Pythia: A Privacy-enhanced Personalized Contextual Suggestion System for Tourism

George Drosatos*,†, Pavlos S. Efraimidis*,‡, Avi Arampatzis*,‡, Giorgos Stamatelatos*,‡ and Ioannis N. Athanasiadis*,‡

*ATHENA, Research & Innovation Center, University Campus, 67100 Xanthi, Greece
†Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece
‡Electrical & Computer Engineering Dept., Democritus University of Thrace, 67100 Xanthi, Greece
E-mail: {gdrosato, pefraimi, avi, gstamat, iathan }@ee.duth.gr

***Abstract*—We present Pythia, a privacy-enhanced non-invasive contextual suggestion system for tourists, with important architectural innovations. The system offers high quality personalized recommendations, non-invasive operation and protection of user privacy. A key feature of Pythia is the exploitation of the vast amounts of personal data generated by smartphones to automatically build user profiles, and make contextual suggestions to tourists. More precisely, the system utilizes (sensitive) personal data, such as location traces, browsing history and web searches (query logs), to build a POI-based user profile. This profile is then used by a contextual suggestion engine for making POI recommendations to the user based on her current location. Strong privacy guarantees are achieved by placing both mechanisms at the user-side. As a proof of concept, we present a Pythia prototype which implements the aforementioned mechanisms as mobile applications for Android, as well as, web applications.**

***Keywords*-Privacy; Personalization; Contextual Suggestion; Personal Data; Non-Invasiveness; Tourism; Mobile Computing**

I. Introduction

Smartphones are ubiquitous devices much more than communication tools, as they combine features of a cell phone along with computing functionalities. As ubiquitous devices, they merge and even interfere with a person's everyday life. In this work, we focus on how a smartphone can be used to make touristic contextual suggestions to its owner. Whatever may attract a tourist can be considered a Point of Interest (POI), and in general, tourists are often overwhelmed by the large number and variety of POIs in the places they visit. Typically, they need to make their choices in short time and without being fully informed about their options regarding the available POIs. Assuming that a contemporary tourist owns a smartphone and is familiar with mobile applications (apps), mobile recommender systems can become valuable tools. In fact, mobile tourism services is currently a very active research topic; for a collection of related papers, see e.g. [1].

A recommender system for tourism, like any recommender system, requires some input data in order to make appropriate suggestions; in this case, which POIs the user should visit while staying in a certain area. A simple approach with regard to input data is to ask the tourist about her preferences, which includes information such as her needs, interests and constraints, enter those into some system, and then have the system correlate these preferences with POIs which have similar parameters [2]. Another option is to use additional input evaluations and ratings of other tourists with similar interests in a collaborative filtering fashion [3]. Additionally, travelers who are in close spatial and temporal proximity often share common travel interests or needs [4], [5]. A common requirement of all these approaches is that users have to enter *themselves* their personal information such as their interests, previous visits and ratings into some system. In this paper, we propose an approach for automatically building a POI-based user profile and show how this profile can be used as input to an contextual suggestion mechanism.

Another, possibly more important, requirement of existing recommender systems is that user profiles have to be stored and managed by the recommender service. User profiles, however, contain personal data, some of which may be considered *sensitive personal data* according to the Data Protection Directive 95/46/EC, which regulates the processing of personal data within the European Union, and similar may hold in other regions. This in turn, raises privacy concerns to potential users of such systems. Moreover, the privacy issues can be further aggravated when user profiles are combined among recommender systems in order to expand the data pool and to enable more intelligent recommendations [6]. The privacy concerns are even more serious in mobile recommender systems, due to the wide range of (sensitive) personal data (e.g. location data) that are available in mobile platforms and can be accessed by mobile apps and transparently get uploaded to remote servers. In fact, user awareness of threats against location and identity privacy aspects has been recognized as one of the greatest barriers to the adoption of context-aware services [7]. In the contextual suggestion system proposed in this work, user profiles are stored at the user-side. We will show that this architectural decision enables strong privacy guarantees for the users.

In this work, we present Pythia, a privacy-enhanced personalized contextual suggestion system for tourism. The main innovation of Pythia is that it combines a set of features which, to our knowledge, is unique in the field of mobile recommendation systems.

User-centric: Pythia is based on a user-centric architecture in which all user data are stored at the user-side. The contextual suggestion algorithms are also executed at the user-side. This choice offers the user a high level of control over the system.

Privacy-preserving: Thanks to the user-centric architecture and other precautions of the system, no personal data are disclosed to any party, including the recommender service. This assures strong privacy guarantees for the user. At the time it is simple and practical without dependencies on trusted third parties and cryptographic or data obfuscation techniques.

Rich user profiles: The system has access to a virtually unbounded - in size and types - set of personal data, since it has access to digital trace of the mobile devices of the user.

Non-invasive: The Pythia components operate non-invasively, without requiring user interaction (unless the user decides to initiate some task). The user profile which represents the user's interests is automatically built and updated.

II. Related Work

Mobile recommender systems. There is a rich literature on recommendation systems and mobile recommendation systems; see [8] and the references therein for general recommendation systems. An up-to-date survey for mobile recommendation systems for tourism is given in [9]. Additionally, a recent work is the myVisitPlanner^{GR} system [10], which recommends activities to visitors along with appropriate schedules to carry out these activities during a visit. The protection of privacy is also discussed, and even though users have to disclose some personal data to the myVisitPlanner^{GR} system, the amount of personal data disclosed is kept at a low level, measures are taken to avoid accidental data leaks, and the privacy policy of the application is clearly documented. Another recent work is the iGuide system [11] which aims at enabling a socially enriched mobile tourist guide service where its recommendations are enriched with multimedia content. However, as stated in [9], concrete mobile recommender system methodologies for protecting user anonymity and privacy are required. Those methodologies should guarantee the effectiveness and accuracy of recommendations without compromising the privacy of user profiles and sensitive contextual information, and this work contributes towards this direction.

Privacy-enhanced recommender systems. One approach to enhance privacy in recommendations systems is to apply appropriate cryptographic or algorithmic techniques. For example, this work [12] proposes the use of homomorphic encryption as a structural element for its system. Another system that uses randomized perturbation instead of encryption has been proposed in [13]. In mobile recommender systems for tourism, the authors in [14] propose the use of differential privacy methods to extract statistics about users' preferences for POIs, where the actual recommendations are generated by querying those statistics in order to formally enforce privacy. In our view, the user-centric architecture of Pythia combines a set of features which is unique with respect to related work; it is simple and practical since it does not rely on trusted third parties or heavy cryptographic protocols; it is accurate since it does not need to introduce noise into the data (to protect privacy); and it offers strong privacy guarantees since the personal data are not disclosed to any party, including the recommender service provider.

III. The Pythia System

Pythia is a new privacy-enhanced user-centric contextual suggestion system for tourism. The system exploits the user's digital trace to automatically generate and update a profile, which is safely kept at the user side. A contextual suggestion component, executed at the user's side, considers the profile and a location of interest, and generates POI suggestions.

An overview of the Pythia system architecture is shown in Figure 1. The system comprises mobile, conventional (desktop or server-side), and cloud components. A personal cloud storage serves as intermediate storage for the application components and a POI collection framework manages the available POI data of the system. For the user, the most essential part of the system is the mobile app, which consists of the following components:

(a) *Content Collector:* A non-invasive, background service which collects (sensitive) personal data of the digital trace generated mainly by the user's mobile device. There are several kinds of personal data that can be collected. In this work we focus on location traces, browsing history, and web searches. Optionally, the collected data can be uploaded in encrypted form to the personal cloud storage of the user.

(b) *POI-based Profiler:* A key component of Pythia, which uses the raw personal data of the user to build a profile of her interests. The POI-based profile is a list of rated POIs that represent the user's interests. The list and the ratings are automatically generated.

(c) *Contextual Suggestion Engine:* This component uses the POI-based profile of the user to generate a personal query, which is then used to retrieve the personalized POI suggestions for the user's area of interest.

The latter two components are replicated also as desktop applications, in case users have access to a desktop computer. An additional component of the desktop version is:

(d) *Content Retriever:* A tool that retrieves the encrypted personal data from the personal cloud storage of the user. The data can then be used by the desktop version of the POI-based Profiler.

Finally, there are two supporting components, the user's Personal Cloud Storage, and the POI Collection Framework.

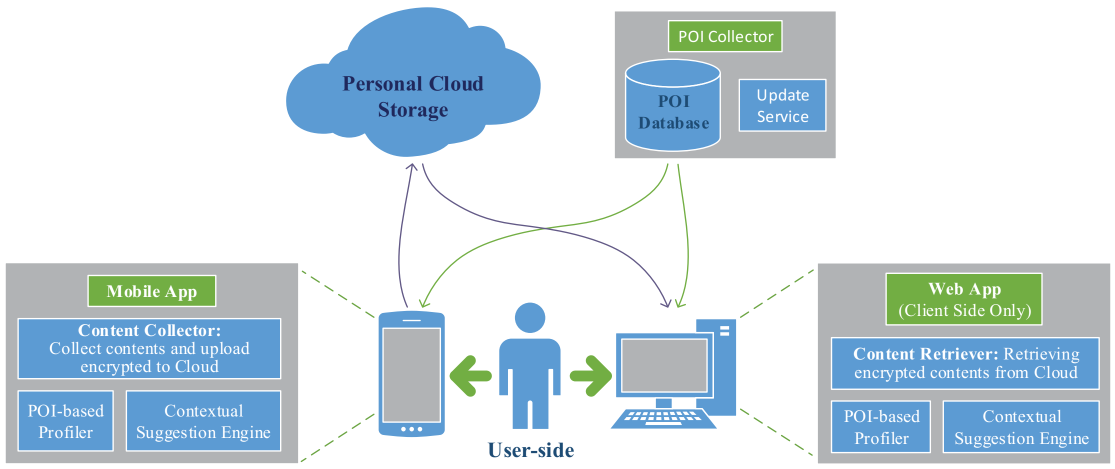

Personal Cloud Storage
POI Collector
POI Database
Update Service
Mobile App
Content Collector: Collect contents and upload encrypted to Cloud
POI-based Profiler
Contextual Suggestion Engine
User-side
Web App (Client Side Only)
Content Retriever: Retrieving encrypted contents from Cloud
POI-based Profiler
Contextual Suggestion Engine

Figure 1. Overview of the architecture of the Pythia system.

(e) *Personal Cloud Storage:* In principle, any popular cloud provider such as *Google Drive*, *Microsoft OneDrive*, *Dropbox*, or some user-controlled platform like *OwnCloud*[1] can be used. For simplicity, the current option is *Google Drive* since any Android device has by default access to it. This component is used for:
- *Aggregation of raw personal data from different devices:* When a user collects content from more than one mobile device, the personal cloud storage is the common storage space where each device uploads its raw personal data. To protect user privacy even from the cloud provider, the data uploaded to the cloud are encrypted (e.g. with AES[2]).
- *Flow of data between the application components:* The cloud storage is used by the desktop application to access the raw personal data generated on the mobile device(s).
- *Backup of the application data:* In case of data loss from some mobile device, the data stored on the personal cloud space is preserved.

(f) *POI Collection Framework:* A central component of the Pythia system, running at the server-side. The framework comprises a POI database, a service to automatically update its contents, and a Web API. The POI database maintains a collection of POIs from selected areas in several countries. The update service continuously populates and updates the database with data retrieved from two popular place search engines, *Google Places*[3] and *Foursquare*[4]. The POI database contains the POIs that are used by the POI-based Profiler and the Contextual Suggestion Engine.

[1]http://www.owncloud.org
[2]http://en.wikipedia.org/wiki/Advanced_Encryption_Standard
[3]https://developers.google.com/places/
[4]https://foursquare.com

IV. How Pythia Works

In this section, we describe how the Pythia system works, with emphasis on the algorithmic and computational aspects of its components. The Personal Data Collector and the POI Collection Framework are responsible to collect the data which is used by the POI-Profiler and the Contextual Suggestion engine. The Content Retriever and the Personal Cloud Storage support the data flow between the application components.

A. *Personal Data Collection*

As noted earlier, the Content Collector collects personal data of the digital trace generated by the everyday usage of the devices. This data is stored locally on each device. Currently, we focus on the following types of personal data.
Location traces: The Content Collector records the location of the mobile device periodically every t_ℓ (e.g. $t_\ell = 5$ min). The location is requested from the GPS provider of the device. In case the GPS provider fails to coordinate (for example, if the user is inside a building), the location is requested from the Network provider.[5] If both location providers fail to retrieve location information, the Content Collector temporarily reduces the frequency of location data requests to avoid unnecessary power consumption.
Browsing history: The browsing history of the user is also is collected with a content observer that is triggered when the user visits a web site with a mobile web browser. For each visited URL, a browsing history record is kept with the following data fields: title, url, number of visits, and time of last visit.
Web searches: Web searches are a special type of URLs and can reveal important information about the user's interests. The Content Collector detects and records the web search

[5]The Network provider determines location based on availability of cell tower and WiFi access points.

queries submitted by the user. For each web search, the query and the submission time are recorded.

B. POI Collection Framework

The POI Collection framework implements a Web API which is used by other Pythia components to access the POI database. POIs can be retrieved either one by one or as chunks corresponding to a specific county or area.

The data available for each POI are: title, geo-location (i.e. geographical coordinates of location), address, phone number, categories, several URLs (e.g. website, Foursquare URL, Google+ page, etc.), rating, total unique (physical) visitors and total visits (as measured by Foursquare), and a collection of terms describing the POI. The POI data is obtained from two popular place search engines, namely Foursquare and Google Places, with Foursquare as the main source of information. The update service continuously updates and extends the database with new information about POIs in the defined areas of interest.

C. POI-based Profiling

The raw data used by the POI-based Profiler consists mainly of a series of locations the user has been, plus the browsing and web search history. These data are processed in the following steps:

(i) First, the location traces of the user are used to extract significant places, which are defined as places where the user spends a significant amount of time and/or visits frequently. The extraction process is based on a *time-based location clustering* algorithm.

(ii) Then, the extracted significant places are matched (if possible) with existing POIs of the POI database. The matching is based on distance and popularity criteria of each particular significant place and the nearby POIs.

(iii) Next, the user's rating for each matched POI is estimated. This is also done automatically, using the user's number of visits and the average number of visits per user to this POI.

These steps are described in more detail, next.

Significant places extraction. The significant places for a user are extracted from her raw location traces with the *time-based cluster algorithm* of [15]. The algorithm processes one by one, in one pass, the stored locations, and clusters them along the time axis. Each new location measurement is compared with previous locations. Depending on how far the new location is moving away from previous locations, it is considered either part of the previous cluster or the start of a new cluster.

The default values for the algorithm parameters used in the prototype implementation of Pythia are 30 meters for the distance threshold (for locations measurements to be considered to belong to the same cluster) and 30 minutes for the minimum time duration threshold (for a cluster to be considered to be a significant place). For each significant place, the extraction process also estimates the number of visits, the duration of each visit, and the arrival date and time of each visit.

POI matching. After identifying significant places from the location traces of a user, these significant places are matched with POIs of the POI database. More precisely, given a significant place c, for each POI p_i within a distance of at most $d_{\max} = \max(\text{average accuracy}, 20\,\text{meters})$ from c, we calculate a score

$$M(p_i) = a \cdot \frac{1}{d(c, p_i)} + (1 - a) \cdot \pi(p_i)\,, \tag{1}$$

where $d(c, p_i)$ is the distance between c and p_i, and $\pi(p_i)$ is the popularity of p_i (e.g. the rating of the POI in Foursquare). A usable and effective value of the weighting coefficient a has to be experimentally estimated; its value does not only depend on the relative importance of distance and popularity, but also on the units used for measuring distance (e.g. meters, kilometers, miles, etc) and on the range of the popularity values $\pi(p_i)$. The significant place is matched to the POI that achieves the highest score.

Two comments are in order: (1) The initial significant places are locations important to the user, but not necessary all of them are POIs of general interest. Thus, some of them will not be matched with POIs in the POI database. Only significant places that are actually matched to POIs play a role in the POI-based profile of the user. (2) The POI matching procedure is executed at the user-side, by the application client running on the mobile device or the desktop of the user. The client application retrieves the POIs in the area of interest from the central POI database of the Pythia system. To avoid significant leaks about the locations the user is interested in, the client retrieves the related POIs in chunks that correspond either to predefined regions of counties or to predefined divisions of the geographical area (e.g. 30×30 Km squares).

POI rating. Finally, a rating $R(p_i)$ for each matched POI is calculated. This rating is an estimation of the user's interest for the POI. We use the rating scale of the Contextual Suggestion Track of TREC 2013, that is, each rating is an integer in {0,1,2,3,4}, where 0 means strongly uninterested, 2 means neutral, and 4 strongly interested. Given a user, first we calculate for each matched POI in her profile an index

$$f(p_i) = \log(m(p_i))/\mu(p_i)\ , \tag{2}$$

where $m(p_i)$ is the number of the user's visits to p_i, and $\mu(p_i)$ is the average number of visits per user to the specific POI. The average number can be calculated from public data from the place search engines. Let p_H and p_L be the POIs with the highest and lowest index respectively. Then, p_H is given a rating of 4, p_L a rating of 0, and each of the other POIs a rating proportional to its index value. The outcome of the POI matching and rating algorithms described is the POI-based profile of the user, i.e. a set of rated POIs.

Web searches and browsing history. The basic POI-based profile obtained from the location traces of the user can potentially be enhanced by exploiting the browsing history and the web search logs of the user. More precisely, we propose to use these personal data to adjust the POI ratings of the basic profile. For the adjustment, we apply an information retrieval approach. The terms of the web searches and the titles of the browsing history are used to generate a query which is then submitted to the collection of descriptions of the matched POIs. Depending on how high each POI is ranked, the score of the POI can be adjusted by at most ± 1. Understandably, this procedure may introduce additional noise into the POI ratings, and for this reason the adjustment is restricted to be at most 1.

D. Contextual Suggestion Algorithm

Given the POI-based profile of a user and a location of interest, the Pythia system returns a set of suggested POIs which are expected to be of interest to the user. The underlying algorithm that makes the POI selection is based on a Rocchio-like relevance feedback method [16], and its main steps are depicted in Figure 2. First, the user's POI-based profile is used to generate/train a text query. Then, the generated query is used to score and rank all candidate POIs using their textual representations. The proposed Rocchio-like model was evaluated in the Contextual Suggestion Track of TREC 2013 [17], where the corresponding group ranked with this algorithm at 2nd place among 15 research groups.

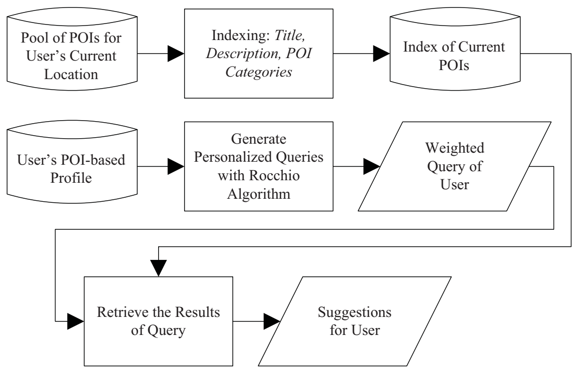

Pool of POIs for User's Current Location
Indexing: Title, Description, POI Categories
Index of Current POIs
User's POI-based Profile
Generate Personalized Queries with Rocchio Algorithm
Weighted Query of User
Retrieve the Results of Query
Suggestions for User

Figure 2. The Rocchio-like Contextual Suggestion model.

The model is implemented in three main steps:

Step 1: Indexing all candidate POIs. In the first step, a text document is built for each candidate POI. The document is a concatenation of the title, the description, and categories of the POI. The description consists of a collection of terms that describes the POI and it is taken by the POI's web page. The POI documents are indexed with Indri v5.5 (http://www.lemurproject.org), using the default settings and Krovetz stemming [18].

Step 2: Building a query in a Rocchio-like relevance feedback fashion. A personalized weighted query is generated from the POI-based profile of the user using a Rocchio-like relevance feedback method. This query represents the user's preferences. First, a text document is built this time for each POI in the POI-based profile of the user. Let N be the number of distinct terms appearing in these documents. Each POIs p_i of the POI-based profile is rated and can be used as a training POI. Let $P_i =< w_{i,1}, \cdots, w_{i,N} >$ be a weighted vector representing p_i, where $w_{i,j}$ is the weight of term j in P_i; we use tf-only logarithmic weighting: $w_{i,j} = log(1 + f_{i,j})$, where the $f_{i,j}$ is the frequency of term j in the textual representation of POI i. Then, the trained weighted query of the user is the vector

$$Q = \sum_{j=0}^{4} \left((j-2) \frac{1}{|S_j|} \sum_{i:p_i \in S_j} P_i \right), \qquad (3)$$

where the S_j is the set of all POIs in the POI-based profile of the user having a rating of $j \in [0, 4]$. In other words, we take the centroids per rating j, multiply them with the normalized rating $j - 2$ (so that the neutral rating's centroid, i.e. $j = 2$, does not contribute anything—it is zeroed), and then add the centroids in a Rocchio relevance feedback fashion. All the terms of the weighted query Q that have weight less or equal than zero are excluded from the query. The weight of every term is included in the query by using the Indri Query Language and has, e.g., the following form:

```
#weight( 3.0 museum 2.7 art ··· 0.1 nice )
```

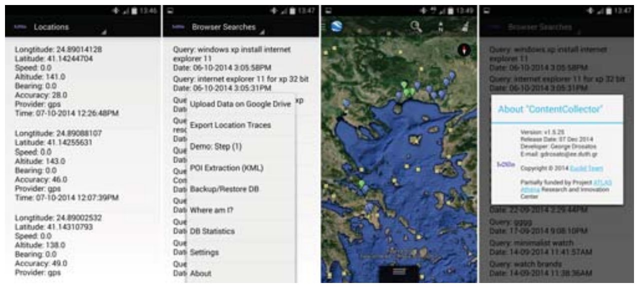
Step 3: Retrieving suggestions. The personalized weighted query Q of the user is submitted to the index generated in Step 1 for the POIs near the user's current location. The search engine is again Indri v5.5 with the default (LM) retrieval model. The results of the query, with a possible cutoff threshold (e.g. top-50 results), are the suggestions to the user for the specific location.

V. Prototype Implementation

To evaluate the feasibility, usability, efficiency, and effectiveness of our approach, we developed a prototype of Pythia that implements the proposed privacy-preserving user-centric architecture. In the current status of our implementation, the POI Collector and the Content Collector (Figure 3) are fully implemented, and the POI-based Profiler and the Contextual Suggestion Engine prototypes are only implemented as web applications (Figure 4). The components of the current prototype are implemented as a collection of standalone applications which communicate through the the personal Cloud storage of the user.

VI. Conclusion

The Pythia system is a privacy-enhanced non-invasive contextual suggestion system for tourists, with important architectural innovations. The main system components operate in the background without user interaction, and all


Locations

Longtitude: 24.89014128
Latitude: 41.14244704
Speed: 0.0
Altitude: 141.0
Bearing: 0.0
Accuracy: 28.0
Provider: gps
Time: 07-10-2014 12:26:48PM

Longtitude: 24.89088107
Latitude: 41.14255631
Speed: 0.0
Altitude: 143.0
Bearing: 0.0
Accuracy: 46.0
Provider: gps
Time: 07-10-2014 12:07:39PM

Longtitude: 24.89002532
Latitude: 41.14310793
Speed: 0.0
Altitude: 138.0
Bearing: 0.0
Accuracy: 49.0
Provider: gps

Browser Searches

Query: windows xp install internet explorer 11
Date: 06-10-2014 3:05:58PM
Query: internet explorer 11 for xp 32 bit
Date: 06-10-2014 3:05:31PM

About "ContentCollector"

Version: v1.5.25
Release Date: 07 Dec 2014
Developer: George Drosatos
E-mail: gdrosato@ee.duth.gr

Copyright © 2014 Euclid Team

Partially funded by Project ATLAS Athena Research and Innovation Center

Date: 22-09-2014 2:29:44PM
Query: gggg
Date: 17-09-2014 9:08:10PM
Query: minimalist watch
Date: 14-09-2014 11:41:57AM
Query: watch brands
Date: 14-09-2014 11:38:36AM

Figure 3. Android app screenshots.

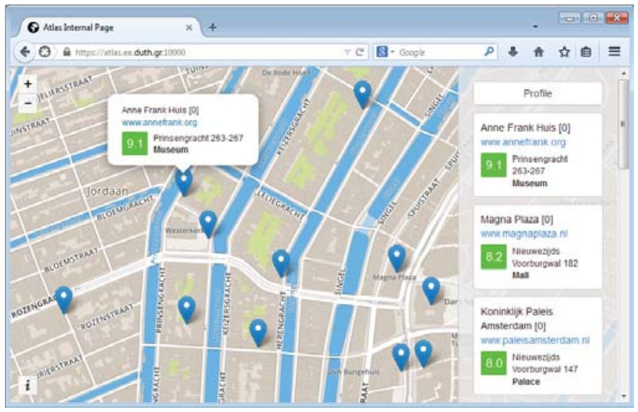


Figure 4. Contextual suggestion Web UI.

personal data of the user are kept on her own device. The system's architecture allows to use sensitive personal data for personalized suggestions of touristic attractions without violating the individuals' privacy. The resulting recommendations are context-based (i.e. related to current geographical place of the user), and the whole system operates non-invasively. The protection of user privacy is achieved by placing the profiling and recommendation procedures at the user-side, either on a mobile device or a desktop. Finally, we developed a prototype implementation of Pythia and presented preliminary results that confirm the feasibility of our approach.

The development of the Pythia system is continuing. Our current plans are to improve the automatic POI matching and POI rating algorithms, to upgrade the implementation of Pythia to a more complete and stable system, and to perform a more extensive evaluation of the complete system.

Acknowledgment

This work has been partially supported by project ATLAS (Advanced Tourism Planning System), GSRT/CO-OPERATION/11SYN-10-1730.

References

[1] Z. Xiang and I. Tussyadiah, *Information and Communication Technologies in Tourism 2014*. Springer, 2014.

[2] D. Gavalas and M. Kenteris, "A web-based pervasive recommendation system for mobile tourist guides," *Personal and Ubiquitous Computing*, vol. 15, no. 7, pp. 759–770, 2011.

[3] S. Stabb, H. Werther *et al.*, "Intelligent systems for tourism," *Intelligent Systems, IEEE*, vol. 17, no. 6, pp. 53–66, 2002.

[4] W.-S. Yang and S.-Y. Hwang, "iTravel: A recommender system in mobile peer-to-peer environment," *J. of Systems and Software*, vol. 86, no. 1, pp. 12 – 20, 2013.

[5] A. De Spindler, M. C. Norrie *et al.*, "Spatio-temporal proximity as a basis for collaborative filtering in mobile environments," in *CAISE Workshop on UMICS '06*, 2006.

[6] J. Zhan, C.-L. Hsieh *et al.*, "Privacy-preserving collaborative recommender systems," *IEEE Trans. on Systems, Man, and Cybernetics, Part C, 40(4)*, pp. 472–476, July 2010.

[7] L. Barkhuus and A. Dey, "Location-based services for mobile telephony: a study of user's privacy concerns," in *IFIP Interact '03*. IOS Press, 2003.

[8] J. Bobadilla, F. Ortega *et al.*, "Recommender systems survey," *Knowledge-Based Systems, 46*, pp. 109 – 132, 2013.

[9] D. Gavalas, C. Konstantopoulos *et al.*, "Mobile recommender systems in tourism," *J. of Network and Computer Applications*, vol. 39, no. 0, pp. 319 – 333, 2014.

[10] I. Refanidis, C. Emmanouilidis *et al.*, "myVisitPlanner^{GR}: Personalized itinerary planning system for tourism," in *AI, SETN '14, LNCS 8445*. Springer, 2014, pp. 615–629.

[11] S. Tsekeridou, V. Tsetsos *et al.*, "iGuide: Socially-enriched mobile tourist guide for unexplored sites," in *AI, SETN '14, LNCS 8445*. Springer, 2014, pp. 603–614.

[12] Z. Erkin, T. Veugen *et al.*, "Generating private recommendations efficiently using homomorphic encryption and data packing," *IEEE Trans. on Information Forensics and Security, 7(3)*, pp. 1053–1066, June 2012.

[13] H. Polat and W. Du, "Privacy-preserving collaborative filtering using randomized perturbation techniques," in *IEEE ICDM '03*, 2003, pp. 625–628.

[14] D. Riboni and C. Bettini, "Private context-aware recommendation of points of interest: An initial investigation," in *IEEE Workshop CoMoRea at PerCom '12)*, 2012, pp. 584–589.

[15] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello, "Extracting places from traces of locations," in *WMASH '04*. ACM, 2004, pp. 110–118.

[16] J. J. Rocchio, "Relevance feedback in information retrieval," in *The Smart retrieval system - experiments in automatic document processing*. Prentice Hall, 1971, pp. 313–323.

[17] G. Drosatos, G. Stamatelatos, A. Arampatzis, and P. S. Efraimidis, "DUTH at trec 2013 contextual suggestion track," in *TREC '13*, 2013.

[18] R. Krovetz, "Viewing morphology as an inference process," in *SIGIR '93*. ACM, 1993, pp. 191–202.