Abstract— The CS OER Portal was created with the goal of improving the findability and increasing the use of open educational resources in Computer Science. In this paper we describe the portal’s design principles which exploit the domain specificity of the hosted content and are focused on enhancing the search and navigation in the portal. Distinctive retrieval features are the proposed topical recommendation and query-by-navigation. The latter allows users to formulate queries by walking through a directory-like structure based on the ACM/IEEE CS Curriculum Recommendation.

Keywords-OER; open educational resources; information retrieval; search; computer science education

I. INTRODUCTION

The open educational resources (OER) movement is changing the traditional teaching and learning dynamics. It is backed by the view that educational resources should be shared and reused. Instructors have been sharing their materials with colleagues but what is new is the way these resources are produced and the legal framework that allows their reuse, sharing and distribution. As a result, there is a considerable amount of open, publicly available content today however a big portion of it is underused. Due to the distribution of the open resources across many institutional repositories and individual sites their findability is one of the major barriers for the OER large-scale uptake and reuse [4, 5]. Substantial research has been dedicated to this subject (e.g. [1, 2]) and some OER reference repositories, such as OER Commons, have been built [3], but satisfactory results have not yet been attained.

The OER movement can especially benefit Computer Science (CS) education. The rapidly evolving body of CS knowledge forces a continuous re-evaluation of the topics covered in CS courses. This requires both faculty and students to constantly update their knowledge and skills to keep pace with the evolving discipline. OER can aid this continuous process by enabling the sharing of knowledge and experience, and reducing the duplication of efforts for creating educational resources. These two considerations motivated us to develop the CS OER Portal with the goal to increase and ease the access to Computer Science open resources. To inform the design of the portal we conducted two large-scale studies. The first study was aimed at better understanding the Computer Science instructors’ needs for open educational resources [5]. The second one was aimed at obtaining empirical evidence on how instructors typically discover OERs [6].

Among the main features of the created OER reference repository which together distinguish it from the other educational reference repositories, are: (1) It follows a subject-centered model which allows mirroring some computing curricular principles within the search/navigation mechanisms; (2) It provides multiple mechanisms for search and navigation enhanced with filtering capabilities; (3) It contains only links to open educational materials; (4) It provides support for sharing, distributing and receiving updates for CS OER of interest. In particular, we developed two approaches to search and navigate in the CS OER Portal that are novel in the context of educational repositories – topical recommendation and query-by-navigation.

II. THE CS OER REPOSITORY

Our preliminary studies indicated that the dominating reason for the low OER use is the insufficient support for finding needed open content [5, 6]. The problem with findability lies in both its visibility and the employed search and browsing tools. These factors shaped our view on the portal functionality.

Current OER repositories are predominantly institution-based [3]. An institutional alignment of OER content relates to the institutional goals. However, a provider-centered system may not be optimal for learners and educators, given the fact that each university has its own academic structure. Our studies confirmed the need for a centralized platform where learning content from different hosts can be organized uniformly. This is in line with other information seeking behavior studies showing that users prefer going to one centralized site rather than visiting and piecing together data from individual sites [3]. On the other hand, as reported in [7], almost 60% of the surveyed participants have expressed a preference to a subject-based repository.

Many educational repositories use a topic hierarchy as a navigation mechanism. However, there is no standard Computer Science topical hierarchy, so we took the viewpoint of Ochoa and Duval, who confirmed the feasibility of organizing resources in courses, since repositories featuring full courses have a more active user base than repositories containing resources of smaller granularity [8]. A course development often involves borrowing content from different sources and adapting it based on the instructor’s own experience and instructional objectives. To support this process the portal hosts links to materials not only on a course level but also on a resource level (e.g. lecture notes, tests, labs). For structuring the
content in a predictable way the top level categorization is based on the original institutional/individual collections' structure, constrained to CS and IT courses. For each institution we keep the original segmentation of the content:

Collections/Courses/[Sections]/Topics/Resources.

An additional argument in support of this institution-course division is that most repositories feature full courses and shape certain browsing behavior. It preserves the access to the resources in their original course packaging (see Fig. 1), while the “disaggregation” of the packages allows regrouping of resources for different types of faceted search. The three guiding criteria for categorizing resources were:

- Predictability: organize resources in predictable groups;
- Discoverability: provide multiple ways for locating resources;
- Typicality: provide support for typical searches.

Figure 1. A screenshot from the CS OER Portal.

The CS OER Portal currently contains links to more than 9,000 resources and 775 courses coming from 54 institutional and 71 individual collections.

III. Search and Navigation

We provide two types of support for finding content. The first one aims to bring users to the repository, while the second one starts after their landing there. For the first type we implemented (white-hat) search engine optimization strategies. With regard to the local search, our focus was on:

(i) utilizing the subject-orientation to improve the precision and relevancy of the search;
(ii) creating an environment where search and navigation complement each other;
(iii) provide several navigational options to stimulate exploration of the content.

In order to improve the repository search and navigation we employed a multi-strategy approach aimed to:

- offer an interface where searching and browsing operate in a single flow of interaction;
- utilize identified search patterns for bringing the result of an anticipated search before the search actually occurs (preemptive contextualization [9]).

The implemented multi-strategy approach includes: navigation (structural and tag-based), search (standard, faceted, and advanced text search), topical recommendation, and a ‘query-by-navigation’ search. We describe the last two approaches in more detail below.

A. Topical Recommendation of Courses

A frequent OER search task is looking for courses that match a particular topical structure. The major complication in such a task derives from the fact that there is no simple way to define an unambiguous naming convention. Similar courses may have different names. Also, under a particular course name, the course content can vary depending on many factors including the course objectives, the length of the semester, etc. Apart from the course names variation, the desired course content depends on the view of the searcher. Thus finding a useful course involves mapping an intended structure of topics, occurring in the mind of the searcher, to a course from the set of courses in the repository. One of the objectives of this work was to simplify this task by reducing it to a comparison of two relevant metadata sets. Our approach rests on the assumption that users who look for courses are guided by a certain descriptive topical framework in their minds. Instances of such framework appear in course documents under terms, such as “course outline”, “topics covered”, and “schedule”. A table of content of a textbook is yet another example. With this assumption in mind, we consider two courses as matching if they cover a similar set of topics. Accordingly, in the CS OER Portal all courses are represented by descriptive metadata including the course name, a list of topics, and a set of tags.

Assume that we have a native search engine providing a search box with sufficient space, so that a user can copy a course title along with the course topics (e.g. from a syllabus of a previously taught course) and submit it as a search query. The expectation is for the search engine to retrieve all courses matching this query and to display the list of links to the corresponding documents, ranked based on their similarity to the input. Unfortunately, a standard keyword search cannot be used for implementing this scenario. It is based on the Boolean query model, where the indicator of relevance is whether or not all terms in the query occur in the document. The “OR” option, allows retrieving documents that contain at least one of the query terms. In addition, long queries are problematic for the conventional search tools and certain limits on the query length are typically imposed. Even after removing the stopwords from such a query composed of a title and course topics, the corresponding text cannot be fed directly into most existing CMS native search tools.

So, we need a task-specific search, where the number of found query terms is a significant factor in ranking the documents, without requiring them to contain all query terms. Thus, instead of the Boolean queries we use the TF (Term Frequency) representation of the vector space model. The proposed search incorporates the following steps. Each course is presented with its topical description, containing the course title, list of topics, and list of tags. The terms in the descriptions are extended with their synonyms. For a set of n courses this results in n descriptive documents. After their preprocessing with standard linguistic methods, such as stemming and removing stop words, the set of documents is transformed into a document-term matrix T where each
row is a document and each column is a term. In the TF approach, the coordinates $w_{ij}$ in the matrix are a function of the number of occurrences of word $j$ in document $i$ (normalized with the document length). The query is presented in the same vector space for computing the vectors proximity (documents' similarity). As a similarity measure we use cosine similarity. Thus, all term vectors are compared with the query vector and ranked in the descending order of their similarity.

The similarity metric used in our approach is not new, though we had to overcome some challenges, for example, maintaining separate computational strategies for textual and metadata content, avoiding the use of the traditional TF-IDF measure, etc. What is new is the query formulation. The challenge in searching is to find appropriate keywords and our approach removes this challenge by using the topical description of a course of interest as a search query. We used this method in the portal for recommending topically similar courses. There are other possible uses of the proposed method, such as finding course materials matching a syllabus from a past semester or a set of topics central to a program.

B. Query by Navigation

If used within a framework with suitable organization, the above method can ease the search for relevant courses by replacing the classic search with a query by navigation [10]. The idea was to create a directory-like interface for query formulation mirroring the knowledge structure of the CS Curricula. Although there is not such a standard, an agreed upon set of topics is emerging with the maturation of Computer Science. The ACM/IEEE CS Curriculum Recommendation is a community effort intended to tailor CS curriculum to the changing landscape of computing. The identified body of knowledge for CS Undergraduate Programs is topically organized in a set of “Knowledge Areas”, each of which provides a list of learning outcomes and is structured in units and topics. Thus the ACM/IEEE CS Curriculum Recommendation was chosen to provide a structural framework for our directory-like interface.

Assume that we have stored the knowledge areas/topics hierarchy as defined by the ACM/IEEE Curriculum as a searchable structure in the repository and use it as an interface. This interface would allow navigation up and down the curriculum structure until the user recognizes a topical description that is a satisfying formulation of their query. Note that with this approach users can find resources of interest hidden under alternative course names, since the relevance is measured based on the proportion of matching topics rather than on similarity of titles only (for example, topics found in “Human-Computer Interaction”, may be listed under “User Interface Design”). This approach eliminates the need of entering queries and improves the relevance and ranking of search results.

The realization of the above approach resulted in an interface, which is a combination of navigation and search and looks similar to that of a link directory. However, the clickable links incorporate a combined semantics of a directory and a search engine input. When a user activates a link, it opens the next level of the hierarchy showing the corresponding body of topics along with sending that body as a query to the search engine. This way we can capitalize on users’ familiarity with directories and relieve them from guessing the right search terms associated with the notorious search box. The implementation of this approach required the creation of an auxiliary document collection, mapping the ACM/IEEE Curriculum Recommendations’ structure. Each document in this collection contains the name of the corresponding curriculum knowledge structure along with its body and is accessible through the name which appears in the document’s parent page.

IV. Conclusion

The concept of findability implies locating both the repository and the relevant content inside the repository. The focus of this work is on the second task which has also two sides: support for navigation and for search. We take an approach aiming to facilitate typical common search and navigation tasks. Accordingly, the portal provides four search/navigation mechanisms. The navigation challenge is addressed by organizing resources in predictable structures, by preemptive contextualization, and by providing query-by-navigation interface. The quality of the search depends on both the implemented search strategies and the search terms. The support for internal search was addressed by utilizing a domain specific vocabulary within the employed search mechanisms. The challenge of what terms to use was addressed by replacing the search box with a standard CS topical structure driving the proposed query-by-navigation.

ACKNOWLEDGMENT

This material is based upon work supported by the NSF Grant DUE-1044224 “NSDL: Social Bookmarking for Digital Libraries: Improving Resource Sharing and Discoverability”.

REFERENCES