A Search Engine to Categorize LOs

Gianni Fenu

***Abstract*—The main purpose of this paper is to analyze the state of the art of search engines in e-learning platforms, and to elaborate a new model that exploits its best suggestions to perform an efficient and precise search. The algorithms, architecture and system use of this model are discussed, presenting a global vision of the search component.**

***Index Terms*—E-learning, search engine, pre-query, post-query, ranking metrics, search modules.**

I. Introduction

In the last decade, the e-learning technology grew and developed notably. This new way of teaching, made of online courses and multimedia material, exploits the potential of the Internet to supply a personalized and interactive learning, and is placing side by side the traditional one.

The whole e-learning system is built on a platform called Learning Management System (LMS), which is installed on a server and manages the contents through a Learning Content Management System (LCMS). Every elementary didactic unit is a Learning Object (LO): text documents, multimedia contents and audio/video streaming are examples of it. It is the basic component that allows the student to learn.

When a LMS platform starts increasing and gaining importance, the existence of a precise and efficient search engine becomes necessary. The system must be able to provide, after a user's query, the most significant LOs about a certain argument. This is not only a form of help to the user, but also an interdisciplinary instrument: the search connects contents of different subjects, making the student's culture richer.

This work will analyze the existing search engines in Section II (State of the Art), and will develop the structure of a new one, discussing its algorithms in Section III (Methodology) and its architecture in Section IV (Architecture). Section V (System Use) presents a global vision of the system, and Section VI (Conclusions and Future Work) concludes the paper.

II. State of the Art

In order to make an efficient search, several operations must be performed. Some of them are needed to organize the LOs, so they must be executed before the user's query (we can call them *pre-query operations*); some others carry out the search on these LOs, and are executed after the query (*post-query operations*).

We will now analyze these operations, as they have been proposed by different authors, pointing out various principles and summing up the state of the art.

The pre-query operations arrange and catalogue the LOs in a data structure that allows an easy and quick search for them.

The first thing to do is to create semantics of the e-learning domain. As [1] suggests, the ideal is to use a tree structure based on the resources hierarchy: in this way, LOs will be in a kinship. It is also possible to use the reusability [2] trees described by [3] to register as child nodes LOs obtained modifying others.

Once the domain semantics is created, a weight to each object must be assigned. [3] and [4] proposed some weight metrics based both on metadata inserted by the authors (information about the file) and on citation of users during the time (e.g. download frequency). Combining these values, it is possible to compute the importance degree of a LO [5].
On the objects that are text documents, [6] and [7] suggest a series of operations turned to extract the most significant terms: the phases of tokenization, stop words filtration and stemming. The n terms with the highest frequencies are selected and added to the LOs metadata.

[1], [7] and [8] consider fundamental a clusterization of the LOs in macro-groups. Some specific clustering algorithms exist that measure similarities among different objects and group them into categories.

Lastly, it is useful to trace a profile of the registered users, noting down interests and consulted documents, so that the sorting of the search results will be personalized. [1] proposes to create this profile with a "Bottom-up Pruning" algorithm, that selects the visited LOs from the e-learning tree; [8] suggests to generate the semantics as an ontology [9]. [1] uses than these information to find a *recommended cluster* for the user. Also, in [10]'s opinion, user's performance must be registered in the profile; in order to supply additional material and strengthen the study program in case it is needed.

The post-query operations have to look for the LOs that most satisfy the query, and sort them by significance.

Before the search, [11] proposes a pre-processing of the query similar to that one made on text documents, in order to extract the terms and validate them in the dictionary. Also, synonyms of the words can be implicitly added to the query or suggested to the user [3], in order to amplify the search. [6] Suggests weighing the keywords on their appearance order, in order to give more importance to the firsts.

This modified query must be matched with the LOs; all the documents that have keywords in common with it must be selected, and then sorted by significance. This last operation is called re-ranking [12] and, according to [3], has to be based on different factors: similarity with the query, number of occurrences, weight of the LO, interests in the user profile and recommended cluster. The ordered results are finally

Manuscript received March 17, 2013; revised April 25, 2013.

Gianni Fenu is with the Computer Science at University of Cagliari, Italy (e-mail: fenu@unica.it)

DOI: 10.7763/IJFCC.2013.V2.250

returned to the user that carried out the search [13].

The user's profile must now be updated, in order to trace an evolution of it [8]. [7] also proposes to monitor it periodically, to detect possible shifts of interests.

Finally, in order to test the quality of the search engine, it is possible to follow the suggestion of [3], consisting in asking to the user some feedback about the relevance and precision of the search.

III. Methodology

Analyzed this state of the art, we exploit now its best suggestions to elaborate an algorithmic procedure that will perform the search. We distinguish between the cataloguing operations and the search of LOs.

A. Pre-Query Operation

The first step in the pre-query phase consists of cataloguing the LOs in a ordered structure. To represent the domain, we use a tree like that in Fig. 1.

The college is the root, so there is a tree for each different university. Inside the college, we distinguish between different disciplinary fields, that are composed of one or more courses. Every course supplies didactic material, the LOs [14]. Usually a LO is a leaf node of the tree, but it can also have children if there are other objects obtained deriving or modifying it.

To assign a weight to the LOs, we make use of the following metrics:

- A just inserted object has weight 0
- The more a document is downloaded, the more it weights
- The more time passes without a download, the more the weight decreases
- at the same weight, a parent weighs more than its child

The cases *a* and *d* are assigned at the beginning, whereas the other two must be periodically updated in the platform.

So, it is necessary to establish the following weights:

- α: weight increased after a download
- β: weight decreased after the going over of a temporal limit t
- γ: how much a parent weighs compared to its child
- *t*: temporal value after which a LO starts to lose weight

Every learning document is characterized by some keywords, terms inserted by the author as metadata [15] and considered representative of the content.

In the case of text documents, the keywords list can be extended adding the *n* most significant terms of the content. This process is called *text segmentation* and it is composed of the following phases:

- Tokenization: all the adjacent strings of alphanumerical characters, called tokens, are extracted from the document
- Stop words extraction: all the words that are not significant for the document content are deleted
- Stemming: the terms are reduced to their root form

An example of application of this process is shown in Fig. 2. After it, the most recurring terms must be added to the LO keywords.

We represent in the domain tree disciplinary fields and courses; however, there would be a better indexing grouping the LOs in some macro-arguments inside the courses too. We use clusters for this.

We create them manually rather than automatically: an expert of the disciplinary field (a professor) is able to identify some macro-arguments in its subject better than those that would find an algorithm based on keywords extraction.

So, once created the clusters, the LOs must be catalogued among them. We propose an algorithm based on a comparison between keywords, that assigns a document to the cluster with which it finds more similarities.

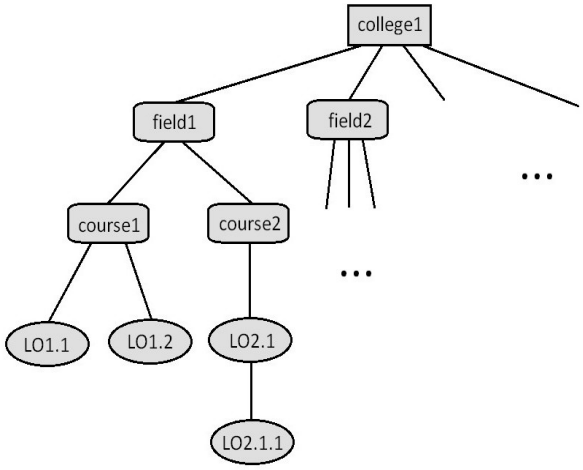

Fig. 1. The domain tree

Every user that is registered in the platform has a profile containing the list of the documents that he consulted or downloaded and a list of weighed keywords. The purpose of the profile is to mark the user's interests, in order to be able to consider them during the re-ranking of the query results [12].

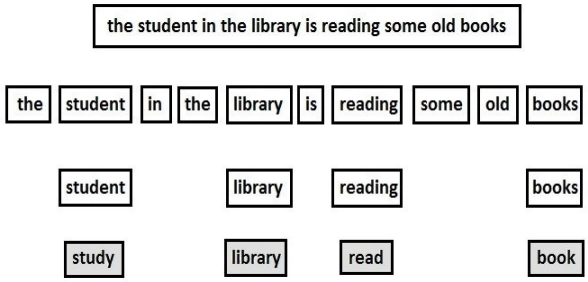

Fig. 2. The process of text segmentation

The list of the user's interests is updated with the metadata of the consulted LOs after the search. Once created it, the clusterization algorithm can be applied to assign a recommended cluster to the user. During the post-query re-ranking, the objects that belong to that cluster will weigh more [16], and this will personalize further on the search.

If the e-learning system registers the grades in single courses too, the student's performance can be measured in the profile. If these performance result inferior to the average, it is possible to give emphasis to material considered integrative by the professor. The functioning is described in Fig. 3.

B. Post-Query

The phases up to now analyze organized the environment in which to execute the searches. Now we must focus on the real search, made on the user's queries in the constructed environment.

Before examining the objects, the query must be processed in a way similar to the one applied to the text documents: the phases of stop words elimination and stemming.

To check if the user wrote correctly the terms, the obtained words must be validated in a dictionary. There, synonyms of the terms can be found too, and added to the keywords of the query: the search range will be wider.

In the end, following the suggestion of [3], we assign to the keywords a weight $w(i)$, where i is the position of the word among n terms, counting from the last one: in this manner a term will have much more emphasis the earlier it has been inserted in the query. An example of application of the query pre-processing is shown in Fig. 4.

The matching phase is simply a comparison between the keywords extracted from the query and the keywords that characterize every LO.

Every time that a document has got at least one correspondence with the query, it is selected. In this way, we get a not-ordered list of results.

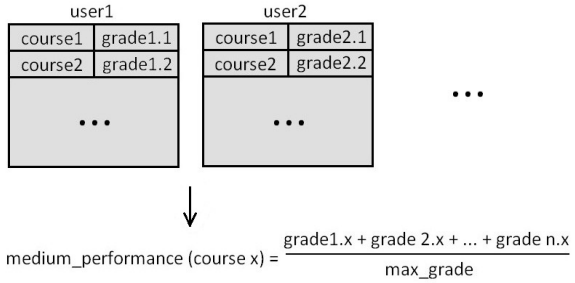

Fig. 3. Operations on performances

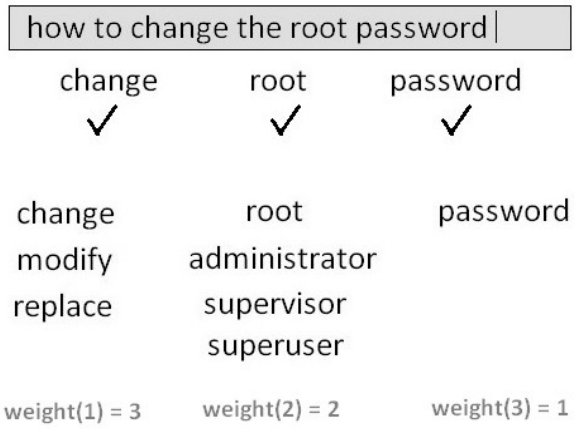

Fig. 4. Query pre-processing

The list of results returned from the matching phase must be ordered by relevance. This operation is called *re-ranking*.

We want to order considering the following metrics:

Number of similarities between the keywords of the query and those of the object

Weight of the query keywords

Kinship among the LOs

Weight of the Los

Interests in the user profile

Belonging of the LO to the recommended cluster

Let us use the following parameters:

count_query: integer number that counts how many of the n query words are present in the document

count_occ: number that counts the occurrences of query words in the document, considering their weight based on the order in the query

LO_weight: field present in every LO that shows its weight considering kinship and number of downloads

usr_intrs: variable that indicates the percentage of similarities among the LOs keywords and those in the user profile

rec_cluster: variable that is equal to 1 if the LO belongs to the recommended cluster; 0 otherwise.

Most variables can be easily calculated. About *count_occ*, let us suppose that the query term has a weight w converted in a range between 0 and 1, and that every occurrence *occ* of the term in the document weighs 1. Now we perform the search for all terms and compute *count_occ* as $\sum_{terms}(occ \cdot v)$

urs_intrs, instead, is a variable between 0 and α, where α can be 1 or a different number depending on the importance that one wants to give to the user profile.

Connecting all these parameters, we compute a new weight for each object:

$$ranking\_weight = (count\_query \cdot count\_occ) + \\ + (\alpha \cdot LO\_weight) + (\beta \cdot usr\_inrs) + (\gamma \cdot rec\_cluster) \quad (1)$$

where α, β and γ can change depending on how much weight one wants to give to each parameter.

Once calculated the ranking weight for each LO, we use a generic sorting algorithm to order the results from the highest to the lowest, and the resulting list is returned.

At the end of the search, the metadata of the consulted LOs must be added to the user profile.

To trace an evolution of it and detect possible shifts of interest, it is possible to count in a variable the time spent from inserting the keyword in the profile, and decrease by a factor of α keyword weight every time that a certain time limit is reached.

To analyze the efficiency of the search engine, it is possible to ask for feedback from the users.

We determine two kinds of feedback:

Most significant result: the user is asked to indicate the most significant result, and the weight of this result is updated adding a factor of α

Quality of the result: the user is asked to give a positive or negative opinion about the given results; in case of negative the weight of the first n returned results is decreased by a factor of β

Conclusions can be taken also computing the *top-n recall* and *top-n precision* parameters, defined in the following way:

$$top-n\ recall = \frac{number\ of\ relevant\ documents\ in\ the\ firsts\ n\ results}{total\ number\ of\ relevant\ documents} \quad (2)$$

$$top\_n\ precision = \frac{number\ of\ relevant\ documents\ in\ the\ firsts\ n\ results}{n} \quad (3)$$

IV. Architecture

The LMS platform in which to execute the search is supposed to be already implemented. It can have different architectural styles; the most common for the e-learning systems are repositories, the client-server architecture and the n-tier architecture.

The chosen architecture is composed by different modules. Their choice depends on the author of the platform, but some of them can be found in every e-learning system: the database, the Learning Content Management System, the LMS, the web server and the user interface.

We implement the search engine module as a component that belongs both to the LMS and the LCMS. It is made of three modules, as shown in Fig. 5: the LOs cataloguing, which is a part of the LCMS because needs to interact with the database, the effective search engine and the user profile, that belong to the LMS.

The cataloguing module must organize the LOs in the database: it creates the domain tree, assigns a weight to the objects, performs the text segmentation and the clusterization. It should be independent and separate from the others, and it should come into action after every alteration of the domain.

The user profile module is independent too. It contains personal data and marks of the exams, and is connected to several other modules of the system. The search engine module exploits it adding interests, inserted queries and the recommended cluster, and use them in the re-ranking phase.

In the end, the search engine module is the most important component for the search. It processes the query and then enters in the database (which has been ordered by the cataloguing module) through the LCMS. The results are ordered depending on LOs weights and contents in the user profile, and the profile is evolved.

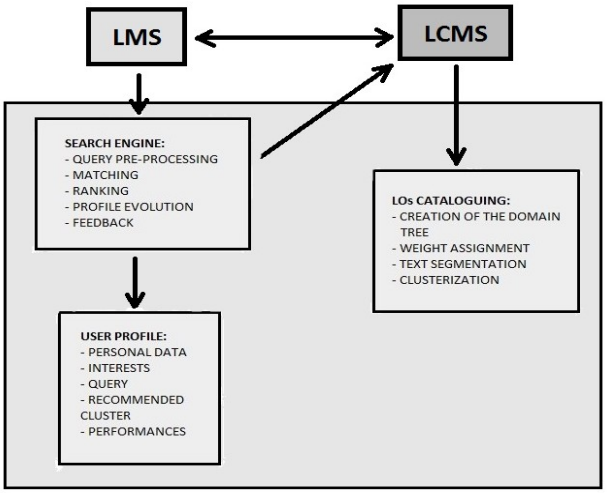

Fig. 5. The search engine architecture

Most of the steps of these three modules are hidden to the user, because of the information hiding principle. The student can consult his profile, the LOs and the result of the search, but all the implementation details are not showed to him.

V. System Use

Fig. 6 summarizes the steps of our article. We now make an example of how our search engine altogether works, showing the interactions between the different components. We have a platform with some catalogued e-learning material, and a list of users registered on it; each one of them has its own profile.

Let us consider a user that writes a query in the search engine bar, through a specific interface. Our system receives it as a list of terms, and processes them extracting the significant words, checking their correctness, adding synonyms and weighting them.

The result is a set of keywords, that must be matched with our whole e-learning system. We organized colleges, courses and didactic material in a tree, and gave to each LO a weight, a list of keywords and a cluster. The task of the system now is to browse these objects and to select those that have keywords in common with the terms processed from the query.

These are our first, partial results. Now they must be re-ranked, i.e. sorted in order to give more importance to the most relevant and significant ones. Our re-ranking algorithm is based on five parameters: similarities between the LO and the query, weight of the object, kinship of objects, pertinence with the user profile and belonging to the recommended cluster.

What we get now is a ordered list of results, that is returned to the user.

In the end, we will update the user profile and ask for feedback that will in case re-distribute the LOs weights.

From the user's point of view, the e-learning platform received his query and returned to him the list of the most relevant, related LOs.

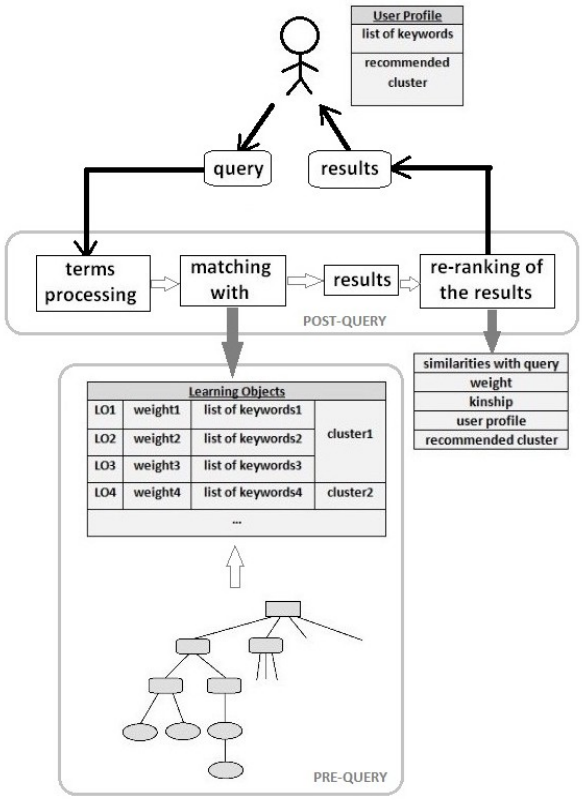

Fig. 6. The whole search engine process

VI. Conclusion and Future Works

We have described the phases and architectures of a search engine whose purpose is to return precise and relevant results in e-learning systems.

The "UnitelSardegna" Consortium, in collaboration with the University of Cagliari (Italy), is actually developing an e-learning system that supplies online distance courses and promotes formation activities and long life learning courses. The existent platform will be integrated soon with the search engine that we are creating, based on the principles discussed in this paper. The purpose is to improve the search that in the e-learning platforms is usually wanting in precision and details.

The next step of our research is to implement it, in order to analyze the quality of the results and to compare it with the performance of other search engine systems. Future progresses will be treated in our next articles.

References

[1] L. Zhuhadar, O. Nasraoui, R. Wyatt, and E. Romero, "Automated Discovery, Categorization and Retrieval of Personalized Semantically Enriched E-learning Resources," in *Proc. IEEE International Conference on Semantic Computing*, 2009

[2] T. K. Shih, C. C. Chang, and H. W. Lin, "Reusability on Learning Object Repository," in *Proc. of the 5th Int. Conf. on Web-based Learning*, pp. 203-214, 2006

[3] N. Y. Ten, T. K. Shih, L. R. Chao, and Q. Jin, "Ranking Metrics and Search Guidance for Learning Object Repository," *IEEE Transactions on Learning* Technologies, 2010

[4] H.W. Lin, M. T. Tzou, T. K.Shih, C. C. Wang, and L. C. Lin, "Metadata Wizard Design for Searchable and Reusable Repository," in *Proc. of Int. Conf. on SCORM*, 2006

[5] J. R. Hilera, S. Oton, A. Ortiz, L. D. Marcos, J. J. Martinez, J. A. Gutierrez, J. M. Gutierrez, and R. Barchino, "Evaluating Simple Query Interface Compliance in Public Repositories," in *Proc. of the 9th IEEE Int. Conf. on Advanced Learning Technologies*, pp. 306-310, 2009

[6] Y. Anistyasari and R. Sarno, "Weighted Ontology for Subject Search in Learning Content Management System," in *Proc. International Conference on Electrical Engineering and Informatics*, 2011

[7] J. C. Prates and S. S. M. Siqueira, "Using educational resources to improve the efficiency of the Web searches for additional learning material," in *Proc. of 11th IEEE International Conference on Advanced Learning Technologies*, 2011

[8] O. Nasraoui and L. Zhuhadar, "Improving Recall and Precision of a Personalized Semantic Search Engine for E-learning," in *Proc. of Fourth International Conference on Digital Society*, 2010

[9] M. Roya, R .Chang, and X. Qi, "Learning From Relevance Feedback Sessions Using A K-Nearest-Neighbor-Based Semantic Repository," *IEEE Int. Conf. on Multimedia and Expo*, pp. 1994-1997, 2007

[10] D. Celik, A. Elci, and E. Elverici, "Finding Suitable Course Material through a Semantic Search Agent for Learning Management Systems of Distance Education," in *Proc. of 35th IEEE Annual Computer Software and Applications Conference Workshops*, 2011

[11] A. N. Segura, M. M. Prieto, and C. C. Vidal, "Query Expansion based on Domain Ontology for Learning Objects Search," 2010

[12] X. Ochoa and E. Duval,; "Relevance Ranking Metrics for Learning Object," *IEEE Tran. on Learning Technologies*, vol. 1, no. 1, 2008

[13] V. Raykar, R. Duraiswami, and B. Krishnapuram, "A Fast Algorithm for Learning Large Scale Preference Relations," in *Proc. 11th Int'l Conf. Artificial Intelligence and Statistics (AISTATS '07)*, vol. 2, pp. 388-395, 2007

[14] A. E. Saddik, S. Fischer, and R. Steinmetz, "Reusable Multimedia Content in Web-Based Learning Systems," *IEEE Multimedia*, vol. 8, no. 3, pp. 30-38, 2001

[15] M. Kastner and G. Furtmüller, "Operationalization of the Metadata Element "Difficulty"," in *Proc. of the 7th IEEE Int. Conf. on Advanced Learning Technologies*, pp. 608-612, 2007

[16] N. Y. Yen and L. R. Chao, "Re-Ranking Mechanism for Learning Resources," in *Proc. Int. Conf. on Hybrid Learning*, 2009

Gianni Fenu was born in Cagliari, Italy on December, 9 1960. He received the Dr. Ing. degree in engineering (*cum laude*) in 1985 from University of Cagliari, Italy. Currently is an Associate Professor of Computer Science at University of Cagliari (Italy), Director of UnitelSardegna Consortium and Head of the degree course of Computer Science at University of Cagliari. He is Coordinator of National and European Development Projects. He has research interests actually in the area of Computer Network, Cloud Computing and E-learning.

Prof. Fenu has authored about 80 scientific articles in national and international conferences and journals.