Data Personalization: a Taxonomy of User Profiles Knowledge and a Profile Management Tool

Mokrane Bouzeghoub and Dimitre Kostadinov

Laboratoire PRiSM, Université de Versailles

45, avenue des Etats-Unis, 78035 Versailles
{Mokrane.Bouzeghoub,Dimitre.Kostadinov}@prism.uvsq.fr

ABSTRACT. The goal of data personalization is to facilitate the expression of the need of a particular user and to enable him to obtain relevant information when he accesses an information system. The relevance of the information is defined by a set of criteria and preferences specific to each user or community of users. The data describing the users' interests and preferences is often gathered in the form of profiles. The content of a profile varies according to the target application and to its context (databases, information retrieval, digital libraries, multimedia applications). This paper elaborates a taxonomy of the most important knowledge composing a profile and proposes a generic model that can be instantiated and adapted for each specific application. The model is supported by a profile management tool (PMT) which also allows matching of profiles, aggregation of profiles and knowledge derivation from profiles. The last section of the paper lists some research challenges related to data personalization as identified in the APMD project initiated within the French Research Program on Data Masses[1].

1. Introduction

The access to relevant information, adapted to the needs and the context of the user is a challenge in an Internet or GRID environment, characterized by a proliferation of heterogeneous resources (structured data, textual documents, software components, images), leading to huge volumes of data. As these volumes as well as the diversify of data increase, information retrieval systems (Web search engines, DBMS, Digital Libraries Systems, etc.) deliver massive results in response to the user requests, thus generating an informational overload in which it is often difficult for users to distinguish relevant information from secondary information or even from the noise. The definition of user (or of communities of users) profiles makes it possible to address this problem by providing a personalized information access model.

[1] This work has been done with the support of French Ministry of Research under the ACI program for Data Masses (Project number MD33/APMD).

A profile is defined by a set of attributes, possibly organized into abstract entities, whose values can be user-defined or dynamically derived from user behavior. A profile is supposed to characterize user domain of interest and all his specific features that help the information system to deliver the most relevant data in the right form at the right place and the right moment. These attributes are generally ranked and organized as a preference model which will serve to drive query compilation, query execution and data delivery. Preference expressions may be of several kinds: introducing partial or total orders within data selection predicates or keywords, putting emphasis on some attributes or predicates by assigning relative weights, imposing the subset of predicates that should be fully or partially satisfied (top K predicates), discriminating between query results (top K results), adapting data delivery to the context (interaction media, spatio-temporal position), etc.

We make a clear distinction between profiles and queries: a profile is a user model which specifies the user domain of interest and the most general preferences that distinguish this user from the others, while a query is an on-demand user need which is evaluated with respect to a certain profile. All queries issued by the same user are evaluated with respect to his specific profile. The same query issued by different users may have different results as it is evaluated using different profiles. Profiles and queries should be considered as orthogonal concepts, although in some information retrieval systems both concepts are superimposed.

Data personalization has been addressed in different technologies such as information retrieval systems, database systems and human-computer interaction systems. In information retrieval systems, the user is fully involved in the query evaluation which is conducted as a stepwise refinement process where the user can decide at each step which data he likes and which data he dislikes. The personalization is then considered as a machine learning process based on user feedback. In database systems, it is not usual to involve the user in the data retrieval process: a database query contains generally all the necessary criteria to a selection of relevant data. Data personalization is thus considered under the viewpoint of query language extensions (generally SQL) and query rewriting process using user profiles. In human-computer interaction, user profiles generally define user expertise with respect to the application domain in order to provide them with appropriate interfaces and dialogues. Data delivery modalities, graphical metaphors and level of expertise of the dialogue constitute the main personalization issues in this context.

The goal of this paper is twofold: first, we propose a taxonomy of knowledge which constitutes a user profile and show how to use this taxonomy as a basis to a generic profile model; second, we list the main operations that can be applied on this model, and we can support profile definition and instanciation. Finally, we summarize the APMD joint research project with its main research directions.

2. User Profiles through examples

This section aims to illustrate, through a few examples, different kinds of profiles as well as their usage in information access.

The profile content varies with technologies and applications. In human-computer interaction, for example, a profile may contain knowledge which allows a given system to adapt its layouts with respect to the media used (e.g. PC, PDA, telephone screen). Many of the Web browsers and mailers allow to specify for each user profile his preferences in terms of aspects (colours, fonts, menu bars), delivery (length of messages or attached files, time interval between two connections), resources (memory cache, folders and applications to use), personal data (name, age, gender, languages, professional activity), etc. Based on this knowledge, the browser adapts its interface and interaction with each user. The following three examples illustrate different kinds of profiles as well as different usages of these profiles.

Example 1

In information retrieval systems, the user profile is often described by a set of possibly weighted keywords. These keywords are matched against those of indexed documents [11, 12, 7, 6], Weights are refined by successive user feedbacks given for the delivered documents or derived from the type of actions made on these documents (download, print, email), the time spent by a user on each document or the number of clicks on each link. CASPER project [3] is an example of such approach. Example 1 shows a typical profile derived by CASPER for a user interested in job announcements. Based on this derived profile, the information retrieval system uses a set of decision rules to filter jobs in further queries, considering for example that job56 and job45 are relevant to the user regarding action types made on these jobs (apply to the job and email the job announcement to a friend) while job5 is less important as it was only read.

Example 1:

Job Type	Action Type	Number of Clicks	Reading Time
job5	Read	1	234
job56	Apply	2	186
job45	email to a friend	1	54

These two announcements constitute interesting prototypes that will be used in a case-based reasoning approach to determine whether other new announcements are relevant or not for this user.

Example 2

Another profile example is given by [4]; it allows to define utility functions over a domain of interest. Two complementary clauses, DOMAIN and UTILITY, permit to respectively define the domain of interest of the user as a set of abstract subjects, and the relative importance of these subjects as a set of equations. Example 2, taken from

[4], defines a traveller profile whose domain of interest is composed of three subjects: car rental companies (CR), airport shuttles schedules (SS) and availability of city maps (CM). Three utility equations complement this definition by setting the relative importance of these subjects as well as their dependencies. The first equation means that the first 2 shuttles schedules have a utility equal to 5 while the remaining schedules have a utility equal to 1. The second equation means that the utility of the first two city maps objects is 3, and the remaining objects of that category have a utility of 0. The third equation defines the context where the car rental utility function has a meaning, that is when there is at least one city map (in that case, the first 3 companies have utility 2 while the others have utility 0).

Example 2 :

```
PROFILE Traveller
  DOMAIN
    CR = www.hertz.com (car rental companies)
    SS = « Airport shuttle schedules»
    CM = « city maps and maps from Airport to the city»
  UTILITY
    U(SS) = UPTO (2, 5, 1)
    U(CM) = UPTO(2, 3, 0)
    U(CR [#CM > =1]) = UPTO (3, 2, 0)
END
```

In [4], such profile is used in cache management. The main idea behind is to maximize the utility of the cache, which is defined as the sum of the utilities of the objects in the cache. Suppose that objects are Web pages and the cache contains 4 pages of car rental companies (CR), 1 page with city maps and maps from Airport to the city (CM) and 3 pages with airport shuttle schedules (SS). The utility of this cache is the sum of utilities of contained pages. The utility of the CR pages is given by the third utility function. First, there should be at least 1 CM page (#CM > =1). If this condition is satisfied, the utility of each CR page will be considered. The 3 first CR pages have a utility of 2 while the remaining one has utility 0 (UPTO (3,2,0)) The utility of the 4 CR pages is 2+2+2+0 = 6. Using the same process, the utility of the CM page is 3 and the utility of the 3 SS pages is 5+5+1=11. The global cache utility is then 6+3+11=20.

Example 3

The third example is given by [14] in the context of relational databases. A user profile is defined with respect to a database schema as a list of weighted predicates which represent the most frequent selection predicates that are supposed to appear in user queries. The user interest on each predicate is represented by a weight between 0 and 1 to emphasize the relative importance of one predicate with respect to others. Example 3, taken from [14] defines the profile of a traveller living in Paris, who generally likes two days trips, prefers better trains than cars and in the worst case boats. He books his rooms in city center hotels.

Example 3:

```
{   TRANSPORT.idT = TRAVEL.idT       1          (a)
    HOTEL.idH = TRAVEL.idH           1          (b)
    TRAVEL.departure = 'Paris'       1          (c)
    HOTEL.area = 'city center'       1          (d)
```

```
TRAVEL.number_of_days = 2        0.7          (e)
TRANSPORT.mean = 'train'         0.7          (f)
TRANSPORT.mean = 'car'           0.5          (g)
TRANSPORT.mean = 'boat'          0.3          (h)    }
```

This profile is further used to enrich user queries by introducing new selection predicates taken from the profile and improving the relevance of the query. This is done with the use of three parameters: (i) the number K of predicates with the highest degrees taken from the user profile that will affect the query, (ii) M which is the number of predicates among the K predicates selected above that will be considered as mandatory (the mandatory predicated are selected with the highest degree of interest) and (iii) the number of the remaining (K-M) predicates that at least must be satisfied by each result. Given a user query, a user profile and three parameters K, M and L, the query reformulation is done in two phases: (i) predicates selection and (ii) predicates integration to the query. Predicates selection consist of finding the top K predicates (those with the highest weights) that are related to the query and are not conflicting with it (i.e. no contradiction between their predicates). Once the top K predicates have been chosen, they are integrated into the query. This is done in two steps: (i) make all mandatory predicates as one conjunctive clause, (ii) make a conjunctive clause with each possibilities of L among (K-M) remaining predicates, (iii) make a disjunctive clause of all these possibilities, and finally (iv) add these new clauses as conjunctives subqueries to the initial user query. For example, given the following query which searches for 3 days trip to Madrid for a given date:

```
// Initial user query
SELECT T.idT
FROM TRAVEL T, DEPARTURE D
WHERE T.idT = D.idT AND D.date = '10/11/2004' AND
    T.arrival = 'Madrid' AND T. number_of_days = 3
```

Having K=6, M.= 3 and L=2, the previous query is reformulated to the following one.

```
// Reformulated user query
SELECT T.idT
FROM TRAVEL T, DEPARTURE D, HOTEL H, TRANSPORT TR
WHERE T.idT = D.idT AND TR.idTR = T.idTR AND H.idH = T.idH
    AND D.date = '10/11/2004' AND T.arrival = 'Madrid'
    AND T.number_of_days = 3 AND T.departure = 'Paris'
    AND( (H.area = 'city center' AND T.mean = 'train')
          OR(H.area = 'city center' AND T.mean = 'car') )
```

As previous examples have shown, there are as many profile definitions as application domains and technologies. But, regardless to a specific application, a profile defines at least a domain of interest (user's interest with respect to a certain business perspective), and a set of preferences over this domain. The domain of interest is mainly defined as a set of object or relations and a set of predicates specifying their semantics. The preferences are defined as weights associated to keywords or to predicates, or as various expressions including utility functions and partial ordering.

But a user profile may contain more knowledge, such as personal data, data delivery preferences, data quality and security preferences. The next section defines the

main categories of knowledge as well as the different ways preferences are defined on this knowledge.

3. A Multidimensional Approach to Profile Definition

Identification and organization of profile knowledge is a key issue to have a global view of data personalization. Different attempts have been done to collect and classify this knowledge. For example, P3P [9], as a standard for profile security, has identified three categories of profile knowledge: *demographic attributes* (e.g. identity, age, revenue), *professional attributes* (e.g. employer, job category, expertise) and *behavior attributes* (e.g. trace of previous queries, time spent at each navigation link).

Another categorization tentative has been done by [1] for the digital libraries field. They have identified four categories of knowledge: *personal data* (identity), *collected data* (content, structure and origin of accessed documents), *delivery data* (time and support of delivery), *behavioural data* (trace of user-system interactions).

Following this categorization effort, we propose six dimensions through which profile knowledge can be defined. These dimensions refine and extend previous categorization attempts, particularly by allowing preference expression and by providing some operators for their exploitation and evolution. These dimensions are open in the sense that they can easily be extended by specific operators. They will serve as a foundation to a generic profile model aimed to be used in a wide application spectrum.

Obviously, a given user may have one or several profiles, depending on the perspectives he considers. These profiles may share some attributes and preferences or may be completely disjoint. A given profile is then defined with respect to a given perspective which is described as the domain of interest in the profile. Consequently, the relevance of other dimensions is closely related to the domain of interest.

3.1. Main Dimensions

We have identified six dimensions through which a user profile can be defined: personal data, domain of interest, data quality, data delivery, security, and history of user-system interactions. A brief description is given below for each dimension.

Personal Data (PD): This dimension groups attributes and preferences related to the user himself, that is all what concern his identity, demographic data, professional data, health care data, and so on. This knowledge can be more or less detailed, depending on the application range on which this profile can be used. This knowledge can be organized into different entities, possibly organized as a generalization-specialization hierarchy. In some information systems, personal data will be used to filter query results with respect to the age of the user, his gender or his area of work. In others, personal data is useless for information filtering itself but it is still useful as an 'exchange currency' between the user and the information provider. This is the case in many e-commerce applications and web-based systems which collect personal data for statistics purpose or for publicity dissemination.

Domain of Interest (DI): This dimension groups all attributes and preferences related to general needs of a given user. It describes the perspective on which the user wants his profile be referred to. The domain of interest may describe the user's expertise or qualification in a specific field as well as the main object types he is interested in. The DI can be defined in different ways ranging from an entity-relationship diagram or a set of queries to a list of weighted key-words or weighted predicates. An ontology may complement the definition of the domain of interest by providing a semantics to the terms used within these entities and queries. Knowledge defined in the DI will be mainly used to reformulate user queries, either by term substitution, by complementing the queries with new selection predicates or by introducing orders between predicates.

Data quality (DQ): Data quality is one of the most important issues in data personalization. Most of the user preferences relate to data accuracy, data freshness, data consistency, etc. Data quality does not only concern data values but also data sources (e.g. confidence, update frequency, completeness) and data derivation process (e.g. response time, reliability). Attributes and preferences of this dimension define quality expected by the user; it will be matched against actual quality values to provide the best results to the user. Quality attributes can be organized into distinct entities describing quality of values, quality of sources, quality of derivation processes, etc. Attributes and preferences of the quality dimension are used by the query processor or more generally by the data management system.

Data Delivery (DL): Data delivery dimension describes different modalities related to user interface (e.g. media type, presentation style, results size), user mobility (e.g. geographical location, connectivity), temporal accessibility (e.g. moment at which queries are issued and result notified, login duration), etc. Most of the web-based search engines propose such modalities. Delivery modalities may depend on the media used, same query results will not be presented in the same way depending on whether the media is a laptop, a PDA or a mobile phone. Delivery preferences can be used to define such modalities and to decide in which context they are used.

Security and privacy (SV): This dimension describes security rules and constraints that can be applied either to the data resulting from queries, to the queries themselves, to the user identity or to the whole profile as a sensitive knowledge. Security dimension mainly refers to privacy policies as described in different standards such as P3P [8] and PAPI [20] for example.

History of User-System Interactions (HI): This dimension describes major attributes which keep track of interactions between the user and the system. Recorded data may concern a sample of liked or disliked information accessed by the user as well as his implicit or explicit feedback. It may also record samples of queries or streams of clicks on specific links. Information in the history of interactions is not necessarily used as it is in the queries, but is rather processed to extract new attributes and preferences which update those of other dimensions.

Each dimension is defined as a view on the generic profile, defined by a list of attributes grouped into entities and by a list of queries defining the semantics of these entities.

As mentioned before, a given user may have one or several profiles, each devoted to a specific domain of interest. Each profile is defined by one or several dimensions,

possibly restricted to the necessary attributes and preferences which are relevant to the targets applications addressed by the user.

3.2. Preference definition over these dimensions

Preference definition may concern any of the six dimensions of the generic profile. They can be defined either on the data instances which can be queried by the user, on the predicates which define dimension entities, or on other elementary preferences (e.g. preferences ordering). This section presents a taxonomy of the most used preferences and discusses the languages in which they are defined.

Preference categories

There are mainly three operators which defines preferences over a set E of elements: Best(E), Top(k,E) and First(k,E).

The *best-of* operator has been introduced by [5] and [19]. It is defined on the basis of a preference relation $\rangle$ between two elements of E: $Best_{\succ}(E)=\{e\in E/not(\exists e'\in E, e'\succ e)\}$. The preference relation $\succ$ introduces a partial order between elements of E. Intuitively, it may correspond to an SQL aggregate function with a possible grouping attributes defined on E. If we have a set of weighted elements, the aggregate function computes the subset of elements that have the higher weight. It may also be a match which minimizes a similarity distance between selected elements. We will come back later on the main preference relations used.

The *top-k* operator behaves almost like the *best-of* operator but imposes the number of results to k elements. If the number of the best elements is greater than k, options can be defined to select randomly the k ones or to take the first k ones among these. If the number of best elements is less than k, the top-k operator takes the remaining elements among the ones ordered immediately after the best elements, that is those defined by Best(E-Best(E)). If E is ordered by the preference relation $\succ$, the definition of the top-k operator is the following: $Top(k,E)= \{e_i\in E / 1\leq i\leq k \wedge not(\exists j>i, e_j \succ e_i)\}$.

The *first-k* operator allows to limit the search to a subset of possible results, regardless to any selection criteria, except the order on which the elements are computed or delivered. In an instance-based computation approach, the first-k operator limits the computation to the first k elements. In a set-based approach, it limits the delivery to the first k elements. Depending on the context, if necessary to avoid confusion between both semantics, it might be interesting to differentiate the name of both; e.g. Firstc(k,E) and Firstd(k,E) for respectively first computed and first delivered. Given a precedence operation $x\lhd y$ which stands for *x is computedbefore y* or *x is deliveredbefore y*, the first-k operator can be defined as follows: $First(k,E) = \{e_i\in E/ 1\leq i\leq k \wedge not(\exists e_j\in E, e_j\lhd e_i)\}$.

These three preference operator types can apply to any set E where a preference relation or an order can be defined among its elements. The set E may be a set of symbolic values, a set of attributes, a set of predicates or even a set of preferences. They

can be used either to define ground preferences on attribute values or composite preferences on ground preferences.

Preference relations

As seen before, the main operators expressing preferences on set elements are based on preference relations. Preference relations establish a binary link between two elements e1 and e2 to specify that the first is preferred to the second. Preference relations on the set E can be defined in various ways:

- A weight associated to each element of the set E, leading to a total or partial ordering of its elements. The third user profile given in Section 2 is an example of such an approach. Preferences are represented by a set of weighs attached to predicates. The Top K predicates used to reformulate a given query are selected among the ordered list of predicates.
- A fuzzy operation such as ARROUND(v) which partitions elements of E into two subsets using an appropriate distance. It generally applies to numerical values, splitting E into elements whose values are close to v and elements whose value are far from v [13].
- A classification procedure which classifies elements of E into predefined classes characterized by generic patterns or use-cases. The pattern-matching operation underlying this procedure takes an element of e and decides whether it matches with one of the given cases. This classification is often termed as a case-based reasoning. For example, in CASPER [3], a system for electronic recruitment, the user profile is defined as two disjoint classes of jobs: those liked by the user and those disliked by the user. When a new job announcement arrives, the server pushes it to the client side and classifies it with respect to existing cases.

We assume that preference relations are used-defined relations which should be specified within the profile or imported from outside. This way of doing allows each user to give a specific semantics to operators like ARROUND, LOWEST and HIGHEST.

Preference composition

Given a user profile, satisfying all the defined preferences is not always necessary nor always possible. Hence, it becomes useful to define a preference among ground preferences. Depending on whether preferences are dependent or independent, ordered or not, their composition should follow semantic rules which clarify how to evaluate resulting expressions. For example, in [15], preferences are assumed as independent and of equal importance. Introduced with the clause PREFER, they behave as a filter on the query results, applying the maximum number of preferences. Selected results satisfy the same number of preferences although each of them does not necessarily satisfy the same list of preferences.

Composing dependent preferences or preferences of non-equal importance needs more semantics to clarify how composed preferences are evaluated. Given a set of ordered preferences, different semantics can be defined for its evaluation:

- Do the best and stop at the first preference that gives non empty result. This evaluation is meaningful if the preferences are independent.
- Do all preferences in descending order until one gives an empty result. This preference composition is used when preferences are strongly dependant. Satisfying a preference is meaningful only if its predecessors have been satisfied
- Do all of them without taking into account preferences giving empty results. In this case the aim is to satisfy as many preferences as possible but in the given order.

As an illustrative example, suppose that a user is interested in apartment announcements. His profile has two ground preferences, respectively on the apartment price and on the apartment exposition. The composite preference should specify whether the two preferences are of equal importance or whether one (say price) dominates the other (say exposition).

Specifying user preferences in information retrieval or database system is not a new issue, early work has been done in the past two decades [2, 4, 5, 6, 7, 10, 12, 13, 14, 15,,19]. With the increasing development of web systems and the explosion of distributed data sources, the problem has gained a new interest. An intensive effort has been devoted recently to preference languages [8, 9, 13, 4] and preference theory [17, 5, 21]. In the database field, a kind of preferences is first introduced through specialized operators such as WINNOW [5], SKYLINE [2] and TOP-N operators. Other preferences have been introduced through partial ordering of selection predicates, using specific clauses such as PREFERRING-CASCADING [13], PREFER-FROM WHICH PREFER [15], utility functions [4] or weighted predicates [14]. There is a close link between preference languages and approximate queries. Many approaches use fuzzy operators, such as ARROUND operator or approximate joins, to express a specific preference [13, 18]. The semantics of the operator is usually given using a semantic distance.

3.3. Profile Ontology

Defining ontologies within a user profile is a necessary complement to give more meaning to the terms and operators used in the profile. A profile is generally defined independently of any specific database schema and any specific query language. Hence, the terms used in the profile do not easily match with those of a specific database and query language. Adding linguistic knowledge as ontologies may facilitate this matching.

The profile ontology can be represented as a set of conceptual graphs [16] associated to the vocabulary used in the profile. For example, given an entity DOCUMENT defined in the DI dimension, this entity name cannot match with specific database entities if the term DOCUMENT is not defined as possibly being one of the following terms: REPORT, WEB PAGE, ARTICLE, BOOK, PROCEEDINGS, etc. Such terms can be organized as an ISA hierarchy and used to validate elements of the profile with respect to a given context. Conceptual graphs do not concern only entity and attribute names, but also attribute values. For example, values of the attribute COLOR form a lattice that can be used as a partial order over which a preference relation can be specified.

Conceptual graphs are not the only way to explicitly define a concept. Some user-defined operators like ARROUND, LOWEST and HIGHEST may have their definition dependent of each profile. Then, their semantics can be given in some formal specification language based on abstract data type or rules.

The ontology part of a profile aims to define all the necessary knowledge to ease interpretation of profile elements, then facilitating their use in query reformulation and query execution.

4. The Generic Profile Model (GPM)

The aim of the taxonomy of profile dimensions and preferences, done in previous section, is to elaborate a generic profile model that can be used as a baseline to a large class of personalization systems. The term "generic" does not mean here a closed set of attributes and preferences, but rather a list of high level concepts which can be instantiated, specialized, refined and augmented in each personalization environment.
The requirements we have fixed to this generic model are the followings:

1. it should capture the main knowledge categories known in current personalization systems,
2. it should be independent of any DBMS or any IR system and independent of any specific application,
3. it should be easy to specialize, generalize and instantiate any entity type of any dimension of the profile,
4. it should be easy to override or to hide any attribute definition, any query and any preference definition,
5. it should be open, that is easy to add any new dimension, new attribute and new preference type,
6. it should provide a set of profile operations, including at least import/export of profiles, matching profiles, finding differences between profiles and merging profiles. Other operations such as deriving knowledge from a profile instance, validating a profile instance are also desired although not mandatory,
7. it should be portable from one environment to another and possibly transportable within a mobile device.

Providing a generic model with such features is equivalent to providing design guidelines which help developers and users create and manipulate profiles. The following subsections give a flavour of the GPM we have developed in the APMD project.

4.1. Generic entities and attributes

Figure 1 reports the main entities composing the GPM. A profile type is seen as a complex object composed of one or several dimensions and profile operations. A dimension is composed of one to several entity types. Each entity type is defined by one or several attributes over which a set of predicates can be defined. Entities are

also characterized by ground preferences which may be composed to form a composite preference characterizing a given dimension.

The model is implemented as a schema of a metabase over which a set of manipulation operations are defined to allow instance definition and manipulation.

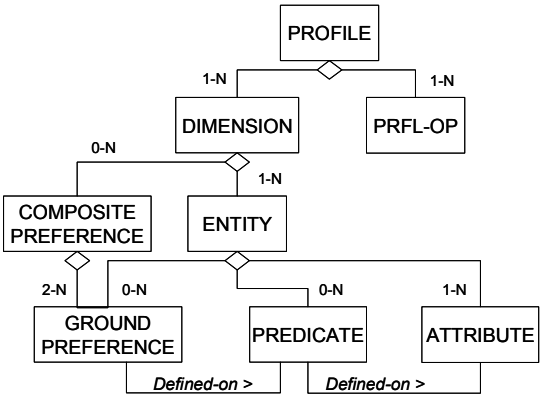

Figure 1. Meta entities of the GPM

4.2. Basic operations on profiles

Profile management means providing facilities to ease user profiles definition and evolution. The following operations are identified as fundamental tools for profile management:

- Instantiation: this operation is done in three phases: the first phase selects within the PGM concepts those which are relevant to the given user (dimensions, entities, attributes and preferences). The second phase, which is optional, extends these elements by specialization, renaming and overriding attribute and preference definitions. The third steps instantiates the profile model with respect to the user needs.
- Import profile: this operation allows to instantiate the PGM using a profile defined elsewhere. The instantiation consists in reformulating elements of the imported profile into those of the PGM. This needs the use of primary operation which is a match of profiles.
- Export profile: this operation allows to validate a given user profile with respect to the application environment in which it will be exploited. For example, given a specific database environment described by its schema and metadata, the export operation a of user profile to this environment consists in checking whether there is any link between the database definition and the domain of interest in the profile. More generally, it will check at which extent the application environment is able to handle the whole or part of element of the given profile. If this operation succeeds in binding some elements of the profile to elements of the database schema, we say that the export operation has

succeeded. This operation uses also the primary match operation plus specific derivation rules.
- Profile matching: compares two profiles and detects theirs matches. A similarity distance can characterize each matching pair of concepts. Two variants of this operation are defined, profile matching at the type level and profile matching at the instance level.
- Profile merging: instantiates a new profile by fusion of 2 or several profiles. The fusion is similar to schema integration, it has to solve all conflicts between object types and between preferences. If the input profiles have well-defined ontologies in their definition, this will considerably simplify their fusion.
- Diff of profiles: compare two profiles and detects their differences.

The current version of the PMT we have developed implements the instantiation, the two variants matching and the diff.

4.3. The GPM usage

Having the GPM and its associated PMT, the first question which arises is: to which purpose this is useful in data personalization? The answer comes from the following observation: a significant effort has been devoted to specific aspects of data personalization based on profiles, such as query reformulation, query optimization, cache management, relevance feedback evaluation, specific database operators, preference languages and so on. Important results have been obtained and many researches are still ongoing on these topics. However, there is not a global view on data personalization and on the underlying profile concepts. Although sparse, the knowledge exists and the GPM model aims to provide a generic view of it.

Having the GPM model and the PMT, the remaining questions are how to use them, where and when to use them. The answer to the first question is: both elements are tools that contribute to define and manipulate profiles. They do not constitute a personalization system but provide support to build this latter one, just as a database design tool complements a DBMS but is not a DBMS.

The answer to the two other questions is dependent of the query life cycle. Parts of the profile can be used at compile time to rewrite queries, others can be used to derive an optimized execution plan, others can be used at execution time and others at data delivery to the user. Depending on the query types (continuous queries, ranked queries, approximate queries) and their execution models, this life cycle can be refined and other steps may emerge where some elements of the profile might be relevant.

Our profile management platform (figure 2) is designed as a support to profile definition and manipulation. Its main feature is its flexibility and ability to integrate new dimensions, new entity types and attributes types and new preference types. The ontology which accompanies each profile guarantees the interpretation of the profiles and their manipulation through an extensible list of operations.

Many of the functionalities listed before are far from being fully defined or implemented. The work reported here is the result of an early phase of a multi partners

project on data personalization, funded by the French Ministry of Research under the specific program on Data Masses (APMD Prooject, ACI Masses de Données).

This project is not limited to the generic model presented here but has ambitious goals of dealing with query reformulation, query optimization, quality of data, fuzzy data and applications on different domains. The project is a three-year projet and is at the end of its first year.

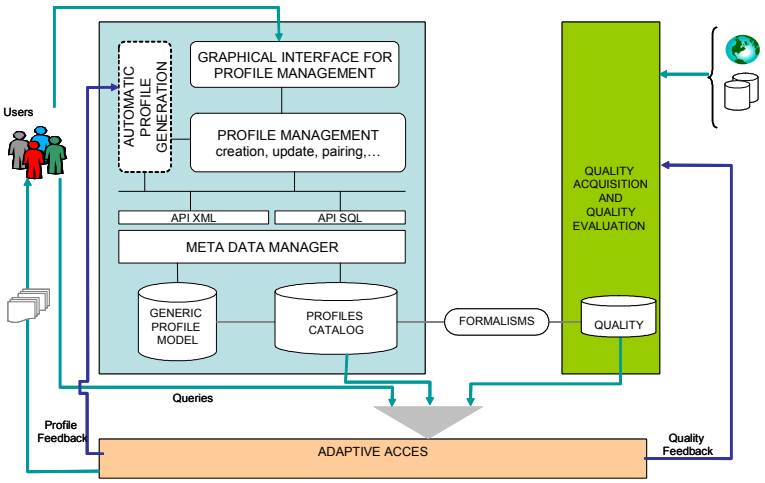

Figure 2. The PMT architecture

5. Concluding remarks

Elaborating a generic model for profile management is a first step to a global view of data personalization. This paper has defined the taxonomy of different knowledge which constitute this generic model. A set of predefined operations allow to adapte, instantiate and validate profile instances. A first prototype of a profile management tool has been developed and demonstrated. Current and further developments aim to have a complete set of manipulation operations that can be used in different tasks of the APMD project.

References

1. Amato G., Straccia U., User Profile Modeling and Applications to Digital Libraries. In: Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libraries, Paris, France (1999)

2. Borzsonyi S., Kossmann D., Stocker K., The Skyline Operator. In Proceedings of the IEEE Conference on Data Engineering, pages 421-430, Heidelberg, Germany (2001)
3. Bradley K., Rafter R., Smyth B., Case-Based User Profiling for Content Personalisation. *In: Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, Trento, Italy, August (2000)
4. Cherniack M., Galvez Ed., Franklin M., Zdonik St., Profile-Driven Cache Management. *In: Proceedings of the 19th International Conference on Data Engineering*, Bangalore, India (2003)
5. Chomicki J., Querying with Intrinsic Preferences. *In: Proceeding of the 8th International Conference on Extending Database Technology*, Prague, Czech Republic (2002)
6. Croft W. B., Cronen-Townsend St., Lavrenko V., Relevance Feedback and Personalization: A Language Modelling Perspective. *In: Proceedings of the Second DELOS Network of Excellence Workshop on Personalisation and Recommender Systems in Digital Libraries*, Dublin City University, Ireland, 18-20 June (2001)
7. Crabtree B., Soltysiak S., Automatic learning of user profiles-towards the personalisation of agent services. *BT Technol J.* Vol 16 No 3, July (1998)
8. Cranor L., Dobbs B., Egelman S., Hogben G., Humphrey J., Langheinrich M., Marchiori M., Presler-Marshall M., Reagle J., Schunter M., The Platform for Privacy Preferences 1.1 (P3P1.1) Specification. W3C Working Draft 1 July (2005), http://www.w3.org/TR/2005/WD-P3P11-20050701/
9. Cranor L., Langheinrich M., Marchiori M., A P3P Preference Exchange Language 1.0. W3C Working Draft 15 April (2002), http://www.w3.org/TR/P3P-preferences/
10. Dai H., Mobasher B., Luo T., Nakagawa M., Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization. *Data Mining and Knowledge Discovery*, 61-82 (2002)
11. Ferreira J., Silva A., MySDI: A Generic Architecture to Develop SDI Personalised Services. *In: Proceedings of the 3rd International Conference on Enterprise Information Systems*, Setubal, Portugal, July 7-10 (2001)
12. Gauch S., Pretschner A., Ontology Based Personalized Search. *In: Proceeding of the 11th IEEE Intl. On Tools with Artificial Intelligence*, pp. 391-398, Chicago, November (1999)
13. Kießling W., Foundations of Preferences in Database Systems. *In: Proceedings of the 28th Conference on Very Large Data Bases*, Hong Kong, China (2002)
14. Koutrika G., Ioannidis Y., Personalization of Queries in Database Systems. *In: Proceedings of the 20th International Conference on Data Engineering*, Boston, Massachusetts, USA, April (2004)
15. Lacroix M., Lavency P., Preference:Putting More Knowledge into Queries. Proceeding of the 13th VLDB Conference, Brighton (1987)
16. Mineau G. W., Moulin B., Sowa J. F., Conceptual Graphs for Knowledge Representation, In Proceedings of International Conference on Conceptual Structures (ICCS '93), Quebec City, Canada, August 4-7, (1993)
17. Ozturk M., Alexis T., Vincke P., Preference Modeling. Kluwer Academic, Dordrecht (2004). Also DIMACS technical report 34 (2003)
18. [GIJ+ 01] Gravano L., Ipeirotis P., Jagadish H.V., Koudas N., Muthukrishnan S., Srivastava D., Approximate string joins in a database (almost) for free. In Proceedings of the 27th International Conference on Very Large Databases (VLDB), Rome, Italy (2001)
19. R. Torlone R., Ciaccia P., Which Are My Prefered Items?. Workshop on Recommendation and Personalization in eCommerce (RPEC 2002), Malaga, Spain (2002)
20. PAPI, http://icl.cs.utk.edu/papi/pubs/index.html
21. Dagstuhl seminars, http://www.dagstuhl.de/04271/Materials/