A Recommendation System in a Collaborative Digital Library Environment

M. Elena Renda and Umberto Straccia

Istituto di Elaborazione della Informazione -C.N.R.
Via G. Moruzzi,1 I-56124 Pisa (PI) ITALY
Number: 2002-TR-06[1]

Abstract

We envisage a Digital Library not only as an information resource where users may submit queries to satisfy their information need, but also as a collaborative working and meeting space. We will present a personalized collaborative Digital Library environment, where users may organise the information space according to their own subjective view, may become aware of each other, exchange information and knowledge with each other, may build communities and may get recommendations. After formalising the main concepts of the personalized collaborative environment, major emphasis is given to the computational model for recommendation generation.

ACM Categories and Subject Descriptors: H.3.2 [**INFORMATION STORAGE AND RETRIEVAL**]: Information Search and Retrieval - *Information filtering*; H.3.4 [**INFORMATION STORAGE AND RETRIEVAL**]: Systems and Software - *User profiles and alert services*;

1 Introduction

It is widely recognised that the internet is growing rapidly in terms of the number of users accessing it, the amount of *Digital Libraries* (DLs) created and accessible through it, and the number of times users use them in order to satisfy their information needs. This has made it increasingly difficult for individuals to control and effectively seek for information among the potentially infinite number of DLs available on the internet.

Typical available DLs provide a search or retrieval service to the web community at large. A common characteristic of most of these retrieval services

[1] E-mail: {renda,straccia}@iei.pi.cnr.it

is that *they do not provide any personalized support to individual users*, or poorly support them. Indeed, they are oriented towards a generic user. In fact, they answer queries crudely rather than, for instance, learn the long-term requirements idiosyncratic to a specific user. In practice, users use the same information resource over and over and would benefit from customization: the time consuming effort that the user put in searching documents and possibly downloading them from the DL is is often forgotten and lost. Later, the user may wish to perform a search about the same topic to find relevant documents that have, *e.g.* appeared since the last time a search was performed.

This requires a repetition of the manual labour in searching and browsing to find the documents just like the first time. Additionally, users are highly interested in being able to organize the information space according to their own subjective perspective (see *e.g.* [11,16]). The requirement of personalized search in the context of DLs is already known and some DLs provide related functionality (see *e.g.* [5,11,14,16,17,20,24,26]). Many of them fall in the category of *alerting services*[2] , *i.e.* services that notify a user (by sending an e-mail), with a list of references to new documents deemed as relevant. Typically all these personalized services are based on the so-called notion of *user profile*, which is a machine representation of the user's information need. It can be acquired either automatically (through the interaction of the user with the system) or set-up manually (by the user). Technically, the acquisition of a user profile and the successive matching of documents against it, in order to *filter out* the relevant ones, is known as *Information Filtering* or *Content-based Filtering* [3,15].

Very seldom, except for few cases like *e.g.* [11], DLs are also considered as *collaborative meeting* or *working places*, where users may become aware of each other, open communication channels, and exchange information and knowledge with each other or with experts. Indeed, usually users access a DL in search of some information. This means that it is quite probable that users may have overlapping interests if the information available in a DL matches their expectations, backgrounds, or motivations. Such users might well profit from each other's knowledge by sharing opinions or experiences or offering advice. Some users might enter into long-term relationships and eventually evolve into a community if only they were to become aware of each other. A service of this kind is important for a DL since it supplies very focussed information. With respect to the information seek task the *recommendation* of items based on preference patterns of others users is probably the most important one. The use of opinions and knowledge of other users to predict the relevance value of items to be recommended to each user in a community is known as *Collaborative Filtering* or *Social Filtering* [4,6,9,18,19,25]. Col-

[2] For instance, many publishing houses provide an alerting service (we omit the long list).

laborative filtering methods are built on the assumption that a good way to find interesting content is to find other users who have similar interests, and then recommend items that those similar users like. In contrast to information filtering methods, collaborative filtering methods do not require any content analysis as they are based on aggregated user ratings of these items.

Both approaches share the common goal of assisting in the users' search for items of interest, and thus attempt to address one of the key research problems of the information age: locating relevant information in a haystack that is growing rapidly. Providing personalized information organisation and search in the context of a collaborative DL environment as additional services to the uniform and generic information search offered today, is likely to be an important step to make relevant information available to people with minimal user effort [1].

The contribution of our paper towards this step is as follows: (*i*) we will formalise a simple collaborative DL environment, where users and communities of users may search, share and organize their information space according to their own personal view; and (*ii*) we will define recommendation algorithms that profitably rely both on personalized information organisation and on the users' opinions. The underlying techniques used for recommendation fall in the afore mentioned categories of content-based filtering methods, collaborative filtering methods and their combination. Additionally to the quite general and self-contained presentation of the environment and the computational model for recommendation generation, we will present an instance of them as the system currently being under development within the EU funded project CYCLADES [3].

The outline of the paper is as follows. In the next section we will formalise the main concepts of our personalized collaborative DL environment. In Section 3 the recommendation methods will be presented, while in Section 4 we present CYCLADES briefly. Finally, Section 5 concludes.

2 A personalized collaborative DL environment

Our personalized collaborative DL environment is made out by several concepts, namely *actors*, *objects* and *functionality*: actors will be able to act on objects by means of the DL's functionality. At first, we will give a brief overview of the environment we have in mind and then move on to its formalisation. Roughly, our collaborative environment is based on the *folder paradigm*, *i.e.* users and communities of users may organise the information space into their

[3] `http://www.ercim.org/cyclades`

own folder hierarchy, as *e.g.* may be done with directories in operating systems, bookmark folders in Web browser and folders in e-mail programs. The idea of organising the information space into folders is not new within DLs. For instance, in [11] users are allowed to define folders of bookmarks (*i.e.* URLs). A folder becomes a holder of information items, which are usually semantically related and, thus, implicitly determines what the folder's topic is about. Therefore, rather than speaking about a user profile, we will deal with a *folder profile*, *i.e.* a representation of what a folder is *about*: the user's set of folder profiles represents the set of topics the user is interested in.

2.1 Actors

In the environment we have two types of *actors*: the set $\mathcal{U}$ of *users* $u \in \mathcal{U}$ and the set $\mathcal{C}$ of *communities* $C \in \mathcal{C}$. A community may be seen as a set of users sharing a common (scientific, professional) background or view of the world. In particular, within our collaborative environment, communities are characterised by a shared interest in the information made available. We postulate that a community $C \in \mathcal{C}$ has a membership function $\mu_C{:}\mathcal{U} \rightarrow \{0, 1\}$, where $\mu_C(u) = 1$ (for ease of presentation we will write $u \in C$) indicates that the user u belongs to the community C. We do not require that a user has to belong to at least one community, *i.e.* we assume that it is a user's choice to join a community or to leave it. A user may belong to different communities as well. It is not our purpose to address the issue of how a community may be created and which are the policies concerning to join and to leave it. We simply assume that there is a *community administrator* (a user $u^C \in \mathcal{U}$) for each community $C \in \mathcal{C}$, who is in charge of defining these policies [4].

2.2 Objects

We will consider three types of *objects*, which may be managed within the environment by users and communities: data items, collections and folders. The objects are organised according to a multilevel model.

Data Items. At the lowest level, we have the set $\mathcal{D}$ of *data items* $d \in \mathcal{D}$. The set $\mathcal{D}$ is the information space and the data items are the information resources that a user is usually interested in discovering or searching for within the DL. The data items may be *e.g.* papers, reports, journals, proceedings,

[4] Similarly, we will not address the issue of becoming a community administrator within the environment.

notes, annotations, discussions, URIs. A data item might also be just a metadata record, which consists of a set of attributes and related values specifying features of a document, according to a specific schema, *e.g.* Dublin Core [12]. The set of data items $\mathcal{D}$ might well be distributed, heterogeneous in content, format and media (video, audio).

Collections. At the next higher level, we allow the data items $d \in \mathcal{D}$ be grouped into *collections*. A collection may be seen as a set of data items, which are grouped together according to some relatedness criteria, *e.g.* the set of data items created within the same year, or those created by the same author, or those about the same topic, say "collaborative digital libraries", or, more obvious, the set of data items belonging to the same digital archive. We assume that there is a set $\mathcal{L}$ of collections $L \in \mathcal{L}$ such that for each collection L there is a membership function $\mu_L: \mathcal{D} \rightarrow \{0, 1\}$, where $\mu_L(d) = 1$ (for ease we will write $d \in L$) indicates that the data item d belongs to the collection L. We also assume that there is at least one collection in $\mathcal{L}$, called *universal collection* and denoted $L_\top$, which includes all the data items $d \in \mathcal{D}$, *i.e.* $\forall d \in \mathcal{D}.d \in L_\top$. Note that a data item may belong to several collections. Furthermore, we do not specify whether the collections are materialised or are just "views" over $\mathcal{D}$. This does not play a significant role in our context. Finally, like for communities, we will assume that for each collection $L \in \mathcal{L}$ there is a *collection administrator* (a user $u^L \in \mathcal{U}$), who is in charge of defining both the collection L and the access policies to it.

Folders. At the third level, we have *folders*. A folder is a container for data items. A folder should be seen as the main environment in which users will carry out their work. Folders may be organised by users according to their own folder hierarchy, *i.e.* a set of hierarchically organised folders, each of which is a repository of the user's selected data items. Each folder typically corresponds to one subject (or discipline, or field) the user is interested in, so that it may be viewed as a thematic repository of data items. In order to accomplish a truly personalized interaction between user and system, this correspondence is implemented in a way which is fully idiosyncratic to the user; this means that *e.g.* a folder named `Knowledge Representation and Reasoning` and owned by user `Tim` will not correspond to any "objective" definition or characterisation of what "knowledge representation and reasoning" is, but will correspond *to what Tim means by* "knowledge representation and reasoning", *i.e.* to his personal view of (or interest in) "knowledge representation and reasoning". As we will see later on, this user-oriented view of folders is realised by learning the "semantics of folders" from the current contents of the folders themselves. We will allow two types of folders: (*i*) *private folders*, *i.e.* a folder owned by a user only. This kind of folder can only be accessed and manipulated by its owner. For other users, they are invisible; and (*ii*) *community folders*, which

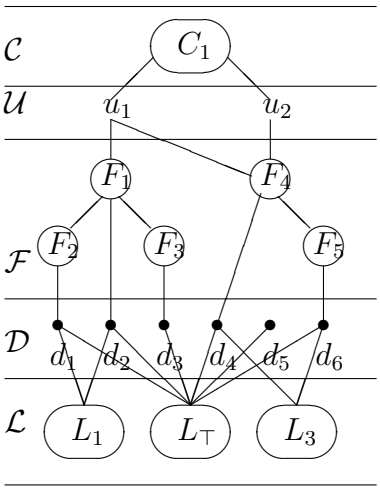

Fig. 1. personalized information space organisation.

can be accessed and manipulated by all members of the community who owns the folder. Community folders are used to share data items with other users and to build up a common folder hierarchy. Community folders may also contain *discussion forums* (a kind of data item) where notes may be exchanged in threaded discussions (similar to news groups). Formally, we assume that there is a set $\mathcal{F}$ of (either private or community) folders $F \in \mathcal{F}$. For each user $u \in \mathcal{U}$, with $\langle \mathcal{F}^u, \preceq^u \rangle$, we indicate the user's folder hierarchy, where $\mathcal{F}^u \subseteq \mathcal{F}$, $\preceq^u$ is a tree-like order on $\mathcal{F}^u$ and with $F^u_\top$ we indicate its *home folder* or *top folder*, *i.e.* the root folder of the hierarchy $\langle \mathcal{F}^u, \preceq^u \rangle$. Furthermore, given a folder $F \in \mathcal{F}$, we assume that (i) there is a membership function $\mu_F{:}\mathcal{U} \rightarrow \{0,1\}$, where $\mu_F(u) = 1$ (for ease $F \in u$) indicates that the folder F belongs to the user's u folder hierarchy, *i.e.* $F \in \mathcal{F}^u$; (ii) there is a membership function $\mu_F{:}\mathcal{C} \rightarrow \{0,1\}$, where $\mu_C(d) = 1$ (for ease $F \in C$) indicates that the folder F is a community folder and belongs to the community C; and (iii) there is a membership function $\mu_F{:}\mathcal{D} \rightarrow \{0,1\}$, where $\mu_F(d) = 1$ (for ease $d \in F$) indicates that the data item d belongs to the folder F. Figure 1 shows an example of community, users and object organisation. In it, users u_1 and u_2 belong to the same community C_1. User u_2 has no private folders, while F_4 and F_5 belong to the same community C_1.

2.3 Actions

A user may perform a certain set of actions, according to whether she is a member of a community or not, and whether she is a collection administrator or a community administrator. The actions are briefly described below. At any time, the user performs her actions with respect to the *current folder*: at the beginning, the user's home folder is the current folder.

Folder management. A user can perform basic folder management actions on the folders she has access to: (*ii*) with respect to "folder hierarchy", folder management operations include creating a new folder as a child of an existing folder, deleting a folder, moving a subfolder from an existing parent folder to a new parent folder (community administrators are allowed to manage the folder hierarchy of a community); and (*ii*) with respect to "folder content", folder management actions include saving data items from a search session in folders (see below), deleting, undeleting and destroying data items, moving and copying data items from one folder to another, rating and annotating data items, downloading and uploading data items.

Collection management. A collection administrator can create, edit, delete and define the access policies of collections. New collections may be defined in terms of others, *e.g.* using meet, join and refinement operators.

Collaborative support. Collaboration between users is supported through the possibility of sharing community folders along with their contents and folder structure. Discussion forums may be created within folders to allow informal exchange of notes and arguments. Rating and annotation of data items also may take the form of discussions among the members of a community. In order not to loose shared activity in the collaborative DL environment, mutual awareness may be supported through event icons (a kind of data item) displayed in the environment. Activity reports that are daily received by email may also be possible. Also, users may view the list of all existing communities so that they become aware of ongoing community activity. This does not mean that they can look inside communities, but only *e.g.* the title, the description and the identity of the community administrator are available. To become a member, users may directly join the community if this is allowed by the community's policy, or may contact the administrator to be invited to the community. In summary, collaboration support concerns with inviting or removing members to or from a community, leaving a community, viewing communities, joining a community (only for communities open to subscription), contacting community managers or other users (*e.g.* via email), creating discussion forums, adding notes to a discussion forum, editing event notification preferences (icons, daily report) and rating data items.

Search data items. The user can issue queries. The result of a query q is an ordered subset (the result list) of data items $d \in \mathcal{D}$. The user is allowed to store selected data items of the result list within her folder hierarchy. In *ad-hoc search* a user u specifies a query q and the action of the system will be to look for relevant data items within a set of user specified folders $F_i \in \mathcal{F}^u$ $(1 \leq i \leq |\mathcal{F}^u|)$ she have access to, *i.e.* to search within $\{d \in \mathcal{D} \colon d \in F_i\}$, or to

search within a specified collection C, *i.e.* to search within $\{d \in \mathcal{D}: d \in C\}$ [5]. We further allow a kind of *filtered search*. Filtered search, is like to the usual ad-hoc search, except that the user u specifies a query q and a folder $F \in u$, and the action of the system will be to look for data items $d \in \mathcal{D}$ such that d is relevant both to the query and to the folder F. Anyway, for both types of search there exists widely known methods. Ad-hoc search is the usual task of information retrieval (see *e.g.* [27]), while filtered search may be accomplished in at least two ways: (i) through techniques of query expansions [8,22], *i.e.* we expand the query q with significant terms of the folder profile f of F and then submit the expanded query; or (ii) we first issue the query q as an ad-hoc query, and then filter the result list with respect to the folder profile [2,3,7,15,23].

Recommendation. A user may get recommendations of *data items*, *collections*, *users*, and *communities*, which are issued to users based on other users' (implicit or explicit) ratings, and on the perceived similarity between the interests of the user, as represented by a given folder, and the interests of these other users, as represented by their folders. All recommendations are specific to a given user folder. This means that the recommendation has always to be understood in the context not of the general interests of the user, but of the specific interests (topic) of the user represented by a folder.

Without doubt, the above set of actions provides us an enhanced personalized collaborative DL environment. Several of the items above are eligible to be the subject of deeper investigations but, in this paper we will put more emphasis to the recommendation issue.

3 Recommendation algorithms

A main feature of our personalized collaborative DL environment is that, (i) by allowing users to organise the information space according to their own subjective view; and (ii) by supporting a simple, yet powerful collaborative environment, it is possible to provide a set of recommendation functionality that, to the best of our knowledge, have not yet been investigated. Indeed, the recommendations regard not only the data items and the collections made available by the DL, but also the users and communities. After a section presenting preliminary definitions, we will describe in detail the recommendation algorithms. In our computational model, we will rely on well known techniques for two reasons: (i) to illustrate the main concepts in the model by means of a concrete example, leaving out unnecessary mathematical complications; and

[5] We do not specify the syntax of queries. This depends on the indexing capabilities of the underlying DL.

(*ii*) this is the computational model currently being implemented in the CYCLADES system.

3.1 Preliminaries

We introduce some notation. For ease of presentation, we will assume that data items are pieces of text (*e.g.* text documents). It is worth noting that our algorithms can be extended to manage data items of different media kind, like audio and video.

By t_k, d_j, and F_i we will denote a text term, a data item, and a folder, respectively. Terms are usually identified either with the words, or with the stems of words, occurring in data items. For ease, following the well-known vector space model [27], a data item d_j is represented as a vector of *weights* $d_j = \langle w_{j1}, \ldots, w_{jm} \rangle$, where $0 \leq w_{jk} \leq 1$ corresponds to the "importance value" that term t_k has in the data item d_j, and m is the total number of unique terms in the indexed universal collection $L_\top$.

The *folder profile* (denoted f_i) for folder F_i is computed as the *centroid* of the data items belonging to F_i; this means that the profile of F_i may be seen as a data item itself [3] (*i.e.* the mean, or prototypical, data item of F_i) and, thus, is represented as vector of weighted terms as well, *i.e.* $f_i = \langle w_{i1}, \ldots, w_{im} \rangle$. The weights w_{ik} of term t_k of the folder profile f_i for folder F_i are then computed as

$$w_{ik} = \frac{1}{|\{d_j \in F_i\}|} \cdot \sum_{\{d_j \in F_i\}} w_{jk} \tag{1}$$

Of course, more complicated approaches for determining the folder profile may be considered as well, *e.g.* taking into account the hierarchical structure of the folders (see, *e.g.* [13]). Conceptually, they do not change much in our algorithm. In order to make the paper self-contained and the formalisation as light as possible, we leave them out and rely on the well-known notion of centroid.

Given a folder F_i, a data item $d_j \in F_i$ and an user $u_k \in \mathcal{U}$ such that $F_i \in u_k$, by $0 \leq r_{ijk} \leq 1$ we denote the *rating* given by user u_k to data item d_j relative to folder F_i [6]. We further assume that whenever a data item d_j belongs to a folder F_i of a user u_k, an *implicit* default rating $\breve{r}$ is assigned. Indeed, the fact that $d_j \in F_i \in \mathcal{F}^{u_k}$ is an implicit indicator of being d_j relevant to folder F_i

[6] Remember that a data item within a community folder, may be accessed (*e.g.* read, annotated and rated) by many different users.

for user u_k. Finally, we average out the ratings given by users relative to the same data item–folder pair, by defining r_{ij} as

$$r_{ij} = \frac{1}{U_{ij}} \cdot \sum_{k=1}^{U_{ij}} r_{ijk}$$

where U_{ij} is the number of users for which the rating r_{ijk} is defined. Note that if F_i is a private folder then $U_{ij} = 1$.

In summary, we may represent (i) the data items as a 2-dimensional matrix, where a row represents a data item d_j and a column represents a term t_k. The value of the cell is the weight w_{jk} of term t_k in the data item d_j; (ii) the folder profiles as a 2-dimensional matrix, where a row represents a folder profile f_i and a column represents a term t_k. The value of the cell is the weight w_{ik} of term t_k in the folder profile f_i; and (iii) the ratings as a 2-dimensional matrix, where a row represents a folder F_i and a column represents a data item d_j. The value of the cell is the rating r_{ij}. The three matrixes are shown in Table 1, where $v = |\mathcal{F}|$ is the number of folders and $n = |L_\top|$ in the number of data items.

We define the *content similarity* of two data items d_1 and d_2 (denoted $CSim(d_1, d_2)$) as the *cosine* of the angle that separates the vectors representing d_1 and d_2, *i.e.*

$$CSim(d_1, d_2) = \frac{\sum_{k=1}^{m} w_{1k} \cdot w_{2k}}{\sqrt{\sum_{k=1}^{m} w_{1k}^2} \cdot \sqrt{\sum_{k=1}^{m} w_{2k}^2}} \tag{2}$$

The smaller the angle, the closer (more similar) the vectors, the closer to 1 is the similarity value. This formula also allows us to determine the content similarity of a data item d_j and a folder profile f_i, since this latter is, mathematically speaking, also a data item. Similarly, the content similarity between two folder profiles f_i and f_h can be determined. Indeed, these similarities are $CSim(f_i, d_j)$ and $CSim(f_i, f_h)$, respectively. Note that the content similarity between two data items, or between a data item and a folder profile, or between two folder profiles is the computation of the similarity among two rows within the matrixes (a) and (b) of Table 1.

Content similarity does not take into account ratings. For accomplish this, we follow a well-known approach for collaborative filtering [6,9,19]. We define the *rating similarity* of two folders F_1 and F_2, written $RSim(F_1, F_2)$, as the *Pearson correlation coefficient* of the ratings given in the folders F_1 and F_2, *i.e.*

$$RSim(F_1, F_2) = \frac{\sum_{d_j \in \mathcal{D}} (r_{1j} - \overline{r}_1) \cdot (r_{2j} - \overline{r}_2)}{\sigma_1 \cdot \sigma_2} \tag{3}$$

	t_1	...	t_k	...	t_m
d_1	w_{11}	...	w_{1k}	...	w_{1m}
d_2	w_{21}	...	w_{2k}	...	w_{2m}
...	...	...	...	...	...
d_j	w_{j1}	...	w_{jk}	...	w_{jm}
...	...	...	...	...	...
d_n	r_{n1}	...	w_{nk}	...	w_{nm}

(a)

	t_1	...	t_k	...	t_m
f_1	w_{11}	...	w_{1k}	...	w_{1m}
f_2	w_{21}	...	w_{2k}	...	w_{2m}
...	...	...	...	...	...
f_i	w_{i1}	...	w_{ik}	...	w_{im}
...	...	...	...	...	...
f_v	w_{v1}	...	w_{vk}	...	w_{vm}

(b)

	d_1	...	d_j	...	d_n
F_1	r_{11}	...	r_{1j}	...	r_{1n}
F_2	r_{21}	...	r_{2j}	...	r_{2n}
...	...	...	...	...	...
F_i	r_{i1}	...	r_{ij}	...	r_{in}
...	...	...	...	...	...
F_v	r_{v1}	...	r_{vj}	...	r_{vn}

(c)

Table 1
(a) The data item matrix. (b) The folder profile matrix. (c) The folder-data item rating matrix.

where $\overline{r}_i$ is the mean of the ratings $r_{i1}, \ldots, r_{in}$, and σ_i is their standard deviation. Note that, similarly to content similarity, rating similarity between two folders is the computation of the similarity between two rows of the matrix (c) in Table 1.

Given the two similarity measures $CSim$ and $RSim$, we may define a new similarity measure, which takes into account both the content and collaborative aspects, as follows. The *similarity* $Sim(F_1, F_2)$ between two folders F_1 and F_2 will be determined as a linear combination between their content similarity and their rating similarity, *i.e.*

$$Sim(F_1, F_2) = \alpha \cdot CSim(f_1, f_2) + \beta \cdot RSim(F_1, F_2) \tag{4}$$

where $\alpha, \beta \geq 0$. From now on, we will simply speak of the similarity of two folders to mean the similarity defined in Equation 4.

We have now all ingredients to define our recommendation algorithms. All recommendation algorithms follow a similar four-step schema described roughly below. In all algorithms that follow, let $F \in u$ be a folder of a user u for which the recommended items should be found. The sketch of the algorithm is as follows:

(1) **Select most similar folders.** At first, select a set of most similar folders F_i to $F \in u$;
(2) **Create a pool of recommendable items.** Second from this set, determine a pool of possible recommendable items;
(3) **Compute recommendation scores.** Third, for each of the items in the pool compute a recommendation score;
(4) **Select items to recommend.** Fourth, select as items to be recommended a subset of items with highest score, and not yet recommended to the target folder $F \in u$.

We proceed with a more detailed description of the above algorithm, specialised for the four cases: recommend users, communities, collections and data items.

3.2 Recommendation of users, communities and collections

The four step algorithms for recommending users, collections and communities are quite similar, so we present them together. For the recommendation of users we have

(1) **Select most similar folders.** Compute the similarity $Sim(F, F_i)$ (according to Equation 4) between the target folder $F \in u$ and each other folder $F_i \in \mathcal{F}$, and rank the folders F_i in descending order according to their similarity value with respect to F. From this rank list, select the top k folders or those having similarity above a certain threshold [7]. Let $MS(F)$ be the set of selected folders.
(2) **Create a pool of recommendable users.** For each folder $F_i \in MS(F)$, consider the users for which the folder F_i belongs to their folder hierarchy, *i.e.* compute the set of users $P_U = \{u' \in \mathcal{U} : \exists F_i . F_i \in MS(F), F_i \in u'\} \setminus \{u\}$, called *pool of possible recommendable users*.
(3) **Compute recommendation scores.** Compute the recommendation score for each possible recommendable user, *i.e.* for each user $u' \in P_U$

[7] The choice between the two methods of folder selection does not play an important role in the algorithm.

determine the *user hits factor* $h(u') = |\{F_i : F_i \in MS(F), F_i \in u'\}|$ (the number of folders F_i judged as similar to the target folder F belonging to user u'). For each user $u' \in P_U$ the *recommendation score* $s(F, u')$ is computed as follow:

$$s(F, u') = h(u') \cdot \sum_{F_i \in MS(F), F_i \in u'} Sim(F, F_i) \tag{5}$$

(4) **Select users to recommend.** Rank the possible recommendable users $u' \in P_U$ in descending order according to their recommendation score value. From this rank list, select the top k users u' or those having score above a certain threshold and recommends those ones not yet recommended, checking against the list of users already recommended to the target folder $F \in u$.

Some comments are in place. In order to determine a set of candidate users (our pool P_U of possible recommendable users) we exploit the similarity measure among folders (Equation 4). The more a folder $F_i \in u'$ is similar to the target folder $F \in u$, the more related, in terms of interests, are the users u' and u. This explains why we first compute the most similar set of folders and then from this set we compute the set of users (P_U), which are the candidates to be recommended. Finally, concerning Equation 5, we would like to note that we borrow an idea taken from [21]. It presents a model to combine into an unique rank the rank lists of several search engines. The model is based on the assumption that the more search engines retrieve the same document, the more this document may be considered as relevant to a query. This assumption can be rephrased in our context as: "the more folders $F_i \in MS(F)$ belong to user u', the more u''s interests are related to u".

Recommendation of communities. The recommendation algorithm of communities is similar to the one for recommending users. Step 1. is similar to above, except that the target folder $F \in u$ is matched against a folder $F_i \in \mathcal{F}$, where F_i has to be a community folder. Once the set of most similar community folders $MS(F)$ has been determined, the *pool of possible recommendable communities* is simply $P_C = \{C \in \mathcal{C} : \exists F_i . F_i \in MS(F), F_i \in C\} \setminus \{C : F \in C, u \in C\}$. Finally, the computation of the recommendation score $s(F, C)$ of a community $C \in P_U$ and the selection of the communities to be recommended is similar to the one for users (the notion of user hits factor is replaced with the notion of *community hits factor*, *i.e.* $h(C) = |\{F_i : F_i \in MS(F), F_i \in C\}|$), and is determined as

$$s(F, C) = h(C) \cdot \sum_{F_i \in MS(F), F_i \in C} Sim(F, F_i) \tag{6}$$

Recommendation of collections. Quite similar to the above two. The only point which is worth to work out is the definition of *pool of possible recommendable collections*, P_L. Once $MS(F)$ has been determined, like in Step 1. above, we compute P_L as $P_L = \{L \in \mathcal{L}: \exists F_i \exists d.F_i \in MS(F), d \in F_i, d \in L\}$. The recommendation score $s(F, L)$ of a collection $L \in P_L$ is based on the *collection hits factor*, *i.e.* $h(L) = |\{d : \exists F_i.F_i \in MS(F), d \in F_i, d \in L\}|$), and is determined as

$$s(F, L) = h(L) \cdot \sum_{d \in L, d \in F_i, F_i \in MS(F),} Sim(F, d) \tag{7}$$

3.3 Recommendation of data items

We conclude this section with the data item recommendation algorithm. This algorithm has much in common with those we have seen above. Additionally, we will exploit the fact that data items are pieces of text and that there might be ratings associated. Step 1. (select most similar folders) and Step 4. (select items to recommend) are as for the "user case", so we omit them. Step 2. (create a pool of recommendable items), is quite similar as seen above: the *pool of possible recommendable data items* is determined by the set of data items belonging to the folders $F_i \in MS(F)$, *i.e.* $P_D = \{d \in \mathcal{D}: \exists F_i.F_i \in MS(F), d \in F_i\} \setminus \{d \in \mathcal{D}: \exists F' \in u, d \in F'\}$ (we do not want to recommend data items already known to the user). Finally, Step 3. (compute recommendation scores) is as follows. The recommendation score for $d_j \in P_D$ with respect to $F \in u$ is computed as a linear combination of a *content-based recommendation score* and a *rating-based recommendation score*. The *content-based recommendation score* of data item $d_j \in P_D$ with respect to the target folder F is (see Equation 2):

$$s^C(F, d_j) = CSim(f, d_j) \tag{8}$$

Note that f is the folder profile of F. The *ratings-based recommendation score* of data item $d_j \in P_D$ with respect to the target folder F is (see Equation 3):

$$s^R(F, d_j) = \overline{r} + \frac{\sum_{F_i \in MS(F)} (r_{ij} - \overline{r}_i) \cdot RSim(f, f_i)}{\sum_{F_i \in MS(F)} \cdot RSim(f, f_i)} \tag{9}$$

where $\overline{r}$ is the mean of the ratings in the target folder F and for $F_i \in MS(F)$, $\overline{r}_i$ is the mean of the ratings of folder F_i, *i.e.* the mean of $\{r_{ik} : 1 \leq k \leq n\}$. Note that if $d_j \notin F_i$, then r_{ij} is not defined and the correspondent term in the sum must be omitted.

For each data item $d \in P_D$ the *recommendation score* $s(F, d)$ is then computed as a linear combination of the recommendation based on content and the recommendation based on ratings:

$$s(F, d) = \alpha \cdot s^C(F, d) + \beta \cdot s^R(F, d) \tag{10}$$

where $\alpha, \beta \geq 0$, and which completes the recommendation algorithms.

4 An application: CYCLADES

The model of a personalized collaborative DL environment we have presented, is currently under implementation in the CYCLADES system [10]. The main goal of CYCLADES is the development of a system, which provides an open collaborative virtual archive environment, which (among others) supports users, communities (and their members) with functionality for (*i*) advanced search in *large, heterogeneous, multidisciplinary digital archives* (*ii*) collaboration; and (*iii*) filtering and recommendation. With respect to the model described in Section 2, a main feature of CYCLADES is that it will use the protocol specified by the Open Archives Initiative [8] (OAI) to harvest and index metadata records from any archive that supports the OAI standard. As a consequence, the set $\mathcal{D}$ of data items includes the set of metadata records harvested from the OAI compliant archives. As a reminder, the OAI is an agreement between several Digital Archives in order to provide interoperability. The specifications give *data providers* (individual archives, ≥ 60 to date) easy-to-implement mechanisms for making the documents' metadata records in their archives externally available. This external availability then makes it possible for *service providers* to build higher levels of functionality. The CYCLADES system is indeed such a service provider. From a logical point of view we may depict the functionality of the CYCLADES system as in Figure 2, Figure 3 shows a mock-up of the user interface, while Figure 4 shows its architecture. It should be noted that from an architecture point of view each box is a Web service distributed over the internet. The CYCLADES system, which will be accessible through Web browsers, provides the user with different environments, according to the actions the user wants to perform.

The *Collaborative Work Service* provides the folder-based environment for managing metadata records, queries, collections, external documents, ratings and annotations. Furthermore, it supports collaboration between CYCLADES users by way of folder sharing in communities.

The *Search and Browse Service* supports the activity of searching records

[8] http://www.openarchives.org

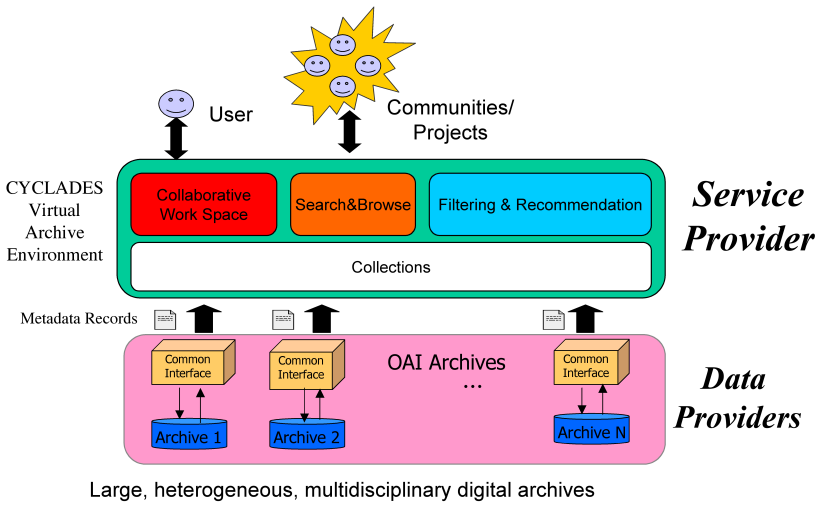

Fig. 2. Logical view of CYCLADES functionality.

from the various collections, formulating and reusing queries, and browsing schemas, attribute values, and metadata records.

The *Access Service* is in charge of interfacing with the underlying metadata archives. In this project, only archives adhering to the OAI specification will be accounted for; however, the system is extensible to other kinds of archives by modifying the Access Service only.

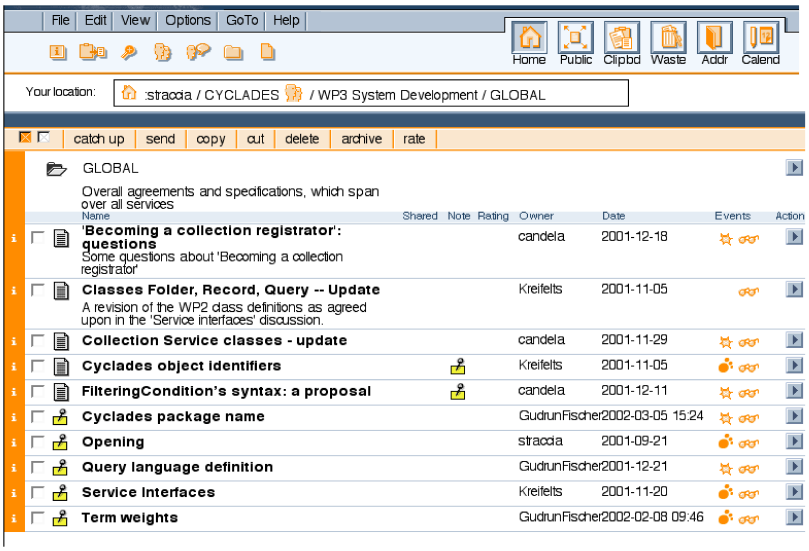

Fig. 3. User interface (mock-up).

The *Collection Service* manages collections (*i.e.* their definition, creation, and update), thus allowing a partitioning of the information space according to the users' interests, and making the individual archives transparent to the user.

The *Filtering and Recommendation Service* provides personalized filtering of queries and query results, provides recommendations of records, collections,

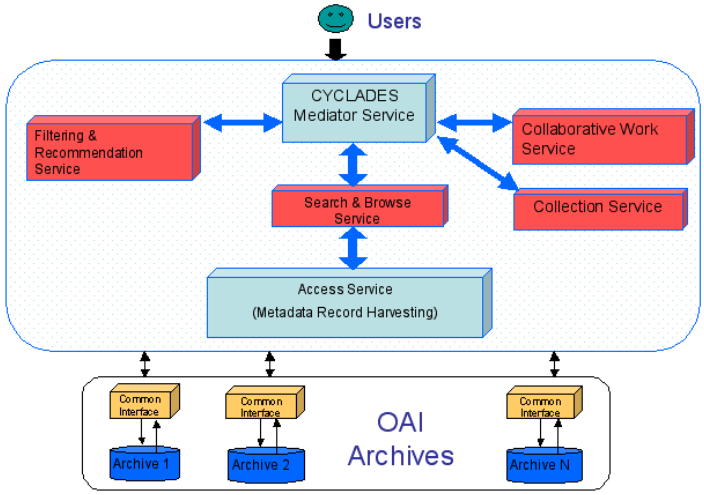

Fig. 4. Architecture.

users, and communities.

The *Mediator Service*, the entry point to the CYCLADES system, acts as a registry for the other services, checks if a user is entitled to use the system, and ensures that the other services are only called after proper authentication.

All of these services interoperate in a distributed environment. Security and system administration will be provided for centrally (by the Mediator Service). The CYCLADES services can run on different machines, and will only need a HTTP connection to communicate and collaborate.

5 Conclusions

We envisage a Digital Library not only as an information resource where users may submit queries to satisfy their information need, but also as a collaborative working and meeting space. Indeed, users looking within an information resource for relevant data might have overlapping interests, which may turn out to be of reciprocal interest for the users: users might well profit from each other's knowledge by sharing opinions and experiences. As such, we have formalised a personalized collaborative Digital Library environment in which the user functionality may be organised into four categories: users may (*i*) search for information; (*ii*) organise the information space (according to the "folder paradigm"); (*iii*) collaborate with other users sharing similar interests; and (*iv*) get recommendations. We also described the CYCLADES system,

which is indeed an on going implementation of the environment. We are aware that many concepts and techniques presented in this paper are eligible to be the subject of further investigations, which we will address in the future.

Acknowledgements

This work is funded by the European Community in the context of the CYCLADES project IST-2000-25456, under the Information Societies Technology programme.

References

[1] Giuseppe Amato and Umberto Straccia. User profile and applications to digital libraries. In *Proceedings of the 3rd European Conference on Research and Advanced Technology for Digital Libraries (ECDL-99)*, number 1696 in Lecture Notes in Computer Science, pages 184–197, Paris, France, 1999. Springer-Verlag.

[2] Marko Balabanovic. An interface fro learning multi-topic user profiles from implicit feedback. In *AAAI-98 Workshop on recommender Systems*, pages –, 1998.

[3] Nicholas J. Belkin and Bruce W. Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–38, 1992.

[4] Daniel Billsus and Michael J. Pazzani. Learning collaborative information filters. In *Proc. 15th International Conf. on Machine Learning*, pages 46–54. Morgan Kaufmann, San Francisco, CA, 1998.

[5] Kurt Bollacker, Steve Lawrence, and C. Lee Giles. A system for automatic personalized tracking of scientific literature on the web. In *Digital Libraries 99 - The Fourth ACM Conference on Digital Libraries*, pages 105–113, New York, 1999. ACM Press.

[6] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52, Madison, Wisconsin, USA, 1998.

[7] Jamie Callan. Learning while filtering documents. In *Proceedings of the 21th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR-98)*, pages 224–231, Melbourne, Australia, 1998.

[8] Claudio Carpineto, Renato De Mori, Giovanni Romano, and Brigitte Bigi. An information-theoretic approach to automatic query expansion. *ACM Transactions on Information Systems*, 19(1):1–27, 2001.

[9] *Berkeley Workshop on Collaborative Filtering*, 1996. `http://www.sims.berkeley.edu/resources/collab/`.

[10] CYCLADES Home Page: `http://www.ercim.org/cyclades/`.

[11] Mariella Di Giacomo, Dan Mahoney, Johan Bollen, Andreas Monroy-Hernandez, and Cesar M. Rouiz Meraz. Mylibrary, a personalization service for digital library environments, 2001.

[12] DublinCore. Dublin core metadata element set:resource page: `http://purl.org/metadata/dublin_core`, WWW.

[13] Susan Dumais and Hao Chen. Hierarchical classification of web content. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR-00)*, pages 256–263, Athens, Greece, 2000.

[14] Daniel Faensen, Lukas Faulstich, Heinz Schweppe, Annika Hinze, and Alexander Steidinger. Hermes: a notification service for digital libraries. In *ACM/IEEE Joint Conference on Digital Libraries*, pages 373–380, 2001.

[15] Christos Faloutsos and Douglas W. Oard. A survey of information retrieval and filtering methods. University of Maryland Technical Report CS-TR-3514, 1995.

[16] Lourdes Fernandez, J. Alfredo Sanchez, and Alberto Garcia. Mibiblio: personal spaces in a digital library universe. In *ACM DL*, pages 232–233, 2000.

[17] Peter W. Foltz and Susan T. Dumais. Personalized information delivery: An analysis of information filtering methods. *Communications of the ACM*, 35(12):51–60, 1992.

[18] David J. Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[19] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR-99)*, pages 230–237, Berkeley, CA USA, 1999.

[20] *Information Filtering Resources*. `http://www.enee.umd.edu/medlab/filter/filter.html`.

[21] Joon Ho Lee. Analysis of multiple evidence combination. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR-97)*, pages 267–276, Philadelphia,PA, July 1997.

[22] Mandar Mitra, Amit Singhal, and Chris Buckley. Improving automatic query expansion. In *Proceedings of the 21th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR-98)*, pages 206–214, Melbourne, Australia, 1998.

[23] J. Mostafa, S. Mukhopadhyay, W. Lam, and M. Palakal. A multilevel approach to intelligent information filtering: Model, system, and evaluation. *ACM Transactions on Information Systems*, 15(4):368–399, 1997.

[24] Alexandros Moukas. *Amalthaea*: Information discovery and filtering using a multiagent evolving ecosystem. In *Proceedings Practical Applications of Agents and Multiagent Technology*, London, GB, 1996.

[25] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, North Carolina, 1994. ACM.

[26] Luis Mateus Rocha. Talkmine and the adaptive recommendation project. In *ACM DL*, pages 242–243, 1999.

[27] Gerard Salton and J. Michael McGill. *Introduction to Modern Information Retrieval*. Addison Wesley Publ. Co., Reading, Massachussetts, 1983.