tweets. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, 956-963, Pittsburgh, PA, USA. New York, NY, USA: Association for Computing Machinery.
28. Cheng, Z., Caverlee, J., Lee, K. (2010). You Are Where You Tweet: A Content-Based Approach to Geo-locating Twitter Users. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, 759-768, Toronto, ON, Canada. New York, NY, USA: Association for Computing Machinery.