MYVISITPLANNER^{GR}: Personalized Itinerary Planning System for Tourism

Ioannis Refanidis[1], Christos Emmanouilidis[2], Ilias Sakellariou[1], Anastasios Alexiadis[1], Remous-Aris Koutsiamanis[2,3], Konstantinos Agnantis[1], Aimilia Tasidou[2,3], Fotios Kokkoras[4] and Pavlos S. Efraimidis[3]

[1] University of Macedonia, Greece
[2] ATHENA Research & Innovation Centre, Greece
[3] Democritus University of Thrace, Greece
[4] Technological Educational Institution of Thessaly, Greece

yrefanid@uom.gr, chrisem@ceti.gr, iliass@uom.gr, talex@java.uom.gr, akoutsia@ee.duth.gr, kagnadis@gmail.com, atasidou@gmail.com, fkokkoras@teilar.gr, pefraimi@ee.duth.gr

Abstract. This application paper presents MYVISITPLANNER^{GR}, an intelligent web-based system aiming at making recommendations that help visitors and residents of the region of Northern Greece to plan their leisure, cultural and other activities during their stay in this area. The system encompasses a rich ontology of activities, categorized across dimensions such as activity type, historical era, user profile and age group. Each activity is characterized by attributes describing its location, cost, availability and duration range. The system makes activity recommendations based on user-selected criteria, such as visit duration and timing, geographical areas of interest and visit profiling. The user edits the proposed list and the system creates a plan, taking into account temporal and geographical constraints imposed by the selected activities, as well as by other events in the user's calendar. The user may edit the proposed plan or request alternative plans. A recommendation engine employs non-intrusive machine learning techniques to dynamically infer and update the user's profile, concerning his preferences for both activities and resulting plans, while taking privacy concerns into account. The system is coupled with a module to semi-automatically feed its database with new activities in the area.

1 Overview

Undoubtedly the Web has revolutionized the way visitors obtain information regarding activities they can attend during their trip and how they form their itinerary. A number of services, such as Yahoo Trip Planner, Trip Advisor and Lonely Planet aim at assisting the discovery of such information and visit organization, however, they fail to provide more intelligent services such as personalized recommendations and automatic itinerary generation. This results to the user manually selecting activities

and forming plans, a process that might prove to be time consuming and error-prone.

This paper presents MYVISITPLANNER[GR1], a web-based recommendation and activity planning system, aiming at providing the visitor or the resident of Northern Greece with personalized plans concerning available activities. A broad range of activities that might be of his interest are considered, such as visiting museums, archaeological sites, churches and galleries, attending a concert or a performance, walking through interesting urban and rural paths, mountaineering, rafting or swimming, and many others. Personalization in the system has three aspects: preferences regarding the type of activities that are of the visitors' interest; preferences with respect to the scheduling of activities; and, finally, constraints imposed by other tasks already scheduled within the visitors' calendar. Preferences are described by user profiling, dynamically customized in a non-intrusive, user-specific manner, by monitoring user interaction with the system. Weighted activity types and soft scheduling constraints impact on the plan definition. Scheduling preferences concern the preferred time of day to schedule an activity and the tightness of the plan. Other profile aspects taken into account include age, gender and spoken languages. A set of default profiles has been created, to facilitate initialization of a visit plan. Finally, constraints imposed by other tasks are defined by integrating information from the user's calendar.

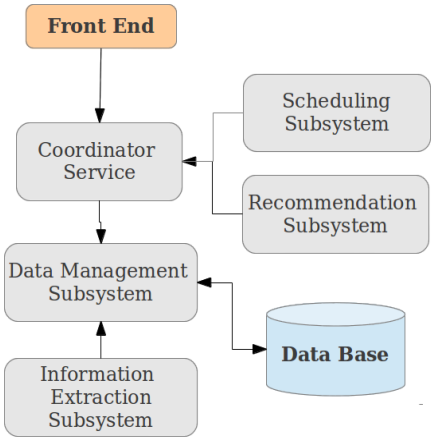

Fig.1. MYVISITPLANNER[GR] System Architecture

A typical use case consists of three steps: setting the visit framework; selecting activities; and, finally, forming the plan. In the first step the user defines the time period, the geographical areas and the user profile. In the second step the system recommends activities, taking into account the user's profile and the constraints imposed by the activities such as location, availability, estimated visit duration, as well as user-related constraints. The user can edit the recommended list, by removing and/or adding activities. Finally, in the third step, the system presents to the user an ordered list of distinct alternative plans for the selected activities.

[1]MYVISITPLANNER[GR] is currently available at http://mvp.gnomon.com.gr/

Since the success of MYVISITPLANNER^{GR} heavily depends on making valid recommendations, a semi-automatic process for information extraction from web sites feeds the database on a regular basis, besides information manually entered by the cultural activity providers. In all cases, a system administrator validates new entries. MYVISITPLANNER^{GR} adopts a service oriented architecture (Fig**.1**), with services providing the data management, recommendation and scheduling functionalities.

The rest of the paper is organized as follows: First related work concerning other trip management systems is briefly discussed, taking also into account their capacity to offer personalized recommendations and planning capabilities. The activity types ontology is presented next, followed by a description of the recommendation module. The scheduling engine of the system is then presented and the information extraction mechanisms are outlined. Next privacy concerns are highlighted and finally the paper concludes with a discussion of challenges for future work.

2 Related Work

There are several available web-based systems supporting trip organization. The motivation behind our work was Yahoo!'s Trip Planner (http://travel.yahoo.com/trip). After defining the trip dates as well as the geographical area covered by it, Yahoo!'s Trip Planner suggests activities and the user selects the ones to be included in the trip. For each activity, information is given about open hours and cost (in text form), as well as reviews. It is the user's responsibility to schedule manually each selected activity in time, with the risk of violating constraints imposed by the selected activities or by his other tasks.

Trip Advisor (http://www.tripadvisor.com.gr/), Lonely Planet (http://www.lonelyplanet.com/) and Travel Muse (http://www.travelmuse.com/) offer similar functionalities like Yahoo!'s Trip Planner. Other sites, like Expedia (http://www.expedia.com/) and Travelocity (http://www.travelocity.com/), focus on booking flights, hotels, cars and activities, thus suggesting only activities that have some cost. In all the aforementioned cases, there is no personalization concerning the suggested activities or user's preferences about the way the activities are placed in his calendar. Furthermore, there is no support for retrieving and updating the user's calendar and no automated scheduling functionality is offered.

plnnr (pronounced 'planner', http://plnnr.com/) is a recent web application offering similar functionality to MYVISITPLANNER^{GR}. By the time of writing this paper it covers 20 cities all over the world. After selecting the trip dates, the user can select one of four predefined themes (i.e., profiles), that is, 'family', 'outdoors', 'first time' and 'culture'. The user also selects one out of five levels of plan intensity, as well as a luxury level (e.g., hotel stars). Finally, the system creates a plan for each day of the visit, with the user being able to add or remove activities to/from the plan. The user can print the plan in the form of an agenda, similar to other web based trip planning applications. To the best of our knowledge, plnnr is the only system that offers some customization, in the form of predefined profiles used to suggest activities, as well as automated scheduling of the selected activities. Compared to MYVISITPLANNER^{GR}, it

lacks deep and broad activity ontology and a user profiling mechanism for personalization; it does not support a rich model of preferences over the way activities are scheduled in time; it does not encompass collaborative filtering for the recommendation module; and, finally, it does not integrate with the user's calendar.

There are many other systems that support automated scheduling of personal activities, most of them focusing on meeting scheduling. To the best of our knowledge, SELFPLANNER [15, 14], is the only one that focuses on scheduling personal individual activities, while encompassing a rich model of activities, with unary and binary constraints and preferences. It also exploits a rich scheduling engine based on deterministic and stochastic greedy search algorithms to schedule user's activities in time and space. Since SELFPLANNER is a general system, it could be used in principle to schedule tour activities as well. However, without a coupled information system providing data, mainly location and temporal availability of each activity, it would be impractical to use the system to create itineraries.

Other systems cope with the problem of automated meeting scheduling [7, 8, 17, 18]. RCal [19], an intelligent meeting scheduling agent, supports parsing and reasoning about semantically annotated schedules over the web[13].PTIME [5], developed under the CALO project [12], learns user's preferences about the way meetings are scheduled.

Tour planning and personalization is particularly useful for mobile guidance applications, which offer a rich, ubiquitous and interactive user experience, which may be personalized by exploiting context-adaptive features. The opportunities offered by adding such high-added value futures, such as planning/scheduling and information harvesting in a privacy preserving manner have not been well-explored yet [6].

3 The Ontology

MYVISITPLANNER^{GR} employs a dedicated ontology to describe activity types in a structured manner. The simplicity of the ontology was a design requirement, since it is intended to be directly handled by activity providers to input activity descriptions. Since these users will not generally be familiar with formal ontological descriptions, rather than defining a formal cultural activities ontology, the choice was to define a simple tour activities structure employ commonly perceivable terms. A representative subset of the employed activity ontology is presented in Fig. 2 and Fig. 3.

The main hierarchy contains the types of available activities, such as "Monument" or "Archaeological Site". The activity types are further analyzed at deeper hierarchy levels. An activity provider, thus, has the flexibility to either stay at the more abstract hierarchical level, or provide more accurate categorizations of provided activities. The rest of the hierarchies express auxiliary cross-cutting categorizations of the main activity type hierarchy and help mitigate a potential combinatorial explosion of activity types that would have otherwise been introduced by a categorization of very fine granularity. More specifically, the theme hierarchy allows the expression of the thematic category of the activity; the historical era (epoch) hierarchy enables a categorization according to the historical period of interest; and the target group hierarchy

assists in linking activities with different target groups.

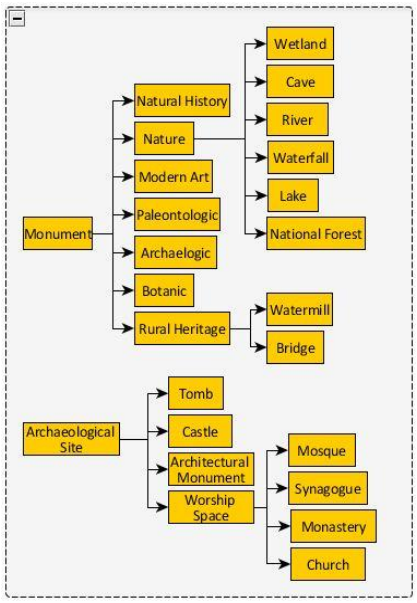

Fig.2. Activity Type Hierarchy

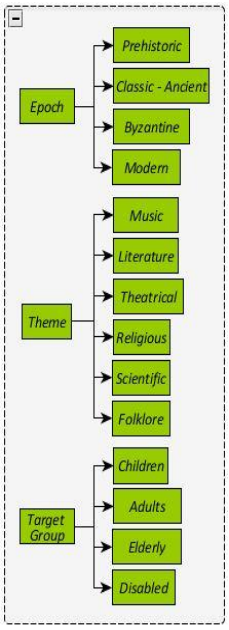

Fig.3. Auxiliary Hierarchies (Epoch, Theme and Target Group)

A key target of the defined ontology usage is for describing activities and user profile preferences. In the former case, the description allows sets of ontology entries to be specified. For example, in describing a castle on the shore of a lake, the set {Castle, Lake} can be specified. In the latter case, the description requires sets of weighted ontology entries. For example, in describing a user who is interested in caves, does not like castles and is indifferent to bridges, the set {(Cave, 1.0), (Castle, 0.0), (Bridge, 0.5)} can be specified. The simplicity in profiling is served by defining preferences over the activity type rather than the auxiliary hierarchies.

An evident advantage of the adopted ontological approach is that it combines simplicity towards the user with the ability to handle more complex associations by employing a composite similarity metric, to achieve improved performance in the recommendation results.

4 The Recommendation Subsystem

The recommendation subsystem in MYVISITPLANNER^{GR} assists the users in selecting the set of activities they wish to engage with during their trip (Fig.**4**). It is implemented as a hybrid collaborative filtering recommendation system [16]. It comprises two independent recommendation engines whose output is fed into a fusion function in

order to derive a final ordered list of activities (Fig.5). Each recommendation engine displays advantages and weaknesses in different cases. The hybrid approach builds on the individual strengths of the two engines to provide recommendations of improved quality, thus offering relevant recommendations, even in the case that insufficient user interaction data is available, while it can properly exploit such data, if available in sufficient quantity.

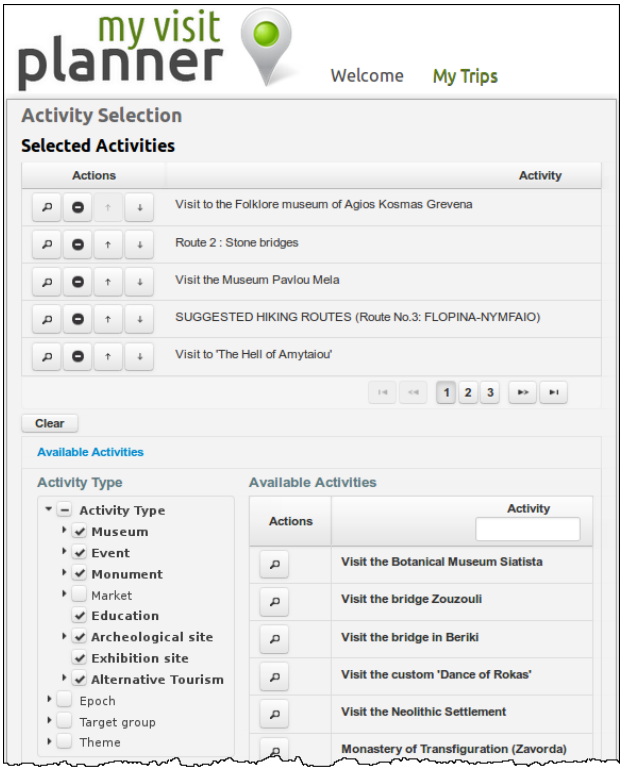

Fig.4. Selecting Activities

The first engine performs recommendation by suggesting user activities which are similar to the activities that have already been rated by the same user. The similarity of the activities is calculated via the Hausdorff distance (Eq. 1) between the ontological description of each activity, where the description is represented as a non-empty set of tree nodes taken from the activity type ontology. This distance expresses the greatest of all distances of a given activity from an activity type, described by a set of activity types to the closest activity type of the other activity. This metric has been selected as it effectively expresses the maximum dissimilarity between two activities, while having low computational requirements.

$$\mathrm{d}_H(A,B) = \max\left\{\sup_{a\in A}\inf_{b\in B}\mathrm{d}(a,b), \sup_{b\in B}\inf_{a\in A}\mathrm{d}(a,b)\right\} \quad (1)$$

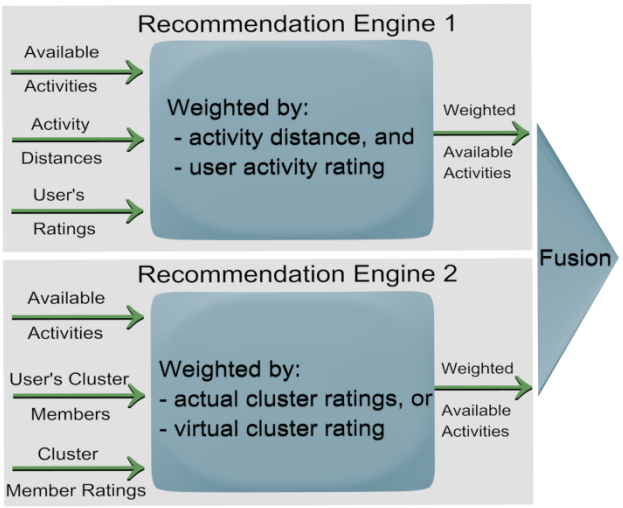

Fig.5. Hybrid Recommendation Subsystem

The distance between the individual activity types, denoted d(*a*, *b*) above, is equal to the length of the shortest path between them, when the activity types are taken as tree nodes in the hierarchical ontology. This engine takes advantage of the ontological information available for each activity, as well as the user's ratings for activities. Initially, the available activities are collected, consisting of all the activities which conform to the trip's time and location restrictions and the user's language restrictions. Then, the user's past activity ratings are fetched. For each of the available activities, the most similar set of rated-by-the-user activities is estimated. Each of the available activities' recommendation weight is calculated as a function of the Hausdorff distance between itself and the most similar rated activities and the mean rating of the rated activities. One advantage of this approach for generating recommendations is that ratings for activities are not required from other users, since only the user's own ratings are used. Another advantage is that a large part of the calculations can be precomputed off-line, since the activity descriptions change infrequently and as a result the distance between the activities remains unchanged. The disadvantages are that the user needs to provide ratings for some activities and that the other users' ratings are not taken advantage of. The former can be improved by deducing ratings from a user profile, albeit with somewhat limited accuracy. The latter is addressed by the second recommendation engine.

The second engine performs a variation of collaborative filtering recommendation. It suggests user activities by clustering users via top-down clustering and suggesting activities rated by other cluster members to members of the cluster. The similarity of the users is calculated via the distance between the ontological description of the user profile preferences and the similarities in age, gender, spoken languages and scheduling preferences. This engine takes advantage of the ontological information available for each user profile as well as the activity ratings of other users. As before, the set of

available activities is collected. Afterwards, the user's cluster is employed as a proxy for the user's ratings. For each available activity, if the activity has been rated by one or more members of the cluster, the activity's recommendation weight is assigned as the mean of the other members' ratings. If an activity has not been rated by any of the cluster members, the cluster's aggregate preferences are used to rate the activity, behaving as a virtual cluster-average user, but weighted with a factor signifying the diminished confidence in this approach. Among the advantages of the second engine are the exploitation of other users' ratings and the fact that a large part of the calculations, but not all, can also be pre-computed as clusters should be relatively stable and the cluster's aggregate preferences need not be frequently updated. Additionally, this engine also takes advantage of user profile preferences, which are updated from their initial values using machine learning techniques on the user provided feedback. The most important, though, is that the prior availability of user ratings is not a prerequisite for the system to make recommendations. The main disadvantage is the increased computational load, given the need to perform user clustering and that users need to belong to a cluster. However, this is not a major concern, since user clusters are formed and adjusted off-line, by periodically recalculating the clusters, while the prior definition of default representative user profiles enables usage by new users.

In the final merging stage the outputs of each of the two engines are combined. Each engine produces an independent list of (Activity, Weight) tuples. The merging function expresses the confidence in each engine by examining the richness of the information processed by each engine, such as user profile preferences generality, ratings, cluster size, cluster virtual profile preference generality, and weighs the two lists accordingly. Finally, the list is returned ordered from the most to the least recommended activity. Some parts of the user model are also used in an auxiliary manner to filter recommended activities out before inputting them into the recommendation engines. Age will filter age-inappropriate activities and spoken languages will filter out activities performed in unfamiliar languages. Scheduling preferences are forwarded to the scheduling engine.

One of the problems many systems with explicit user profile preferences have is the lack of user engagement in defining their preferences. Therefore, user profile preferences tend to be generic, neither strongly preferring nor strongly disliking anything. A remedy to this adopted by the present approach is to perform non-intrusive learning of these preferences by logging user choices during system usage, such as selecting, deselecting and viewing activities as a proxy for actual ratings. Obviously, direct user feedback in the form of plan and activity ratings is considered more significant, therefore the information gleamed in this manner is appropriately weighted such that the low confidence in these measurements is appropriately represented.

The recommendation subsystem executes the off-line calculations using the Apache Mahout[2] machine learning library on the Apache Hadoop[3] MapReduce framework.

[2] http://mahout.apache.org/
[3] http://hadoop.apache.org/

5 The Scheduling Engine

MYVISITPLANNER^{GR} exploits the planning engine of SELFPLANNER[14, 15]. This gives advantage to users of the latter system, since rescheduling of their non-cultural activities is possible, provided that these activities have been added to their calendars through SELFPLANNER; otherwise, activities manually inserted into a user's calendar are never rescheduled in order to accommodate new activities originated by MYVISITPLANNER^{GR}.

There are many types of cultural activities, from a scheduling point of view. An activity may have a fixed time and location. For example, a one-time concert may be of this type. Most activities however, e.g. a visit to a museum, are flexible, in the sense that the user is able to select when to perform them, within some specified time window. Similarly, some activities (e.g., concert) have a fixed duration, whereas others (e.g., museum visit) have a variable duration, depended on the user's profile.

Most activities have a specific location, however there might exist activities that are offered in several locations, like, e.g., watching a movie in any of the cinemas in the area. Furthermore, there are activities that have a different starting and ending location; for example, walking through the city does not require necessarily returning back to the starting point on foot, before performing any other activity. Locations are taken into account by the scheduling engine, in order to ensure that there is enough time for the visitor to move from the location of each activity to the location of the next one in his plan.

Bundles of activities are also supported. A bundle encompasses many elementary activities that are usually offered in reduced price as a bundle than when bought individually. Activities of a bundle may have ordering constraints among them.

Defining the temporal domain of an activity can be a laborious task for the cultural activity provider. MYVISITPLANNER^{GR} supports a structured and, at the same time, intuitive way to define temporal domains, based on an ordered list of statements concerning periods when the activity is provided or not [2]. Each statement has priority over the previous ones. For example, the following statements:

```
Every MoTuWeThuFri 09:00 to 21:00
Every Sat 10:00 to 18:00
Every Sun 10:00 to 17:00
Except every December 25th
```

define that an activity is offered 09:00 to 21:00 from Monday to Friday, 10:00 to 18:00 the Saturdays, 10:00 to 17:00 the Sundays, but is not offered the Christmas day.

A rich model of constraints and preferences is supported. Each activity is characterized by a wishfulness for the user. Furthermore, the user can express his preferences over the activity's temporal domain, that is, when he prefers the activity to be scheduled. Although the scheduling engine supports arbitrary preferences over the temporal domain, MYVISITPLANNER^{GR} offers only a limited set of options to the user, such as scheduling the activity in the morning or in the evening of any day. Binary preferences are supported as well. The user can express that he prefers two activities to be scheduled temporarily close or away to each other. Furthermore, the user can

state his preferences not over specific activities but over activity classes in the ontology. In that case, unary and binary preferences are applied to single activities and to pairs of activities respectively, unless they are overwritten by specific preferences.

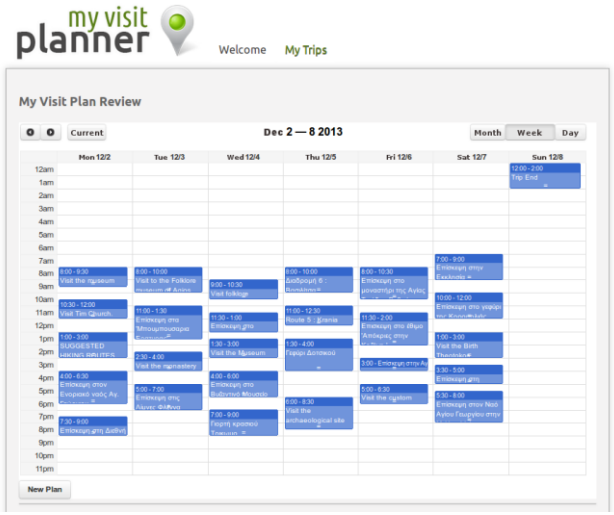

Fig.6. An automatically generated visit plan

In order to solve the scheduling problem, the scheduling subsystem calls the scheduling engine of SELFPLANNER, which is offered as an external TCP/IP server. The scheduling engine works in two phases: In the first phase it finds a good enough solution using Squeaky Wheel Optimization (SWO) [9], whereas in the second phase it employs Simulated Annealing, using SWO's solution as the starting state, in order to rich the nearest local optimum, defined across an extensive set of local transformations [3, 1]. After obtaining the first schedule, the user can ask for alternative schedules (Fig.6). In that case, the scheduling engine attempts again to solve the problem, while trying simultaneously to maximize a metric of the distance between the already found plans and the new one [4].

The scheduling engine supports more features than these exploited by the current version of MYVISITPLANNER^{GR}. Interruptible are the activities that can be accomplished in parts. For example, writing a paper is an interruptible activity. Having collected data of real cultural activities, we did not encounter any one requiring interruptible execution; hence there is no need to support them. Similarly, concurrent are the activities that can be executed concurrently with others, that is, they do not require the user's full attention. For example, attending a teleconference while working on a presentation might be possible. Although one could imagine cases when a cultural activity could be accomplished concurrently with others, we do not offer this option to MYVISITPLANNER^{GR} users. So, all cultural activities are considered as requiring the user's full attention; hence no concurrency is possible for them. However, for users of both MYVISITPLANNER^{GR} and SELFPLANNER systems, interruptibility and concurrency are important, since each time MYVISITPLANNER^{GR} is asked to produce a plan, all user's activities (from both systems) are taken into account.

6 Information Extraction from Semi-structured Data

Being a data intensive application, MYVISITPLANNER^{GR} requires a constant feed of fresh information regarding cultural events. To handle this requirement the system uses DEiXTo [10], a web content extraction suite that includes a GUI application for designing extraction rules (wrappers) and a command line executor that applies these rules to target URLs and stores the retrieved content into a database. The exact role of DEiXTo is threefold: a) extract classified-at-the-source cultural events, b) extract non-classified-at-the-source events, and c) detect new sources.

6.1 Extracting Classified Events

This task is based on the availability of local information sites that post cultural events in a classified manner, that is, they have their content organized in categories such as theater, music, etc. Additionally, these sites are built with modern content management systems and, as a result, they are excellent targets for extraction tasks. This is due to their web pages being template based, thus, one can easily detect HTML patterns reappearing in every event page and design accurate extraction rules based on those patterns. These sites typically organize the posted events in a master-detail fashion, where a master page includes a list of links to individual pages presenting the details of a single event. As a result there are also master and detail extraction rules, usually one pair for every event category of interest, in every site. DEiXTo uses a greedy (first occurrence matching), tree-matching algorithm which is described in detail in [10]. It matches the tree pattern of the extraction rule against the DOM tree of the page under consideration. The system works as described in the following paragraphs.

Master wrappers are executed periodically and extract URLs of pages containing cultural event descriptions. These URLs are the targets of the detail wrappers that extract the title, the body and the category of the event. The reader should recall that the category of the event is already known by design. The body text of the event is stored without any modifications and later is parsed with regular expressions and heuristic based techniques for metadata related to the event (location, time, cost, etc). The complete metadata set extracted for an event is finally presented to a human expert (along with the original page) who ensures that the correct information will be headed to the database.

Duplicate entries are currently detected and removed, based on the URL of the detail page. A similarity measure over the title and possibly over the body text is under consideration, since it is possible to have the same event posted in two or more different sites.

Finally, the extracted body text is cleared up from junk words, is passed to a Greek stemmer and the result is stored to serve as train instance for the classifier.

Currently, there are 12 sites monitored with a total of 54 extraction rules.

6.2 Extracting non-Classified Events

This case is similar to the previous except that the class of the extracted events is not known because the target site does not provide such event separation. This introduces one extra step in the metadata extraction procedure: the event should be classified. This is done using the stemmed body text (as described at the end of Section 6.1) and the classifier of the system. The result is verified by a human operator.

6.3 Detecting New Sources

The web is constantly changing as new technologies and services emerge. This is more intensive in the Greek web in which the transition to second or third generation sites is still in progress. As a result, MYVISITPLANNER^{GR} requires a way to detect new potential sources of cultural events.

There are currently two subsystems for new source detection. The first one queries the Google search engine with well-designed queries regarding specific cultural events in the geographical region of interest. The first ten unseen results are extracted using DEiXTo, their URLs are visited and their content is stemmed and classified as relevant cultural event or not. Relevant pages are checked by a human expert to see if they probably belong to a new site that should be wrapped properly with extraction rules and added to the list of the sources that provide classified events.

The second subsystem for new source detection is a crawler that aims at supporting the human exert mentioned earlier, in the task of detecting new sites that can serve as sources of cultural events. The crawler starts from the domain root address of pages detected using the Google search methodology and classified as relevant. It then crawls the target site at a certain depth and classifies the pages visited with the help of the body text extractor, the Greek stemmer and the classifier. If the percentage of the related pages of the crawled site is above a certain threshold, the site is considered interesting and is forwarded to the human expert for further examination.

7 Privacy Concerns

Storage of large amounts of data concerning user interests, travels, preferences and behaviors is a significant problem for both the user and the service provider who stores this data. The users risk having their private and potentially sensitive data mis-used. The service provider incentivizes more attacks against itself since more data are to be gained by unlawfully acquiring it and is also potentially liable for any data theft. At the same time, the recommendation subsystem requires the availability of large amounts of data to be able to function. We have attempted to reach a trade-off which allows the recommendation subsystem to deliver its intended functionality effectively, while at the same time increasing the users' privacy protection and diminishing the potential for large-scale data exfiltration. The penalty for this decision lies in in-creased implementation complexity, higher computational overheads and optionally, shifting some of the privacy protection burden to the users.

To enhance data protection, apart from the obvious security measures (e.g. access

control, logging, auditing), user data which is deemed sensitive is kept in encrypted form in the database. The data is transparently decrypted whenever the user logs into the system, and is kept decrypted for the duration of the user's session and then re-encrypted automatically. The data in the database is encrypted using the symmetric cipher. The symmetric key is itself encrypted using another cipher, using the KEK (Key Encryption Key) scheme [11], to allow changing user encryption keys without needing to decrypt the data and re-encrypt with the new key. The data which is considered sensitive and thus protected by the privacy mechanism in MYVISITPLANNER^{GR} is shown in Table 1 against the main processes where it is accessed and the entities that need access to the data. At this stage the system allows access to the user data to all entities, when the user is logged in. An additional protective measure could be to limit the access of each entity to the data needed for the processes they perform.

Table 1. Data usage in MYVISITPLANNER^{GR} processes

Entity	**User**	**Recommendation**			**Scheduler**
Scope **Data Item**	**Profile Editing (UI)**	**Activity Similarity Based Recommendation**	**User Clustering**	**User Cluster Based Recommendation**	**Scheduling**
Demographic Data	■		■		
Activity Type Preferences (in User Profile)	■		■		
System Preferences (in User Profile)	■				■
Detailed User Interaction Log	■				
Activity Ratings	■	■	■		

8 Conclusions

This paper presented MYVISITPLANNER^{GR}, an ongoing work aiming at helping visitors and residents of the Northern Greece area to include cultural activities, such as visiting museums churches and archaeological sites, attending performances or doing outdoor activities (walking, swimming, climbing, etc.), in their calendars. In order to schedule the activities, the system takes into account user preferences concerning the types of the activities and the way they are scheduled, as well as constraints imposed by the selected activities and the user's other commitments. A search engine employing greedy search followed by stochastic local search is employed to produce plans, while alternative plans with noticeable differences to the already suggested ones are provided, upon a user's request. The system is supported by a hybrid recommendation engine providing personalized activities recommendations, and by a semi-automated information extraction module to feed the system's database with fresh data. MYVISITPLANNER^{GR} is now entering the deployment and evaluation phases.

Acknowledgements. The reported work is financially **supported by** the General Secretariat for Research & Technology (Grant 09SYN-62-129). The collaboration with all project partners is gratefully acknowledged.

ΕΥΡΩΠΑΪΚΗ ΕΝΩΣΗ
ΕΥΡΩΠΑΪΚΟ ΤΑΜΕΙΟ
ΠΕΡΙΦΕΡΕΙΑΚΗΣ ΑΝΑΠΤΥΞΗΣ
ΑΝΤΑΓΩΝΙΣΤΙΚΟΤΗΤΑ ΕΠΙΧΕΙΡΗΜΑΤΙΚΟΤΗΤΑ
ΕΠΑΝ ΙΙ
η περιφέρεια στο επίκεντρο της ανάπτυξης
ΕΣΠΑ 2007-2013
Υπουργείο Παιδείας και Θρησκευμάτων
ΓΓΕΤ – ΕΥΔΕ-ΕΤΑΚ
Ε. Π. Ανταγωνιστικότητα και Επιχειρηματικότητα (ΕΠΑΝ ΙΙ), ΠΕΠ Μακεδονίας – Θράκης, ΠΕΠ Κρήτης και Νήσων Αιγαίου, ΠΕΠ Θεσσαλίας – Στερεάς Ελλάδας – Ηπείρου, ΠΕΠ Αττικής

References.

1. Alexiadis, A. and Refanidis, I. 2013. Post-Optimizing Individual Activity Plans through Local Search. ICAPS 2013 Workshop on Constraint Satisfaction Techniques for Planning and Scheduling Problems (COPLAS). Rome.
2. Alexiadis, A., and Refanidis, I. 2009. Defining a Task's Temporal Domain for Intelligent Calendar Applications.5th IFIP Conference on Artificial Intelligence Applications & Innovations (AIAI), pp. 399-406, Springer. Thessaloniki, Greece.
3. Alexiadis, A., and Refanidis, I. 2012. Meeting the Objectives of Personal Activity Scheduling through Post-Optimization. First International Workshop on Search Strategies and Non-standard Objectives (SSNOWorkshop'12), CPAIOR, Nantes, France.
4. Alexiadis, A., and Refanidis, I., 2013. Generating Alternative Plans for Scheduling Personal Activities. ICAPS 2013 Workshop on Scheduling and Planning Applications (SPARK). Rome.
5. Berry, P., Conley, K., Gervasio, M., Peinter, B., Uribe, T. and Yorke-Smith, N. 2006. Deploying a Personalized Time Management Agent. In 5th Inter. Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS-06) Industrial Track, pp. 1564-1571.
6. Emmanouilidis, C., Koutsiamanis, R.-A., and Tasidou, A., 2013. Mobile Guides: Taxonomy of Architectures, Context Awareness, Technologies and Applications, Journal of Network and Computer Applications, 36(1): 103-125
7. Garrido, L. and Sycara, K. 1995. Multi-agent meeting scheduling: Preliminary experimental results. In 1st International Conference on Multi-Agent Systems (ICMAS).
8. Jennings, N.R. and Jackson, A.J. 1995. Agent based meeting scheduling: A design and implementation, IEE Electronic Letters, 31(5):350-352.
9. Joslin, D.E., and Clements, D.P. 1999. "Squeaky Wheel" Optimization. *Journal of Artificial Intelligence Research*, 10: 375-397.
10. Kokkoras, F., Ntonas K., and Bassiliades, N. 2013. DEiXTo: A web data extraction suite. In proc. of the 6th Balkan Conference in Informatics, Thessaloniki, Greece, pp. 9-12.
11. Landrock, P. 2005. "Key Encryption Key." In *Encyclopedia of Cryptography and Security*, pp. 326-327. Springer US.
12. Myers, K. 2006. Building an Intelligent Personal Assistant. AAAI Invited Talk.
13. Payne, T.R., Singh, R., and Sycara, K. 2002. Calendar Agents on the Semantic Web. IEEE Intelligent Systems, 17(3):84-86.
14. Refanidis, I., and Alexiadis, A. 2011. Deployment and Evaluation of SELFPLANNER, an Automated Individual Task Management System. *Computational Intelligence*, 27(1), 41-59.
15. Refanidis, I., and Yorke-Smith, N. 2010. A Constraint Based Programming Approach to Scheduling an Individual's Activities. *ACM Transactions on Intelligent Systems and Technologies*, vol. 1 (2).
16. Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B. 2010. Recommender Systems Handbook. New York: Springer-Verlag
17. Sen, S. and Durfee, E.H. 1994.On the design of an adaptive meeting scheduler. 10th International Conference on Artificial Intelligence for Applications, pp. 40-46.
18. Sen, S. and Durfee, E.H. 1998. A formal study of distributed meeting scheduling. Group Decision and Negotiation, 7:265-289.
19. Singh, R. 2003. RCal: An Autonomous Agent for Intelligent Distributed Meeting Scheduling. Technical Report CMU-RI-TR-03-46, Robotics Institute, Carnegie Mellon University.