# Searching Learning Objects from Virtual Universities

Juha Puustjärvi & Päivi Pöyry
*Helsinki University of Technology,*
*Software Business and Engineering Institute,*
*POB 9600, FIN-02015 HUT, Finland*
*Juha.Puustjarvi@hut.fi; Paivi.T.Poyry@hut.fi*

## Abstract

*The fast development of technologies and vocational demands require specialized skills that need to be renewed frequently. Therefore, the role of continuous education and lifelong learning is becoming still more important. A goal of the ONES- project is to develop a system that supports learners in searching higher education courses that match their special needs. A corner stone of such a system is the metadata attached to course descriptions. Ontologies in turn are introduced to standardize the used metadata items. Consequently, ontologies provide a shared and common understanding of the metadata items. Furthermore, information retrieval models provide the framework where we can match learners' profiles (queries) against courses' metadata (course profiles). In this paper, we give an overview of the ONES system, and analyse the relevance of two information retrieval models for virtual universities. We also compare the performance of four algorithms for computing the similarities (matching). We argue that keywords based search (i.e. the Boolean model), though well suited for web searches, is overly coarse for virtual universities. Instead, the vector model, on which our implemented search engine is also based on, seems to be more appropriate, as it provides similarity measure, i.e., the learning object having the best match is presented first.*

## 1. Introduction

The fast development of technologies and vocational demands require specialized skills that need to be renewed frequently. Therefore the role of lifelong learning and continuous education is becoming still more important. However, the traditional higher education system is not appropriate for providing continuous education as it is overly tied to time and place. Instead, distance learning, or e-Learning adopts well for continuous education as it can be done in parallel to work. Yet, e-Learning sets new requirements for universities: they have to build global learning infrastructures, course material has to be in digital form, course material have to be distributed and learners must have access to various virtual universities.

As single virtual universities are independently created they may provide very heterogeneous functionalities and user interfaces. Ideally, the learner should be able to access all the virtual universities in a similar way, i.e., the heterogeneity of various virtual universities should not burden the learner. How this goal can be achieved is the main topic of the ONES-project. Consequently the main functions of the ONES system are the followings:

- To hide the distribution of e-Learning portals, and
- To hide the semantic heterogeneity (i.e., problems arising from using same words in different meaning and vice versa).

In order to achieve these goals the system will deploy many new technologies such as "one-stop portals", web services, service oriented architecture, RDF-based annotation, ontology editors, and distance measures in searching learning objects.

In this paper, we will restrict ourselves on the role of searches in the ONES-system. In particular, we will analyse the applicability of different information retrieval technologies. Our main argument is that the technology based on the Boolean model [1], though well suited for searches in the web, is not suitable for the emerging virtual universities. Instead, for virtual universities we have to develop methods, which allow learners to be more concerned with retrieving information about a subject than with retrieving data, which satisfy a given query. For example, a learner may be interested in courses dealing with object-oriented programming rather than in the courses where the term "java" or "C++" is stated.

When searching information about a subject (e.g. object oriented programming) the search engine must

somehow interpret the metadata of the learning objects and rank them according to a degree of relevance to the learner's query. The primary goal is to retrieve all the learning objects, which are relevant to a learner's query while retrieving as few non-relevant objects as possible. Unfortunately, characterization of the learner's information need is not a simple task. Furthermore, the difficulty is not only in expressing the information need but also in knowing how the learning objects should be characterized with the help of the metadata descriptions.

The rest of this paper is organized as follows. First, in Section 2, we give an overview of the architecture of the ONES-system. Then, in Section 3, the role of metadata and ontologies in virtual universities is illustrated. In addition, the usability of the Boolean and the vector model in a virtual university is analysed. Especially, two interpretations of a hierarchical ontology in the context of the vector model, called *weighted leaves* and *multilevel weighting*, are introduced. Then, in Section 5, the performance of four matching algorithms based on weighted leaves and multilevel weighting principles are compared. Finally, Section 6 concludes the paper by summarizing the feasibility of the proposed ideas.

## 2. The architecture of the ONES system

The name ONES stands for One Stop e-Learning Portal. As this name suggests a salient feature of the system is the aggregation of distance learning information from different learning sources in one portal. The idea of the one-stop portals is originated from one-stop shops, and later on it is also adopted in e-government applications.

The four main components of the ONES-system are (see Figure 1):

- Aggregation portal (mediator),
- Wrappers,
- E-Learning portals, and
- Course providers' tools.

The *aggregation portal* supports the learners in searching the courses that match to their specific needs. It differs from traditional database interfaces in a way that in addition to the traditional database queries it supports fuzzy queries. *Fuzzy queries* are similarity based, which means that if the similarity between the courses' profiles and the learner's query exceeds a certain threshold, they are said to match. A problem is that the current database management systems do not support fuzzy queries and therefore the ONES-system has to support them.

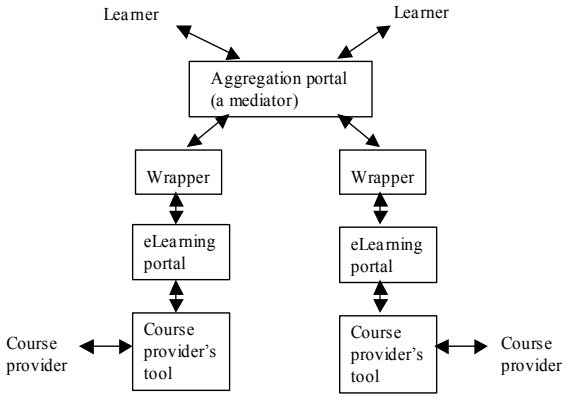


Figure 1. ONES-architecture.

From technological point of view the *aggregation portal* is *a mediator* [2]. It supports a virtual view that integrates several learning sources in much the same way as data warehouses do. However, since the mediator does not store any data, the mechanisms of mediators and warehouses are rather different. Since the mediator has no data of its own, it must get the relevant data from its sources and use that data to form the answer to the learner's query. As the data sources (e-Learning portals) are independently created it is obvious that they provide heterogeneous interfaces.

In order to hide the heterogeneity from the mediator there is a *wrapper* [2] between the mediator and each data source. Each wrapper provides equal functionality for the mediator. Ideally, each wrapper provides an interface for requesting the metadata of learning objects, i.e., descriptive information of courses, course packages and programs offered by educational institutions, e.g., universities.

From technological point of view each mediator is a *web service* [3]. Web services are self-describing modular applications that can be published, located and invoked across the Web. Once a service is deployed, other applications (e.g., an aggregation portal) can invoke the deployed service. In general, a web service can be anything from a simple request to complicated business process.

A course provider can enter data about a course through the *course provider's tool*. The main function of this tool is to provide an interface, which facilitates the creation of the metadata attached to learning objects. Basically this tool is analogous to the tools that support the content providers of electronic newspapers [4] in creating metadata items to news articles. The tool may even generate suggestions of the suitable metadata items, after which the author can make the necessary modifications and enter this information to the system.

## 3. Information retrieval in virtual universities

*Virtual university* has been defined as a space in which higher education studies are delivered to the learners through the newest information and communication technology [5]. Virtual university may be an institution that is directly involved in providing learning opportunities to the learners, or an organisation formed through different kinds of partnerships in order to facilitate university studies without being directly involved in providing instruction [6].

Information retrieval in the context of virtual universities deals with the representation, organization, and access to learning objects. The representation and organization of learning objects should provide the learner with an easy access to the learning objects. The system retrieves all the learning objects, which are relevant to learner while retrieving as few non-relevant learning objects as possible

In this section we will analyse the usefulness of different information retrieval models [7] for a virtual university. The used model determines the way the metadata of the learning objects are given as well as the way the learner's queries (information needs) are presented. Before analysing the information retrieval models we characterize the role of metadata and ontologies in virtual universities.

### 3.1. Metadata and ontologies

The term *metadata* has variable interpretations depending upon the circumstances in which it is used. For example, in the context of documents the common forms of metadata include the author(s), the source of publication, the length of document, etc. This kind of metadata in commonly called *descriptive metadata*. For example, the metadata elements of the Dublin Core [8] represent descriptive metadata.

*Educational metadata* describes any kinds of educational objects, such as study courses. The pedagogical features of the course, the contents, special target groups, and the technical requirements of the study course can be described with the help of educational metadata. Well-designed and sufficient metadata facilitates the learners' decision-making process and aids the educational institutions to provide suitable information about their course supply. Educational metadata is by nature semantic metadata, but a thorough metadata schema must include also at least structural metadata in order to be able to describe the learning objects efficiently [9].

A salient feature of *descriptive* metadata is that it is external to the meaning of the document, i.e., it describes the creation of the document rather than the content of the document. The metadata describing the content of the document is commonly called *semantic metadata.* For example, the keywords attached to many scientific articles represent semantic metadata [10]. *An ontology* provides a general vocabulary of a certain domain [11], and it can be defined as "an explicit specification of a conceptualisation"[12]. In essence, an ontology gives the semantics to the metadata.

In order to standardize semantic metadata specific ontologies are introduced in many disciplines. Typically such ontologies are hierarchical taxonomies of terms describing certain topics. For example, the ACM Computing Classification System is a hierarchy (a tree) in which the nodes represent the classes of the taxonomy. In Figure 2, a subset of that hierarchy is represented.

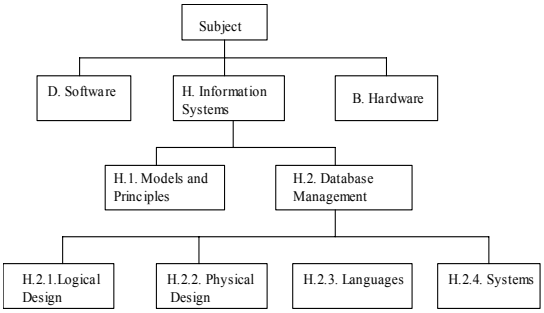


Figure 2. A subset of the ACM Computing Classification System.

### 3.2. The Boolean model

Applying the Boolean model in searches requires that each learning object is augmented by a set of metadata items such as keywords or classification identifiers (e.g., the searches in the CUBER system [8, 13] are based on the Boolean model). A learner can then query learning objects by Boolean expressions comprising of operands and operations. The operands are the used keywords and the operands are typically "*and", "or",* and "*not"*. For example, by using ACM Computing Classification system (Figure 2) the keywords attached to a learning object might be D, H.1 and H.2.2 (corresponding the keywords *Software*, *Models and Principles*, and *Physical Design*). Now, if a learner presents the query "*D and (B or H.1)"* (i.e., learning objects having the keyword "*Software*" and at least one of the keywords "Hardware" and "*Models and Principles*"), then the previous learning object will match that query.

The Boolean model is intuitive and clear. Moreover, it can be efficiently implemented even in the case of huge amount of objects. For example, many Web search engines are based on this model. However, using that model in a virtual university gives rise to following drawbacks:

- First, the model is based on a binary decision criterion, meaning that each learning object is predicted to be relevant or non-relevant. In reality, it is obvious that the resulting learning objects fit more or less to the query, i.e., some kind of grading should be possible.
- Second, expressing the requirements of learning objects by a Boolean expression may be difficult.
- Third, a typical problem concerning search engines based on the Boolean model is that either the result of the query includes too many or too few learning objects.

In the next section we consider a more advanced model, which avoids many of the drawbacks described above.

### 3.3. The vector model

The vector model differs from the Boolean model in that weights can be assigned to each metadata item of a document as well as to the keywords of the query. The idea behind this model is that we can more accurately specify the queries and the contents of the documents (e.g., learning objects).

Assuming that the standard metadata items (e.g., the classes in Figure 2) specify a vector space (i.e., each item (keyword) in the hierarchy represents a dimension in the vector space), we can represent each document and query as a vector in that vector space. Then we can process the query by computing the distance of the query vector and the document vectors. This kind of computing requires that the sum of the weights of each document and query equals to a predefined constant. For convenience, the used constant is usually one.

As the result of the query the documents are sorted in the order determined by the similarity, i.e. the document having the best match with the query is presented first. The number of the documents in the result should be restricted by requiring a certain degree of similarity.

Using the vector model in a virtual university requires that the course provider assign the metadata items and their weights into each learning object. The metadata items to be used are selected from the used domain ontology. Depending on the used course provider's interface this can be done in various ways. For example, as in our prototype system, there may be an ontology structure on which the course provider inserts the weights. In Figure 3, the ontology structure of the Figure 2 is augmented by setting weights on the nodes "B.H.2", and "H.2.2". Note that the node having no weight means that its weight is actually zero. Hence, the profile of the learning object can be presented by a vector in 9-dimensional vector space as follows: [0 x D, 0 x H, 0.3 x B, 0 x H.1, 0.6 x H.2, 0 x H.2.1, 0.1 x H.2.2, 0 x H.2.3, 0 x H.2.4]. That is, the profile is a point in an orthogonal 9-dimensional vector space.

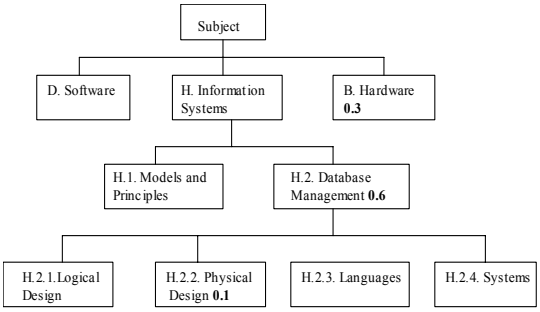


Figure 3. A metadata specification of a learning object.

The gain of attaching metadata description for learning objects is that we can use mathematical distance measures in computing learners' queries. Further, computing the distance requires that the descriptions (vectors) be specified in an orthogonal vector space. In other words, the nodes in the hierarchy that are used in profile vectors must be independent. In practice this means that we have to follow one or the other of the following interpretations:

- *Multilevel weighting interpretation*: the leaves and the nodes of the ontology hierarchy represent independent concepts.
- *Weighted leaves interpretation:* the parent node represents the union of its siblings. In other words, each sibling represents a subset of its parent. Yet the siblings represent independent concepts.

The intuition behind *multilevel weighting* is that we can express the level of a leaning object (as well of a query) by altering the weights on a node and its siblings. To illustrate this let us consider the weighting of the course "Physical design in database management systems". Now, it is obvious that the weights should be given on the node H.2 (Database management) and its siblings H.2.2 (Physical design) and

H.2.4 (Systems). Assuming that approximately half of the course deals with databases in general and the other part deals with physical design and database management systems, then giving weight 0.4 to H.2 (Database management), 0.3 to H.2.2 (Physical design) and weight 0.3 to H.2.4 (Systems) could be an appropriate assignment. On the other hand, if the course is very specific then the weight of H.2 could be zero.

If we follow the *weighted leaves* interpretation, then in determining the profile of a learning object weights are set only on the leave nodes of the hierarchy. Consequently, the profiles of the learning objects are specified by vectors in an orthogonal vector space, which is determined by the leave nodes of the hierarchy. To illustrate this approach let us consider the weighting of the course "Physical design in database management systems". In this case, all the weights are given on the nodes H.2.2 (Physical design) and H.2.4 (Systems) independently of the level of the course.

## 4. Processing learner's queries

The learner presents queries in the same way as the content provider determines the weights of the learning object; both these are presented by vectors. Hence the query presents an ideal profile of the learning objects that satisfy the learner's requirements. For example, assuming that the multilevel weighting interpretation of the ontology is used, and a learner wants to find basic courses concerning database management. In this case the learner will set rather heavy weight on H.2 (database Management) and lighter weights on H.2.1 (Logical Design), H.2.2 (Physical Design) and H.2.3 (Languages). In contrast, if a student is looking more advanced courses on database management then the student will give a lighter weight on H.2 and heavier weights on its siblings.

As the learners interact with the system by submitting queries it is reasonable to require that the response times should be only a few seconds. We investigated the effects of different matching algorithms and the amount of stored learning objects on response times. The test environment was equipped with Pentium II processor and 192 MB memory. The computers were running the Sun Solaris 5.8 operating system. We implemented and tested four matching algorithms, i.e., algorithms, which compute the distance measures of learning objects and learners' queries. We next give a short description of the algorithms.

The *Cosine matching algorithm* [7] calculates the cosine measure between the query (a vector) and the documents profiles. As a matter of fact the algorithm does not compute distance measures but rather approximates distance measures by computing the angles of the query vector and the vectors representing documents, such as the learning objects.

The *Euclidean matching algorithm* [14] calculates the Euclidean distance from the query profile to all learning objects' profiles. The *Manhattan distance algorithm* [15] calculates a so called "city block-distance". The name comes from the fact that this measure in two dimensions tells how many blocks in a city one would have to walk between two points.

Our developed *Fuzzy matching algorithm* attempts to achieve more efficient matching procedure than the "exact" matching algorithms. The improved efficiency is achieved by performing the actual matching on a pre-selected subset of all learning objects. The predefined subset of the documents' profiles is determined by choosing the three biggest weights from the query and then computing the subset based on these weights. Then only the profiles, the weights of which are within a specified tolerance interval are selected for the final query processing. Therefore the result set is not guaranteed to contain all the profiles that are closest to the matching profile. However, the closeness values of the profiles in the actual result set are exact, since they are calculated using the Euclidean measure.

| | 1 000 | 5 000 | 10 000 |
|---|---|---|---|
| Cosine | 0.80 | 0.93 | 1.07 |
| Euclidean | 0.83 | 0.96 | 1.16 |
| Manhattan | 0.83 | 0.98 | 1.23 |
| Fuzzy | 0.61 | 0.73 | 0.89 |

Table 1. Matching times for the algorithms.

The computing time for matching of each algorithm is presented in Table 1. The test was performed for different amount (1000, 5000 and 10 000) of learning objects. Basically, the difference of Euclidean, Cosine and Manhattan algorithms were rather small (less than 10 percents). Fuzzy matching algorithm required least computing time (about 20 percent less than others). However, the test proves that all the algorithms are quick enough in the test environment as the response times are less than 1.2 seconds. If the number of the learning objects or the dimensions of the vector space (i.e. the used attributes in the profile) increases, then it obvious that the Fuzzy Matching algorithm will be more superior to the other algorithms. In our test environment the vector space comprised of 15 dimensions, i.e., each profile could have at most 15 attributes. In practice, the number of attributes cannot increase significantly as otherwise the determining the weights for learning objects would overly burden the coarse creators. In addition, as the system is developed for universities it is not obvious

that number of learning objects can be very huge, e.g., over 10 000.

## 5. Conclusions

E-Learning sets new requirements for universities: they have to build global learning infrastructures, course material has to be offered also in digital form, course material have to be distributed via the Internet and learners must have access to various virtual universities. A problem is that the current virtual university portals provide heterogeneous functionalities, which in turn hampers the learner in accessing various virtual universities.

The main goal of the ONES-project is to investigate the ways of integrating various virtual universities in a way that such an aggregated virtual university would be as easily accessible for a learner as a single virtual university. Achieving such a goal requires mutual understanding of the used technology and standardized descriptions of the learning objects. Furthermore, searching from various virtual universities requires mutual understanding of the information retrieval model to be used.

We argue that keywords-based search (i.e. the Boolean model), though well suited for general web searches, is unsuitable for the virtual universities' purposes. Instead, the vector model (on which our implemented search engine is also based on) seems to be more appropriate as it provides a similarity measure, i.e. the learning object having the best match is presented first. We also introduced two interpretations for the hierarchical ontologies, which allow increasing the power of the used metadata descriptions. And finally, we also compare the performance of four algorithms for computing the similarities of the profiles. It turned out that our developed Fuzzy Matching algorithm requires less computing time as the other "exact matching" algorithms represented in the literature.

## 6. References


[1] Yan, T., & Garcia –Molina, H. 1994. *Index structures for selective dissemination of information under the Boolean Model*, ACM Transactions on Database Systems, 19(2): pages 332-364.

[2] Garcia-Molina, H., Ullman, J. &. Widom, J. 2000. *Database System Implementation.* Prentice Hall.

[3] Vasudevan, V. 2001. *A web service primer*. http://www.xml/lpt/a/2001/04/04/webservices/indea x.html.

[4] Yli-Koivisto, J. & Puustjärvi, J. 2002. *CoMet: an electronic newspaper prototype*. Workshop on XML in Digital Media. In Proc of the 8th International Conference on Distributed Multimedia Systems (DMS'2002), pages 703-707.

[5] Niemi, H. 2002. *Empowering learners in the virtual university*. In Theoretical Understandings for Learning in the Virtual University. Eds: Niemi, H. & Ruohotie, P. Research Center for Vocational Education and Training, University of Tampere, Finland.

[6] Ryan, S., Scott, B., Freeman, H. & Patel, D. 2000. *The Virtual University. The Internet and Resource-Based Learning.* Kogan Page, London.

[7] Baeza-Yates, R.& Ribeiro-Neto, B. 1999. *Modern Information Retrieval*. Addison Wesley.

[8] Pöyry, P., Pelto-Aho, K. & Puustjärvi, J. 2002. *The role of meta data in the CUBER system.* In the Proc. of the Annual Conference of the SAICSIT 2002, pages 172-178.

[9] Lamminaho, V. 2000. *Metadata specification: Forms, Menus for Description of Courses and All Other Objects.* CUBER project: Deliverable D3.1

[10] Jokela, S. 2001. *Metadata Enhanced Content Management in Media Companies*. Acta Polytechnica Scandinavica. Mathematics and Computing Series No. 114. Helsinki University of Technology: Doctoral thesis.

[11] Fridman Noy, Natalya & McGuinness, Deborah L. 2001. *Ontology Development 101: A Guide to Creating Your First Ontology*. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05, Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001. 19 p.

[12] Gruber, Thomas R. 1993. *Toward principles for the design of ontologies used for knowledge sharing*. Padua workshop on Formal Ontology, March 1993. 23 p.

[13] Pöyry, P. & Puustjärvi, J. 2003. *CUBER: A Personalised Curriculum Builder*. To appear in the 3rd IEEE International Conference on Advanced Learning Technologies (ICALT 2003).

[14] Friedman et al. 1977. *An algorithm for finding best matches in logaritmic expected time,* ACM Transactions on Mathematical Software, 3 (3): pages 209-226.

[15] J. Bentley, B. Weide, and A. Yao. 1980.*Optimal expected-time algorithms for closest point problem*. ACM Transactions on Mathematical Software, 6(4): pages 563-580.