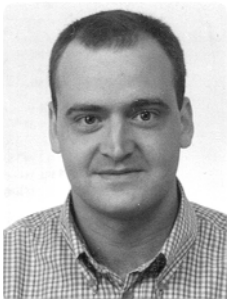
Processing the User Model in IRS[1]

David Bueno Vallejo* and Amos A. David**

* Dpto. de Lenguajes y Ciencias de la Computación, Universidad de Málaga, Spain
** University of Nancy 2, France



David Bueno Vallejo obtained his computer engineering degree at the University of Malaga, Spain in 1996. He is presently working as an assistant professor in the department of Languages and Computer Sciences at the same University. He is presently working on his Ph.D. at LORIA (Laboratoire LOrraine de Recherche en Informatique et ses Applications) Nancy, France and at the University of Malaga, Spain. His research study is on user modeling and cooperative information retrieval systems development.

Amos A. David obtained his B.Sc in computer science from the University of Ibadan, Nigeria in 1981, and his Ph.D. in computer science from the Institut National Polytechnique de Lorraine (INPL), Nancy, France, 1990. He is an associate professor at the University of Nancy 2, France, since 1991. He is the director of studies of DESS IST Nancy, France, an engineering school of Scientific and Technical Information studies. His research work is on user modeling and cooperative information retrieval systems development.

Vallejo, D. B., David A. A. (2000). **Processing the User Model in IRS** *Knowledge Organization, 27(1/2).* 17-26. 9 refs.

ABSTRACT: Our hypothesis is that when a user employs an IRS, he or she has an objective to achieve. This objective concerns the user's information need. In order to achieve this objective, the user generally does some activities using the IRS. The IRS proposes to the user solutions in response to the queries formulated by the user. The main task of an IRS is to provide the user with solutions that are relevant to his information need. This is termed personalization of information. The main axis of our study is how to personalize the system's response according to the user's objective.
We propose the use of a user model for personalizing the system's response. In our approach, the user model defines what to represent for each user. The activities of the user during the use of an IRS are recorded based on the user model. The analysis and synthesis of these activities are used to provide the user with more relevant solutions according to his objective.
Three different applications have been developed to validate our approach of personalizing the system's response and based on an architecture that we defined for a cooperative information retrieval. The three applications are METIORE_STREEMS, METIORE_LORIA and METIORE_REVUES. METIORE_STREEMS is an IRS for managing multimedia information on trees authorized for reforestation by the European Union (EU). The project was sponsored under the EU project LEONARDO. The second application, METIORE_LORIA is used for managing the publications of the computer science laboratory research center, LORIA, Nancy, France. The third application METIORE_REVUES is used for the access and analysis of a journal called Relations Publics Information.

Keywords: cooperative information retrieval, user modeling, personalization of response, relevance degree

[1] Information Retrieval System

I. Introduction

Efforts have been made to adapt recent programs to the user. Two main categories of adaptation are proposed: adaptation of the program's interface and adaptation of the program's responses to each user. A large number of "general" information retrieval systems (as compared to "specialized" systems such as database systems) are presently available. Internet technology has contributed to this proliferation of systems and provides access to constantly increasing number of users. One of the main problems with these systems is the number of responses that are given in response to the user's request. For example, it is common to obtain thousands of responses for a single request.

One of our main objectives in this study is to provide users with the most "relevant" solutions for their information needs. In order words, we try to personalize the solutions. We have introduced the concept of *objective* as a parameter in our proposed user model for the representation of a user. The objective corresponds to the expression of the user's information need, expressed in natural language. To achieve this objective, users will generally do some *activities* of information retrieval that will guide them to the best solutions. Users can evaluate the system's solutions in order to provide *feedback* to the system of how the solutions are relevant to their needs.

In our approach to personalizing the system's response, information from the user is managed as a component of an IRS, by the integration of a user model. The user model that we propose is presented in the next section.

Fig. 1 shows the standard user interface that we use for our systems. Although in some applications, we use hypertext interfaces to facilitate navigation. Zone A of the user interface allows users to make queries by making cross-analysis of the attributes of the objects that are managed by the system. Zone B of the interface allows users to exploit the history of his or her past sessions. Each user can review his or her past objectives, the associated solutions and the evaluations attributed to the solutions. The user is also able to modify the evaluation attached to a solution.

The identifiers of the solutions are sorted and colored by the importance that the system assigns to the solutions for the particular user. The full content of a selected solution is shown in another window as shown in Fig. 1. After the examination of full content of a solution, the user can evaluate. This feedback allows the system to have a better understanding of the information need of the user for calculating improved relevant solutions.

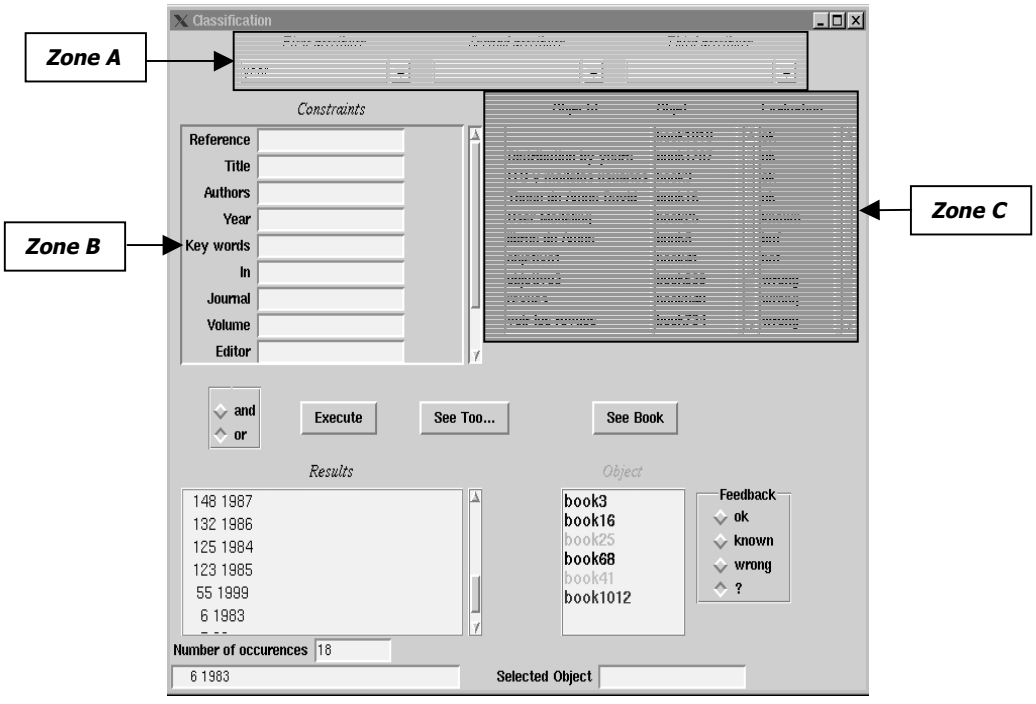


Fig. 1 Interface for the user's activities

II. The user model

We will present in this section the most important part of our system, which is the *user model*. Our studies are centered on the following fundamental questions:

- ✓ What to represent?
- ✓ How to retrieve the information about the user?
- ✓ How to exploit the model?
- ✓ How does the system identify the user's objective?

We present in the following section why the concept of *objective* is fundamental in our approach to user modeling in order to obtain a better relevant and personalized response. We also present what we mean by *session* in the context of our studies.

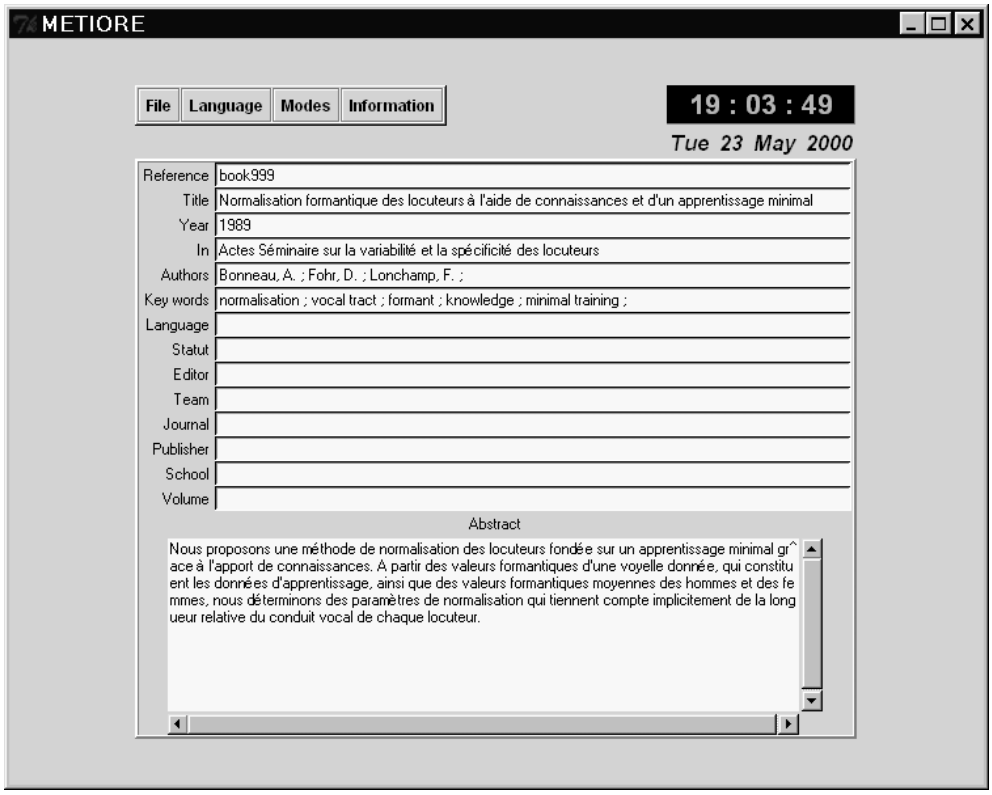


Fig. 2 Full content of a bibliographic reference

II.1 The concept of *objective* and *session*

Why do we introduce the concept of *objective* for obtaining a better relevant and personalized response? In almost all IRS, the systems try to *guess* what the user's main objective is. Of course, the user can express this main information need by a set of queries, but the queries are translations of the user's objective using the system's language. For example, each IRS disposes its own set of acceptable queries. The user's difficulties in knowing what he or she wants and how to formulate queries are presented by J. Vassileva (1994).

In our applications, the use of the system for a particular information need is considered a *session*. At the beginning of each session, the system asks the user to present his objective (that is information need) in natural language. The user also has the possibility of choosing an objective from a list of past ones. Notice that it is the user who presents this objective and it is different from the user's queries.

The objective can be used for many goals. One of the main uses is for the system to group all the inferences that have been made in the course of one or more sessions according to the user's objectives. The importance of this grouping is that the inferences made by the system about a user are associated with that user alone and his or her objectives. As will be developed later, we believe that the system's inference about a user cannot be generalized, that is, it is not applicable to all other sessions. For example:

a. A user may need a completely different type of information from one session to another
b. A user may need the same kind of information already presented in a past session
c. A user may need related information to a past one. This means that there is a factor of similarity between objectives.
d. Several users can have 'similar' objectives

Take for example a history student who goes to a library for books on the "history of Greece". Supposing that the IRS proposes books on the "history of Rome" and that the user evaluates this solution as irrelevant. The IRS cannot generalize this evaluation whenever this book is proposed. It can be applied to the book only for related objectives. For example, in the course of his or her studies, the student may come back to the library for books on the "history of Rome". If the first evaluation had been generalized, the books on the history of Rome would have been prejudged as *not interesting.*

This approach of grouping all the inferences that have been made in the course of one or more sessions according to the user's objectives is also very useful for regular users in the context of similar information need. For example, should the student in our example come back to the library for the same information need, he can select the past objective and the system will produce the associated solutions and the user's evaluations. This allows the user to effectuate information retrieval centered on information need and not on request formulation. In another situation, the student in our example who was looking for books on the history of Rome may remember that he had found and evaluated a book on this subject when he was studying the history of Greece. Using this approach, the user can go through the past history of activities and solutions associated with this objective.

This approach has been implemented in our applications. When the system shows the list of past objectives and all the related information, they are displayed using a factor of similarity with the current objective (if it is a new one).

We present in the next sections our answers to the enumerated fundamental questions.

II.2 What to represent?

As in most IRS that try to personalize the system's response, we have some relatively stable parameters for each user such as identity, preferred interface language and some configuration preferences. Other parameters are dynamic. All of the values of the dynamic parameters are attached to a given *objective*. Information on the user is organized in *sessions* as described in the last section. Each session is composed of a set of *activities.* The user's objectives, sessions and associated activities constitute the user's history. We have defined several types of activities needed in the context of information retrieval. The types of activities are based on what are considered evocative habits in the cognitive sciences. These types of activity are presented in the following section. Similar work on what about users to take into account in order to adapt the system's response and the features that can differentiate one user from another or what makes the user different at different times are treated in Peter Brusilovsky (1996). The solution proposed is to represent the knowledge, goals, background, experiences and preferences of the user.

II.2.1 The user's activities

The history of the user's activities is recorded in order to be able to analyze and synthesize his behavior. All of the user's actions are called *activities*. It is important to note that the information on the user's activities is a record of concrete behaviors, which allows considering our user model as an explicit and individualized model.

II.2.1.1 "Search" activities

The user can make a search using attributes, constraints, or both. The attributes correspond to the properties of the objects managed in the IRS. For example, for bibliographies, the attributes can be title, author, keywords, *etc*. The constraints are criteria each composed of attribute, comparison operator and value. The criteria can be combined using Boolean operators. For example *(author = John) AND (year > 1997)* is a query composed of two criteria and a Boolean operator which we consider a constraint. The user can also present just *author* as his query. And finally, the user can give *author*, *(author = John) AND (year > 1997)* as his query. The query is composed in this case of only one attribute and a set of constraints. The user can give more than one attribute in the query. The type of activity is also recorded in the history (the materialization of the user model) of the user's activities.

II.2.1.2 Exploiting the history of past sessions

Instead of formulating a new objective and trying to obtain solutions to the objective, the user can exploit his or her past history. The concept of history is

also presented in A. Kobsa, *et al.* (1996), but in our model we sort the history by past objectives and evaluations. Its analysis constitutes for us another category of activity. Through this category, the user can review his or her past objectives, the associated solutions and evaluations associated with each solution. This category of activity may be prompted in several ways including the following:

- Recollecting past solutions
- Modifying evaluations of past solutions
- Looking for already presented objects in order to find related ones
- Analyzing past solutions of precedent objectives and the solutions of the current objective.

II.2.1.3 "See also" activities

This activity lets the user find other objects that are related to a particular object in the database. Any object retrieved in response to a query can be selected by the user to obtain other related objects. The user can also choose the attribute to use to constitute the relationship. For example, the user can select an object and chose the attribute author as the relation attribute. In a collection of bibliographic references, the system will give the objects with the same authors as the authors of the selected object.

Our hypothesis is that this category of activity will be efficient for expanding the possible related relevant solutions for a given objective.

II.2.2 The user's feedback

For each of the solutions given by the system, the user has the possibility of giving an evaluation on the relevance of the solution for the given objective. The feedback and its associated solution are also kept in the model.

The user's feedback is based on the various reasons for accepting or refusing a solution. This response is different from the classical types of feedback restricted to interesting or not interesting as in I. Schwab and W. Pohl (1999) or like the proposal in Benaky *et al* (1997) where the possible evaluations are interesting, not interesting or indifferent. In these classical types of feedback the reasons for the user's decisions are not known. The following four types of feedback are implemented presently in our applications.

- ✓ *The solution is pertinent (ok)*
  This means that the solution is valid for the user.
- ✓ *The solution is not relevant because the user knows it already (known)*
  This means that the characteristics of the solution are related to the user's objective, but it's not interesting because it is already known.
- ✓ *No opinion on the relevance of the solution (bof)*
  This evaluation applies to the case in which the user is unable to give any other judgement on the solution.
- ✓ *The solution is not relevant because it doesn't correspond to the user's objective (wrong)*
  The user judges the solution to be irrelevant to the objective.
- ✓ *Unvisited solution (unvisited)*
  This is the default evaluation for all proposed solutions to signify that they are unvisited.

Remember that an object in the database is composed of attributes and each attribute can have one or more values. For example, a bibliographic reference is composed of author, title, keywords, *etc*. The *keywords* attribute can contain one or more keywords.

The feedback given by the user for a particular solution (that is an object) is attached to the object and also to all the values of the attributes. So when new feedback is attributed, the system looks for each value of the actual object and increases the number of times that this feedback has been attached to the value. It is important to note that for the same objective, the same value can have different evaluations. This information is crucial for the proximity calculation using the algorithm (presented in section III) for sorting the solutions according to the user's evaluations.

II.3 How to retrieve the information concerning the user?

In the context of our developments, information concerning the user is obtained in three different ways: automatically, by interaction with the user, and through a human expert during cooperative information retrieval.

II.3.1 Automatic acquisition of information on the user

Each activity (and associated parameters) of the user is automatically saved by the system. The history of the user's activities is automatically updated each time that the user provides feedback.

II.3.2 Acquisition of information on the user through interaction

The user can modify some parameters of his or her model, like preferences, through his feedback and by the introduction of an objective for expressing an information need. The system may initiate dialogue with the user on certain occasions. For example if the system detects that the user makes the same request several times for the same objective, the system may ask the user to explain why. The system may also request the user to confirm certain conclusions that are calculated through the synthesis of the information contained in the user model.

II.3.3 By a human expert

In the context of cooperative information retrieval, an expert may observe certain phenomenon concerning the user and interpret this observation in a particular way. This observation can lead to the updating of the user model. As explained by Brian Logan *et al* (1994), the role of intermediaries, such as the human librarian is very important in the context of information retrieval. We have defined and implemented in our application the concept of cooperative information retrieval through Internet. This concept is presented in A. David and D. Bueno (1999).

II.4 How to exploit the model?

We have developed in the previous sections an explanation of how the user model is represented, how it is fed, and how it is updated. In this section we present how the user model can be exploited. We propose an architecture that makes it possible to exploit the user model in several ways. This architecture has been implemented in the three applications that we have developed.

II.4.1 Exploitation by the user

The user can exploit his or her own model through the history of activities in the past sessions. The user can navigate through past objectives, see the solutions proposed for each of them and also his or her evaluations for the objects in the solutions. The user has the possibility of re-evaluating past solutions. In this case, the user model is updated to reflect the user's re-evaluation.

II.4.2 Exploitation by the system

The system exploits the user model in two ways. The first one is related to the objectives and the second one to the solutions.

As presented in section II.1, the user presents his or her objective at the beginning of each session. The system makes a simple vectorial analysis of the keywords contained in the present objective and the past ones. This analysis gives a percentage of similarity between the present objective and the past ones. The result is used to sort the previous objectives in decreasing degree of similarity before presenting them to the user. In this way, the most similar past objectives are on top of the list of the past objectives.

The other type of exploitation by the system is to use the characteristics of the user's past evaluations to calculate the degree of relevance of all possible solutions to the incoming request. The solutions can be sorted using the calculated relevance degree. The algorithm we propose for calculating the relevance degree is presented in the next section.

III. Proximity algorithm

The objective of our algorithm (called the *proximity algorithm*) is to calculate the probability of the evaluation (see section II.2.2) that the user would give to a particular solution in response to the incoming request, based on the current objective. Our work is based on the user's objective and not on his queries as is the work in I. Zukerman *et* al (1999) where probability and Markov models are used to predict the user's request. In the systems that do not integrate any form of personalization, the list of solutions in response to a request would be presented to the user in any order. Our algorithm is combined with the basic pattern matching algorithms for obtaining related documents to a query. Ideas that are similar to the ones we present here are proposed in Schwab & Pohl (1999). In their approach, the possible types of evaluation are limited to “interesting” and “not interesting”, or the calculation of the relevance of a solution for the user by analyzing only “positive evaluations”. In our proposal, all the possible types of evaluation as presented in section II.2.2 are integrated.

```
evaluation_history :=          element+
element :=                     objective evaluated_object+
evaluated_object =:            evaluated_attribute+
evaluated_attribute :=         attribute evaluated_value+
evaluated_value :=             value ev1 ev2 ev3 ev4
```

Fig. 3 The user's evaluation history in XML form

Fig.3 and Fig. 4 present the history of the user's evaluation for past objectives. The history is composed of elements. Each element contains an objective and a list of evaluated objects. Evaluating an object is considered to be evaluating its attributes. Similarly, evaluating an attribute is considered to be evaluating the values given to that attribute for the current object. Notice that the user evaluates an object as a whole rather than each of the values of the object's attributes. For example, if a reference in a bibliographic database is evaluated as *ok*, it means that all the values of all the object's attributes (the values of the attributes *title, author, year, keywords, etc*) are evaluated as *ok* as well. Designating the evaluation given to an object as a whole as equivalent to the evaluation that would have been given to the values of the object's attributes is a very strong hypothesis. However, we will show that our algorithm tries to calculate the evaluation that would have been given to each attribute's values if the user were to evaluate them one by one.

Each time an object is evaluated in the context of an objective, the type of evaluation is associated with the attribute's value. What we represent for each value is in fact the frequency a particular type of evaluation that is given for that value. Thus, *ev1, ev2, ev3* and *ev4* are the frequencies of evaluation for *ok, known, bof* and *wrong* as explained in section II.2.2.

```
{objective1 {attribute1 {value ev1 ev2 ev3 ev4} {value ...}}...{attributen {...}}} ...
{objectivem {attribute1 {value ev1 ev2 ev3 ev4} {value ...}}...{attributen {...}}}
```

Fig. 4 The user's evaluation history in list form

Suppose that in a bibliographic database there are the following attributes for each reference (our object): *title, authors, keywords, year, editor, place of edition, abstract*. To illustrate our algorithm, we will use only two attributes: *keywords and year*. If the user evaluates a reference as *ok*, we will increase the "ok" frequency *(ev1)* for the year given as a value for the attribute year and also *ev1* for each of the keywords given as attribute keywords.

```
TITLE: The state of art in text filtering
AUTHOR: Douglas W. Oard
YEAR: 1997
KEYWORDS: information filtering, text retrieval, ...
```

Fig. 5 Example of a bibliographic reference

A single attribute alone cannot be used to calculate the relevance degree of an object (in this case a bibliographic reference) to a given objective. Suppose that a user says in his objective that he is looking for references on Intelligent Tutorial Systems (ITS). Suppose that what the user *really wants* are recent publications. Many references containing the keyword ITS may be rejected and if the system uses only the attribute *keywords* to calculate the degree of relevance of the objects, it may conclude that the user is probably not interested in ITS. Of course, this is not the case in the present situation. The user is interested in ITS, but there are other associated parameters. In this example the parameter date should be taken into account for calculating the degree of relevance. Notice that the user's problem is probably an inability to formulate the concept of *recent* reference. Furthermore, the concept of *recent* depends on an individual and on the domain of study.

Our algorithm is based on Bayesian theory and uses the precedent of evaluations of the user. As stated earlier, the result of the algorithm will be the degree of relevance of an object to the present objective for a user. In other words, we are attempting to calculate the probability of the user giving a particular type of evaluation for a given object in the context of the present objective.

With this objective in mind we use an adaptation of the Naïve Bayesian Algorithm (Kononenko 1990) appropriate to our specific objective. The original formula is shown in (1)

$$P(C/V_{1,J_1},...,V_{n,J_n}) = P(C)\prod_{i=1}^{n} Qi(C,J_i) \qquad (1)$$

where

$$Q_i(C,J_i) = \frac{P(V_{i,J_i} \mid C)}{P(V_{i,J_i})} \qquad (2)$$

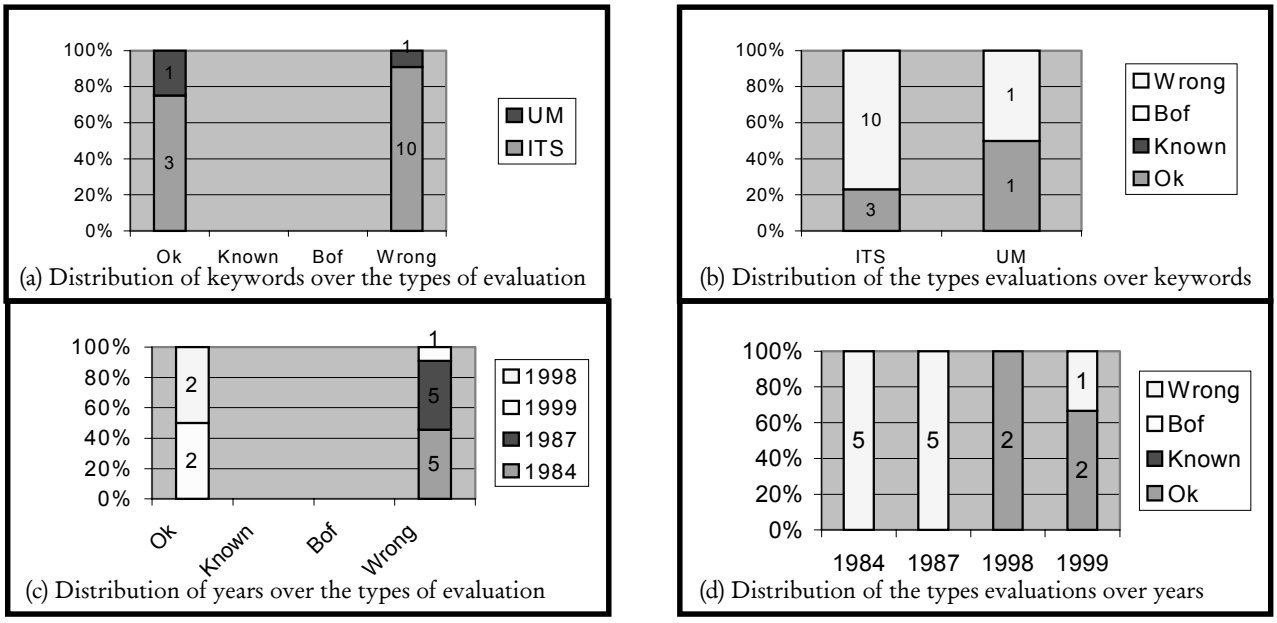
In (1) and (2), C is one of the possible classes (ok, known, bof, wrong).V_{iJi} is a Boolean variable with value 1 if the current instance has value J_i and P(C) is the probability of the class C. The equation (2) is only valid if the possible attributes are independent, as we assume in our case.

The modification of the equation (1) that we propose gives similar results but is less restrictive because we use the average of the weights of each attribute as shown in (3)

$$P(C/V_{1,J_1},...,V_{n,J_n}) = P(C)\frac{\sum_{i=1}^{n} Qi(C,J_i)}{n} \qquad (3)$$

We explain this equation with two examples where the system classifies the possible solutions. This is a form of recommendation by the system before the user starts the process of evaluation of the solutions.

Fig. 6 presents a graphic representation of 15 evaluations. Fig. (a) indicates how many times each keyword is associated with a particular type of evaluation. For example, "ITS" was evaluated *ok* 3 times and *wrong* 10 times. Fig. 6 (b) indicates the distribution of the types of evaluation over the keywords. For example, the type of evaluation *wrong* is associated with "ITS" 10 times. Fig. 6 (a) and (b) represent the same information but presented differently.



(a) Distribution of keywords over the types of evaluation
(b) Distribution of the types evaluations over keywords
(c) Distribution of years over the types of evaluation
(d) Distribution of the types evaluations over years

Fig. 6 Graphic representation of 15 evaluation

Fig. 6 (c) represents the distribution of years over the types of evaluation while (d) is another graphic representation of the same information. For example, 1998 is evaluated as *ok* 2 times and *wrong* once.

Another way of reading the graph is as follows. In (b), out of the 13 references that contain "ITS" as keyword, 10 (10/13) are evaluated as *wrong* while 3 (3/13) are evaluated as *ok*. In (d), 5 references are pub-

lished in 1984. All of the five references containing this year are evaluated as *wrong*. Three references are published in 1999 of which 1 (1/3) was evaluated as *wrong* and 2 (2/3) as *ok*.

Supposing that a bibliographic reference contains the **keyword "*ITS*" and published in the year 1998,** our algorithm should calculate its degree of relevance, that is the probable type of evaluation that the user will give to it.

Here is some information derived from the above graphics:

- Out of the 15 keywords evaluated, 13 concern "ITS." This keyword may be interpreted as an important concept in the user's information need. However, the same keyword was evaluated *wrong* 10 times which can suggest that other attributes of the objects have contributed to this noise.
- The two references published in 1998 were evaluated *ok*.

Our algorithm will use the information on the two attributes and their values to calculate the degree of relevance.

The probability that the user will evaluate a reference as *ok* given the keyword "ITS" is:

$$p(ok/ITS)=\frac{p(ITS/ok)\cdot p(ok)}{p(ITS)}=\frac{3/4\cdot 4/15}{13/15}=\frac{3}{13}=0.231$$

The probability that the user will evaluate a reference as *ok* given the year of publication as 1998 is:

$$p(ok/1998)=\frac{p(1998/ok)\cdot p(ok)}{p(1998)}=\frac{2/4\cdot 4/15}{2/15}=1$$

The degree of relevance of the bibliographic reference analyzed, that is the probability of evaluating it as *ok*, is obtained by the average of all the probabilities of all the attributes being evaluated as *ok*. This is given as:

$$prox(ok)=\frac{p(ok/1998)+p(ok/ITS)}{2}=\frac{1+0.231}{2}=0.615$$

The bibliographic reference in our example will therefore probably be evaluated as *ok*.

The same type of calculation can be done for each type of evaluation. For example, the probability of evaluating the object as *wrong* is:

$$prox(error)=\frac{p(error/1998)+p(error/ITS)}{2}=\frac{0+0.769}{2}=0.385$$

As the user evaluates more objects, the degrees of relevance are updated. Each object will be attached to the highest relevance degree.

In response to a request, the objects will be systematically classified into the most probable type of evaluation. The objects are also sorted in decreasing order of their relevance degree within each class.

From our above example, we will obtain the following relevance degree for the four possible types of evaluation:

prox(ok)=0.615 prox(error)=0.385 prox(known)=prox(bof)=0

In this example, the system will predict that the evaluation of the user will be ***ok*** and therefore classify it into the ***ok* class.**

When the value of an attribute has never been evaluated in the context of the current objective, we use the highest probabilities for each type of evaluation in the other objectives. For example, if the keyword *information filtering* has never been evaluated in the context of the current objective, but has been evaluated in the context of some other objectives, and the highest given degree in these objective is 75%, this highest degree is chosen.

IV. Conclusion

We have presented in this paper an approach for personalizing the responses of an IRS. This approach is based on the employment of an explicit and individual user model. We have also presented the algorithm used to calculate the relevance degree of new solutions based on a user's past objectives and feedback. It should be noted that the system's response is highly personalized in the sense that two users with the same query will not have the same result. Even the same user will have a different result depending on his or her objective and past activities. Our proposals have been implemented in three different applications. One of the applications, METIORE_LORIA, is currently under experimentation.

Most of the users consider the approach of an IRS based on the user's objective to be more natural. This approach corresponds to the user's attitude in front of a librarian, who is for the user a mediator. The user starts by presenting his or her information need and not the system queries. Users may also consider the possibility of exploiting their past history as to accelerate the discovery of solutions to their information needs. This observation is explained by the users as

corresponding to the natural approach towards problem solving. The user explores his or her past cases of information need and starts from the relevant ones. This approach is also similar to that used in Case Based Reasoning in artificial intelligence studies.

The main problem we had was to define an efficient computer algorithm for calculating the relevance degree for possible solutions. This problem was solved mainly by applying our proximity algorithm only to solutions obtained after processing the request by pattern matching. The proximity algorithm is, therefore, not applied to all the objects of the database. The developing language we use for our applications also allows for easy access to the user's history. We have used ProTcl which allows symbolic index arrays.

We are now working on ways to integrate a collaborator's evaluation into the user model in the context of cooperative information retrieval. Another series of experiments is planned for the University of Malaga, which will give us a larger sample of users for the better validation of our proposals.

References

Benaki, E. K., Vangelis A. & Spyropoulos, C. D. (1997). Integrating user modeling into information extraction: The UMIE prototype. In *UM'97*

Brusilovsky, P. (1996). Methods and techniques of adaptive hyperemedia. *UMUAI*

David, A. & Bueno D. (1999). User modeling and cooperative information retrieval in information retrieval systems. *Knowledge Organization, 26(3).* 30-45.

Kobsa, A., Müller, D. & Nill, A. (1996). KN-AHS: An adaptive hypertext client of the user modeling system BGP-MS. *Review of Information Science 1(1).* http://www.inf-wiss.uni-konstanz/RIS

Kononenko, I. (1990). Comparison of inductive and naïve Bayesian learning approaches to automatic knowledge acquisition. In B. Wielinga *et al* (Eds.). *Current trends in knowledge acquisition*. Amsterdam. 190-197.

Logan, B., Steeven, R. & Sparck Jones, K. (1994). Modeling information retrieval agents with belief revision. In *Proceedings of the 17th International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*. Springer-Verlag. 91-100.

Scwab, I. & Pohl, W. (1999). Learning information interest from positive examples. In *UM99.*

Vassileva, J. (1994). A practical architecture for user modeling in a hypermedia-based information system. In *Proceedings of the 4th International Conference on User Modeling*. Cape Cod, MA, 14-19 August. 115-120.

Zukerman, I., Albrecht, D. W. & Nicholson, A. E. (1999). Predicting Users' Request on the WWW.