

Profiles semantics for personalized information access

Pascaline Laure Tchienehom

IRIT, 118 route de Narbonne, 31062 Toulouse cedex 4, France
Pascaline.Tchienehom@irit.fr

Abstract. Heterogeneity of information and users has raised the problem of defining a generic model, which would be used as a basis for various information access applications. In this article, we propose a profile generic model, which describes the general structure and semantics of any information or user profile for information access. Thanks to a graph, combining two profiles instances of the generic model, we define rules for deducing couples of profiles elements that have a compatible semantics and hence that we can match. These rules aim at improving the interoperability between different profiles and applications for information access.

Key Words: Information access, personalization, profiles, semantics.

1 Introduction

Information relevance in information access techniques has led to the definition of various models for information and users. These models are of different types (document parts, documents, documents collection, thesis, articles, individual user, users group, etc.). In information retrieval and filtering, one can restructure information according to users granularity (individual user or users group) and/or according to information granularity (document parts, documents, documents collection). The goal is to discover specialized collections in some specific fields, to discover new information or to find all relevant information units for a given need while adapting to characteristics of each user or users groups. There is a multitude of information access approaches which try to solve these problems. The heterogeneity of these approaches and of the subjacent models gives a greater scale to problems related to generic models definition for the design of these systems and for the interoperability between different models and applications.

In this article, we are interested in the definition of a profile generic model (in UML) which allows to describe any type of profile for information access. The specificity of this generic model is related to the integration of semantics which enable the cooperation or interoperability between different profiles models. This profile generic model describes the profiles structure and contents as well as the semantic relations between profiles elements. These relations will allow the construction of a RDF/RDFS/OWL semantic graph, by combining various profiles instances. The objective of this semantic graph is to identify couples of descriptive profiles elements which have compatible semantics (couples that we can

match). For that, rules (or constraints) on semantics are clarified. These rules will improved the cooperation between profiles describes by different taxonomies (logical structures).

2 Literature review

Access techniques to information allow an individual to obtain information that meets his needs. We can gather them in two main groups: the pull technique, which needs an explicit request of an individual and the push technique, which does not need an explicit demand to return information to users.

Information Retrieval (IR), which is a pull technique, rests on need expression of an individual through a query formulated in a more or less structured free language [1]. However, in Information Retrieval, the real intention of the user is not always obvious in his manner of formulating his query and that can generate ambiguities on the sense of words that it contains. Many solutions exist in order to precise the sense of a query through query reformulation [3].

Information Filtering (IF), which is a push technique, is a relatively passive task because the user does not explicitly formulate his needs through a query, as it is the case in IR. In Information Filtering, we rather use a representation of the user called user profile to send information to him. There are several methods of filtering based on users: interests centers [12]; judgements [7]; demographic data (age, profession, etc.) [11]; etc.

There is a multitude of information access methods. They are based on a description of the data handled by processes of retrieval and filtering that are called profile. The profile of an object is a whole of characteristics which allows to identify and to represent it. The profiles used in information access techniques are of varied nature: user profile, document profile, etc. Their structure can be made up of one or several descriptive elements (or criteria or attributes): centers of interests, data demographic, user preferences, key words, documents metadata, etc. The semantics of these profiles attributes in traditional information access is always considered as implicit and depends strongly on the application. That poses the problem of profiles co-operation described by different structures and taxonomies (attributes names). Consequently, there is a need of: generic models [10]; semantic models [6]; extensible, flexible, re-usable and interoperable models [2]. Our contribution aims at proposing solutions in this framework through the use of semantics.

3 Defining profiles for information access

In this section, we present a profile generic model for the structure, contents and semantics description of any profiles for information access. Thereafter, we describe a profiles semantic graph, which combines instances of the generic model, allowing profiles cooperation describe by different logical structures in order to automatically infer attributes couples of compatible semantics.

3.1 Profile generic model

In order to be able to define various profiles which are reusable, multi-facets, adaptable, extensible and evolutionary, we define a profile generic model.

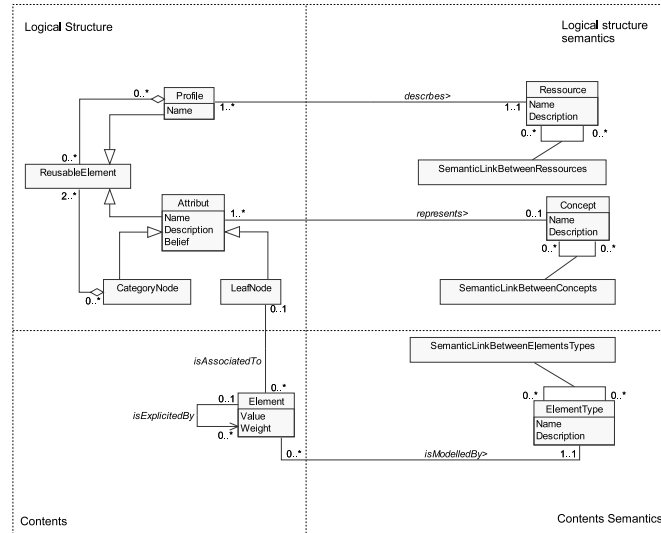


Fig. 1. Profile generic model

The figure *Fig. 1* presents the profile generic model. It results from the analysis of various systems of retrieval and recommendation in order to deduce a general model from them. The existing systems are conceived to achieve particular goals according to specificities of their context: recommendation of Web pages according to bookmarks [13], mails filtering [7], electronic trade [5], etc. Contrary to these systems, our model is enough general to be used by various applications.

The profile generic model of figure *Fig. 1* is subdivided into four levels: *the profile logical structure, the profile contents, the profile logical structure semantics and the contents semantics.*

The *logical structure* presents the general structure of a profile. This structure is in the form of a hierarchy of re-usable elements (*ReusableElement* class) that characterize it. This hierarchy is a tree where node can be: profiles or attributes. There are two types of attributes: the class *CategoryNode* attributes that are categories or attributes classes (for example the attribute *user preferences* can be composed of attributes: language, length and date.) and the class *LeafNode* attributes that are leaves attributes to which we can affect values.

Moreover, profiles derived from the generic model can have the following characteristics:

- *re-usable profiles*: the reflexive aggregation on class "profile" describes the fact that a child node of a profile can have the structure of another existing profile. For example, a long term user profile can be composed of its various usage profiles (or short term profiles);
- *multi-facets profiles*: profiles can be analysed under various aspects (attributes, sub-profiles). Thus, each profile or attribute or combination of profiles or profiles attributes can constitute a facet of it. For example, we can analyse a user profile according to facets: *demographic data and judgements, interest centers, etc.*;
- *adaptive and evolutionary profiles*: our profiles can be modified and can evolve in time. For example, a user profile can evolve if many of his short term profiles are different from his profile long term.

The *profile leaves contents* (see class Element) are lists of *Value-Weight* (see class Element) couples. These lists can contain one *Value-Weight* couple (for example the attribute value of a *document size*) or several *Value-Weight* couples (for instance the attribute value of a *document key words*).

The interest of using a generic model to define a given profile is that the basic structure that it proposes can be used by any type of application in order to define any type of profiles [4]. The figure *Fig. 2* presents instances of our profile generic model that describe mainly the structure and the contents of a user profile and information profile.

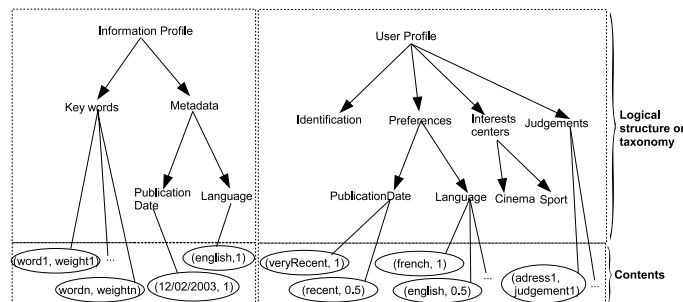


Fig. 2. Information and users profiles examples: structure and contents

The generic model will also enable us to clarify the semantics of a profile logical structure and contents. The *logical structure semantics* of the generic model clarifies what a profile and an attribute represent. A profile is the description of a resource (information or user) in a given context. Thus, the profiles can relate to users (individual or group) or to information placed at disposal (documents parts, documents, collections, etc.). Let us note that a user profile can also be: of short term (profiles built over a short period of time) or of long term (profile built over a relatively important period) [14], positive or negative [9]. The figure

Fig. 3 illustrates instances of profiles that are instances of the class "Resources", with the semantics which connects them.

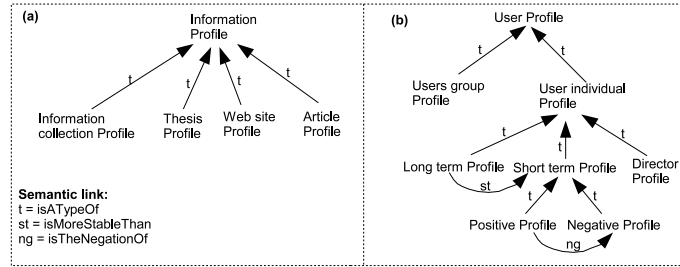


Fig. 3. Semantic relations between instances of the class "Resources"

The *attributes semantics* clarifies the characteristic that the attribute describes. The figure Fig. 4 illustrates an example of profile attributes semantics.

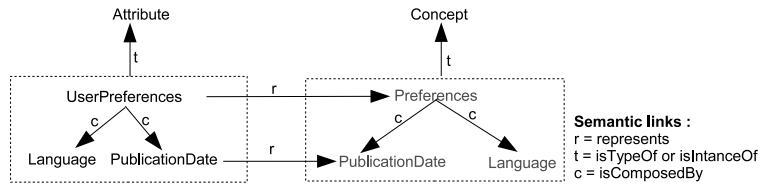


Fig. 4. Instance of profile attributes semantics

The *contents semantics* of a profile clarifies the representation model adopted or type (instance of class TypeElement) for contents elements of a leaf attribute (cf. XMLSchema element type). The figure Fig. 5 illustrates semantics instances of contents elements for leaf attributes: *ArticlePublicationDate* and *UserPreferencesPublicationDate*. These two attributes are not represented in the same reference system but it would however be interesting to be able to deduce that we can compare them by extracting year from date and by changing the representation base. This example shows the interest of clarifying the leaf attributes values semantics.

From the profile generic model, we can derive various profiles structure by applying decomposition rules to attributes and profiles classes. We can also derive the semantics of profiles, attributes and contents. This will facilitate the cooperation between different profiles in information access by deducing, through inferences rules, attributes of compatible semantics.

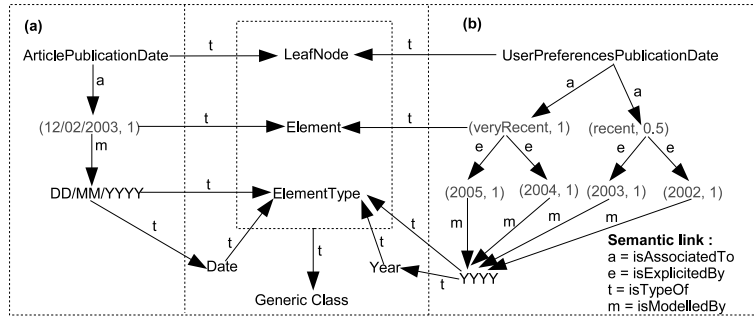


Fig. 5. Instances of leaf attributes values semantics

We have chosen RDF/RDFS/OWL descriptive language as a formal framework to combine instances of the profile generic model. The RDF/RDFS/OWL formalism is also used to formalize rules for deducing attributes couples of compatible semantics, i.e. that we can match, between two profiles of disjointed logical structure. This aspect of semantics is described and illustrated in the following section.

3.2 Semantic graph of profiles

To match two different profiles describing various resources instances in different taxonomies (attributes names), it is necessary to be able to determine attributes couples of leaf type that we can match between these profiles. For that, it is necessary to define the semantics of each profile leaf attribute and contents as well as rules that allow to deduce these couples. The semantics of leaf attributes clarifies the characteristic represented by the attribute while the semantics of a leaf attribute contents elements describes the representation vectorial space and the *value or vectors* "model or representation type". We used formalism RDF/RDFS/OWL to formally clarify the inference rules of attributes pairs that we can match. For that, we built a semantic graph which combines profiles instances derived from the generic model of figure FIG 1. Any semantic relation in our graph is thus defined in the shape of a triplet as follows: $\{subject, predicate, object\}$.

The figure FIG. 6 presents an extract of semantics description of certain information and user profiles attributes. This extract puts forward the interest of the *Semantic Web* for profiles description [6] and especially for profiles matching. This graph can be seen like an ontology of task for the profiles matching. The resources of our semantic graph are:

- preset matching classes describing a concept or characteristic represented by an attribute;
- classes representing profiles descriptive attributes and contents (logical structure and contents);

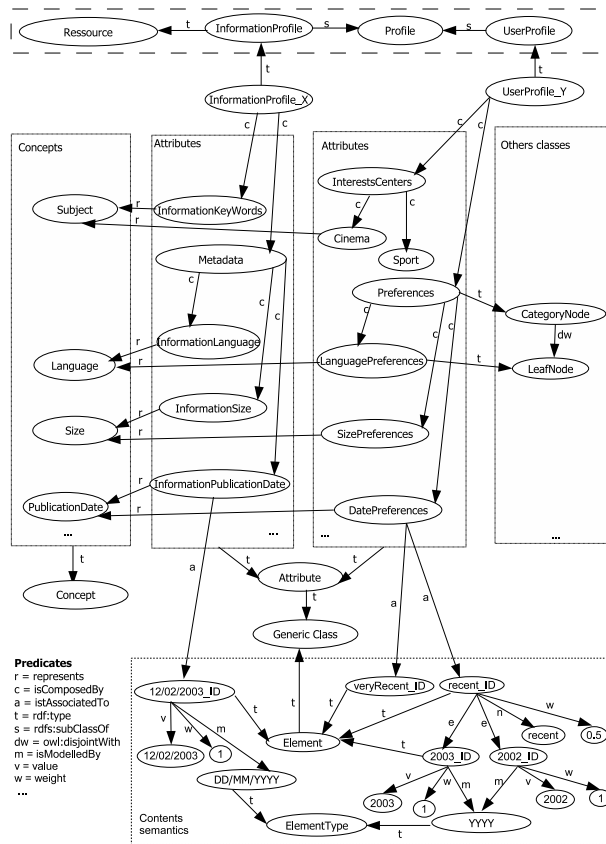


Fig. 6. A semantic graph extract combining profiles instances

- classes giving additional information on the representation model of leaf attributes contents elements like: the measuring unit used for an attribute evaluation (*terms number or bytes* for the attribute *length*), the representation type of leaf attribute element value (text, numerical, date formats, etc.), the reference system representation or vector space (list of elements values), etc.

To describe the semantic relations, we had to define certain predicates as: *represents* (noted *r*) to clarify the characteristic represented by an attribute, *isComposedBy* (noted *c*) to represent the composition link (between attributes and/or profiles), *isModelledBy* (noted *m*) to define the type of contents elements, etc. These predicates are supplemented by RDF/RDFS/OWL predicates which make it possible to typify the various elements of a triplet (*rdf:type* to subsume a class, *rdfs:subClassOf* to subsume all the instances of a class, etc.) and to define

constraints on sets of classes (*owl:disjointWith* to translate disjunction between two classes, etc).

Two leaf attributes can be matched if the following rules are verified:

1. *necessary rule*: the two leaf attributes must describe the same concept or matching class (*Subject, Language, Length, etc.*). They have to be connected to the same semantic concept class by the predicate "represents" (noted *r*). Formally, the *necessary rule* for considering a matching between two attributes *x* and *y*, noted *necessary_rule(x, y)*, is written:

Given $x, y \in A$, *x* and *y* can be matched if and only if, $[x, rdf : type, LeafNode]$, $[x, rdf : type, CategoryNode] \in G$ and $\exists a \in C$ so that $[x, r, a]$ and $[y, r, a] \in G$, where *A* is the profiles attributes set, *C* the concepts classes set and *G* the triplets set of the profiles semantic graph.

2. *necessary and sufficient rule*: to carry out matching between two attributes, it should be checked that these attributes have the same semantic links or predicates (in term of number and type) which start from these attributes towards the same semantic classes. If the semantic links are the same but are not always defined towards the same classes, it should be checked if the object of the triplet is a value. If it is the case, it is necessary to seek the semantics of the terms of the various values and to verify that they are of the same class or that there is a transformation rule between these terms semantics instances (instances of class *SemanticsTerm*). For example in figure FIG. 6, to match the *InformationPublicationDate* attribute and the *DatePreferences* attribute it is necessary that there is a rule that allow to pass from class *DD/MM/YYYY* to class *YYYY*. Thereafter, it is necessary to verify the dimension (terms number of a value) of each attribute value and its reference system (list of the terms name). If the reference system is not the same one, i.e. there is a disjunction between the terms lists, it is necessary to carry out a reference system change (vectorial space change), from the reference system of the smallest dimension towards that of the highest dimension. If there is an inclusion of reference system, it is just necessary to carry out a dimension change from the smallest dimension towards that of the highest one.

Formally, the necessary and sufficient rule for matching two attributes *x* and *y*, noted *necessary_and_sufficient_rule(x, y)*, is written:

Given $x, y \in A$, *x* and *y* can be matched if and only if, $\forall [x, p, a] \in G, \exists [y, p, b] \in G$ so that $p \in P$ and

- (a) $a = b$
- (b) or $a, b \in E$ and *searching(a, a1)* and *searching(b, b1)* returns *a1* and *b1* that are *ElementType* class instances of the values *a* and *b*, respectively.

If $\exists TransformationRule(a1, b1)$ then:

- i. if *A1* (elements values list of attribute a) and *B1* (elements values list of attribute b) are disjointed, it is necessary to carry out a reference system change. For example, in figure FIG. 6, it is necessary to express the value of *InformationPublicationDate* attribute (initially expressed in the reference system "(12/02/2003)") in the reference system of *DatePreferences* attribute which is "(veryRecent, recent)";

ii. if $A1 \subset B1$ or inversely, it is just necessary to change the dimension.

Let us note that the function $searching(a, a1)$ with $a \in E$ is defined recursively as follows:

- i. $\exists [a, isModelledBy, a1]$ and $\exists [a1, rdf:type, ElementType] \in G$
- ii. and if $\exists [a, isExplicatedBy, v1] \in G$ with $v1 \in E$ then $searching(v1, a1)$

Where A is the set of profiles attributes, G is the triplets set of the semantic graph, E is the semantic graph set of instances of class Element, P is the set of predicates, $searching(a, a1)$ is a method that seeks the classes $a1$ that is the *TermSemantics* class instance of the value a , and *TransformationRule*($a1, b1$) is a rule allowing the transformation of a triplet $[x1, rdf:type, a1]$ into a triplet $[x1, rdf:type, b1]$ or conversely.

To match two attributes, it is necessary to check the coherence of their semantics (characteristic represented, contents semantics). For that, we have identified transformations rules for attributes which have compatible semantics. Among these rules, we can quote: the transformation of monovalued attributes into multivalued attributes for a dimension change, the reference system change, the transformation of dates into various "date formats", etc.

The interest of a profiles attributes semantic graph as the one of figure FIG. 6 is that it makes it possible to give the names which one wishes to profiles attributes without disturbing the matching. We only have to specify their semantics. Moreover, one will be able to match profiles from various applications and/or described by different taxonomies. Our semantic graph could also be handled (updating, interrogations, etc.) through RDF interrogation languages [8]. To determine attributes couples that we can match, one can use a parser or RDF analyser that being given a RDF document, returns all the triplets $[Subject, predicate, object]$ of this document. It is the set of triplets obtained (G) that is analysed in order to determine attributes couples of compatible semantics. Examples of figure FIG. 6 triplets are: [Cinema, represents, Subject], [PublicationDate, rdf:type, Concept], [2003, rdf:type, AAAA], etc.

4 Conclusion

In this article, we present a generic model of profile which enables us to describe the structure, contents and semantics of various profiles types. We use this generic model to combine instances of profiles through an RDF graph in order to allow cooperation between different profiles in information access. This graph helps, thanks to some rules, to determine attributes of compatible semantics whatever the profiles taxonomies are and hence optimizes the profiles cooperation. We are now using these inference rules and other general constraints related to the profile generic model to implement an assistant tool for constructing profiles: structure, contents and semantics.

References

1. Baeza-Yates R. and Ribeiro-Neto B.: Modern Information Retrieval. First edition, Addison Wesley, ISBN 0-201-39829-X, (1999).
2. Berners-Lee T., Hendler J., Lassila O.: The semantic web. Scientific american,(2001).
3. Boughanem M., Chrisment C. and Soulé-Dupuy C.: Query modification based on relevance backpropagation in adhoc environment. Information Processing & Management Journal, Elsevier Science, (1999) vol. 35, pages 121-139.
4. Chevalier M., Soulé-Dupuy C. and Tchienehom P. L.: A profile-based architecture for a flexible and personalized information access. IADIS International Conference (IADIS/WWW Internet 2004), (2004), vol 2, p. 1017-1022, ISBN 972-99353-0-0.
5. Cho Y. H., Kim J. K. and Kim S. H.: A personalized recommender system based on web usage mining and decision tree induction. Expert System with Applications, (2002), vol. 23, n° 3, pages 329-342.
6. Dolog P., Nejdl W.: Challenges and benefits of the Semantic Web for User Modelling. In proceeding of AH'03, (2003).
7. Goldberg D., Nichols D., Oki B. M., Terry D.: Using Collaborative Filtering to weave an Information Tapestry. Communications of the ACM, Information Filtering, (1992), vol. 35, n° 12, pages 61-70.
8. P. Haase, J. Broekstra, A. Eberhart, R. Volz: A comparison of RDF Query Languages. In proceedings of the third International Semantic Web Conference ISWC'04, (2004).
9. Hoashi K., Kazunori M., Naomi I. and Hashimoto K.: Document filtering Method using non-relevant information profile. In proceedings of the 23rd Annual International ACM-SIGIR Conference on research and development in information Retrieval, (2000), pages 176-183.
10. Kobsa A.: Generic User Modelling Systems. User Modelling and User-Adapted Interaction, (2001), vol.11, pages 49-63.
11. Krulwich B.: LifeStyle Finder : Intelligent User Profiling Using Large-Scale Demographic Data. AI Magazine, (1997), vol.18, n° 2, pages 37-45.
12. Pazzani M., Muramatsu J. and Billsus D.: Syskill & Webert : Identifying interesting web sites, In Proceedings of the Thirteenth National Conference on Artificial Intelligence, (1996), Pages 54-61.
13. Rucker J. and Polanco M. J.: Siteseer : Personalized Navigation for the Web. Communications of the ACM, (1997), vol. 40, n° 3, pages 73-75.
14. Widiantoro D. H., Ioerger T. R. and Yen J.: An Adaptative Algorithm for Learning Changes in User Interests. In Proceedings of the Eighth International Conference on Information and Knowledge Management (CIKM'99), (1999), Pages 405-412.