Towards a Syllabus Repository for Computer Science Courses

Manas Tungare, Xiaoyan Yu, William Cameron, GuoFang Teng,
Manuel A. Pérez-Quiñones, Lillian Cassel, Weiguo Fan, Edward A. Fox
Virginia Tech and Villanova University
{manas, xiaoyany, perez, wfan, fox}@vt.edu,
{william.cameron, guofang.teng, lillian.cassel}@villanova.edu

ABSTRACT

A syllabus defines the contents of a course, as well as other information such as resources and assignments. In this paper, we report on our work towards creating a syllabus repository of Computer Science courses across universities in the USA. We present some statistics from our initial collection of 8000+ syllabi. We show a syllabus creator that is integrated with Moodle [5], an open-source course management system, which allows for the creation of a syllabus for a particular course. Among other information, it includes knowledge units from the Computing Curricula 2001 body of knowledge. The goal of the syllabus repository is to provide added value to the Computer Science Education community, and we present some such offerings. We conclude by presenting our future plans for the syllabus repository. These include using automated techniques to collect and classify syllabi, providing recommendations to instructors when creating a syllabus, and allowing the community to share their syllabi automatically. The syllabus collection will be part of the Computing and Information Technology Interactive Digital Educational Library (CITIDEL), a collection of the National Science Digital Library (NSDL).

Categories and Subject Descriptors

H.3.m [**Information Storage and Retrieval**]: Miscellaneous

Keywords

Syllabus, Curriculum, Computing Curricula 2001

1. INTRODUCTION

A syllabus forms the backbone of a course offering: a complete syllabus typically includes the course number, title, a description, the learning objectives of the course, a list of the topics covered, links to reference material such as books or publications, and other related information. The various learning objects that are included in a course offering are created based on the syllabus definition, and are tightly integrated with the reference material (also included in the syllabus).

SIGCSE 2007 Covington, Kentucky, USA.

Thus, knowledge of a course syllabus can be used to assess the structure of a course, the exact knowledge units covered, and the relative time devoted to each of them. The syllabus describes how individual learning objects are combined to form larger entities and packaged as a course for students.

We are in the process of creating a repository of Computer Science syllabi. The first generation of this collection is composed of syllabi collected from the Web. We also have created tools that allow professors to create a syllabus and automatically publish it to our collection. We plan to use our collection to provide recommendations to course creators (e.g., by suggesting a textbook to use for a particular course) and to provide other services to the Computer Science Education community. The collection of syllabi will be part of the Computing and Information Technology Interactive Digital Educational Library (CITIDEL), a collection of the National Science Digital Library (NSDL).

In this paper, we discuss our approach to collect the initial syllabi in our repository, tools to help create new course syllabi, tools to compare existing syllabi, and future plans for our collection.

2. CREATING THE COLLECTION

In order to reap the larger advantages of a syllabus repository, a sizable collection of syllabus documents must first be available. However, populating an empty syllabus repository with enough syllabi within a small frame of time would be a monumental task. Professors are more likely to continue publishing their syllabi over the Web as HTML or PDF documents instead of sharing them in a common repository. Including just a few syllabi in a collection would likely not provide enough return over the effort invested in sharing the syllabi. Thus, given the extensive availability of syllabi as published Web documents, we decided to gather them automatically.

We used Google to locate documents that show a high likelihood of being a syllabus. This was accomplished by two specialized sets of queries submitted to Google's search engine: the first, to locate departments of interest within an educational institution, and second, to locate syllabi within that department. Since our current interest is in accumulating syllabi in the field of Computer Science, we issued a query for

"`computer science site:edu`"

The result of this query was a list of pages within the computer science departments of many US university websites (i.e., those whose domain names end with '`.edu`'). We processed each resulting URL to obtain the relevant domain name, i.e., the homepage of each department. The resulting set of departments included 98 Computer Science departments (e.g., `cs.vt.edu`). We then

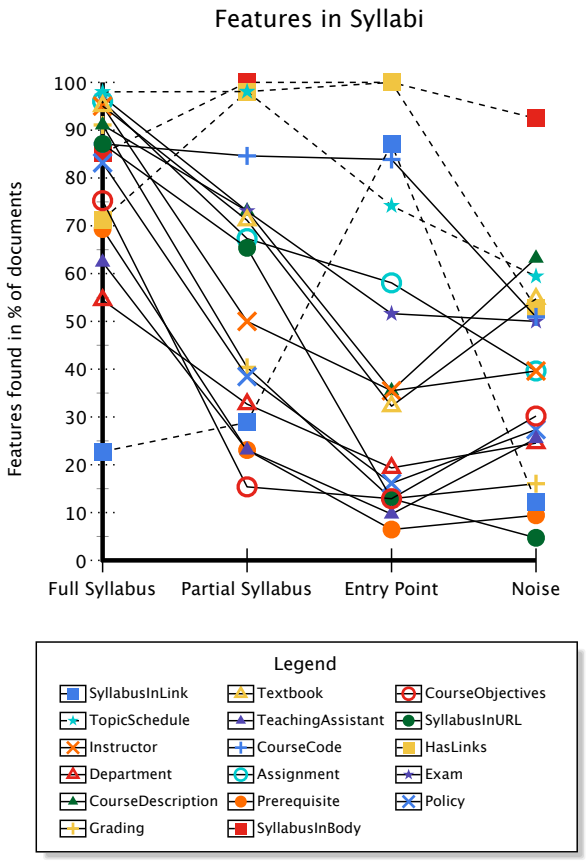

Figure 1: Features found in Syllabus Documents

issued the second set of queries, one per department, to locate documents within that department. As an illustration, the following query resulted in a list of syllabus-like documents from the Department of Computer Science at Virginia Tech that exhibit high likelihood of being a syllabus.

"`syllabus site:cs.vt.edu`"

The syllabus documents thus obtained were stored in a structured organization (for later retrieval), and converted from their native format (either PDF, PS, or HTML) to plain text to enable further text processing. The result of our initial crawl produced 8000+ syllabi.

We plan to extend our document collection strategies to include academic institutions from around the world (non-`.edu` Web sites), and departments named, for example, 'Computing Sciences' or 'School of Information Sciences'. We plan to update this collection several times a year, since it is likely that syllabi are posted and revised in accordance with academic calendars.

2.1 Statistics About the Syllabi We Found

The inclusion of a page within Google's search results is not a definitive indicator that a particular webpage is a syllabus. From a quick scan of the documents retrieved, we identified four categories of documents: a full syllabus, a partial syllabus, a syllabus entry point, and noise. A full syllabus is one that contains most of the basic syllabus components and no links to other related documents. A partial syllabus contains some important syllabus components along with links to other components on another web page.

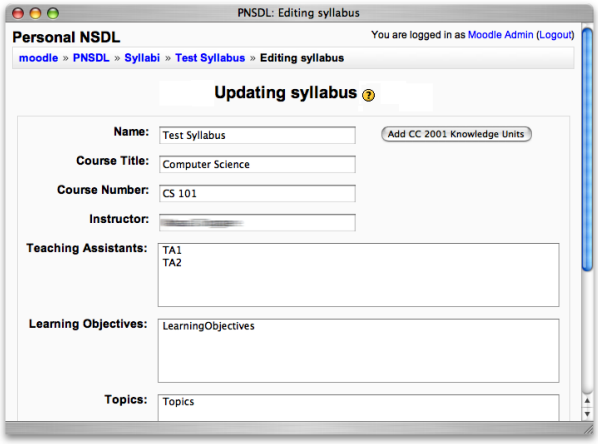

Figure 2: SyllabusMaker, a plug-in for Moodle.

A syllabus entry page (for example, a course web site home page) contains a link to a syllabus, or to the various pieces that make up a complete syllabus. The rest of the documents are considered noise. These often are web pages where the keyword 'syllabus' occurs several times, such as articles about how to write a syllabus, discussion fora, or published exams.

We manually classified a small set (1000) of the 8000+ syllabi into the four categories. We defined 17 features normally found in a course syllabus (e.g., course name, learning objectives, textbook, etc.) as shown in the legend area of Figure 1. We then extracted them from this smaller set using pattern-matching techniques. Figure 1 shows the frequency of occurence of the defined features. It can be seen that the presence of some features is a strong indicator of a document belonging to one of the four categories. For example, from a 2-sample t-test, a syllabus hyperlink (i.e., the keyword *'syllabus'* appearing in the hyperlinked URL or in the anchor text) appears statistically significant (α = .05) more in a entry page than in any other kind of documents.

2.2 Updating the Syllabus Collection

At the start of each new academic term (semester or quarter), syllabi are often updated to reflect new course offerings or changes to existing ones. An effective method for capturing these updates is necessary to maintain a fresh index of syllabi within the repository. We plan to have an automated classifier so that new results from our web crawl will be automatically processed.

With the information learned from the manual classification, we then can build an automatic classifier that will allow us to quickly identify a full syllabus versus a web document that simply contains a mention of a syllabus. The 1000 documents classified are a random subset of our larger syllabus collection. The size of 1000 is large enough to design and train a syllabus classifier for future automatic syllabus identification purposes. With the extracted features and the 1000 documents as a training set, we can train a syllabus classifier using a machine learning approach such as the decision tree algorithm [4], support vector machines [11], or naïve Bayes methods [4].

3. TOOLS

We are currently loading our syllabus collection into DSpace [1] so that users can navigate to and contribute to the collection in a seamless way. However, having the repository in a centralized collection is hardly enough to encourage the use of this information.

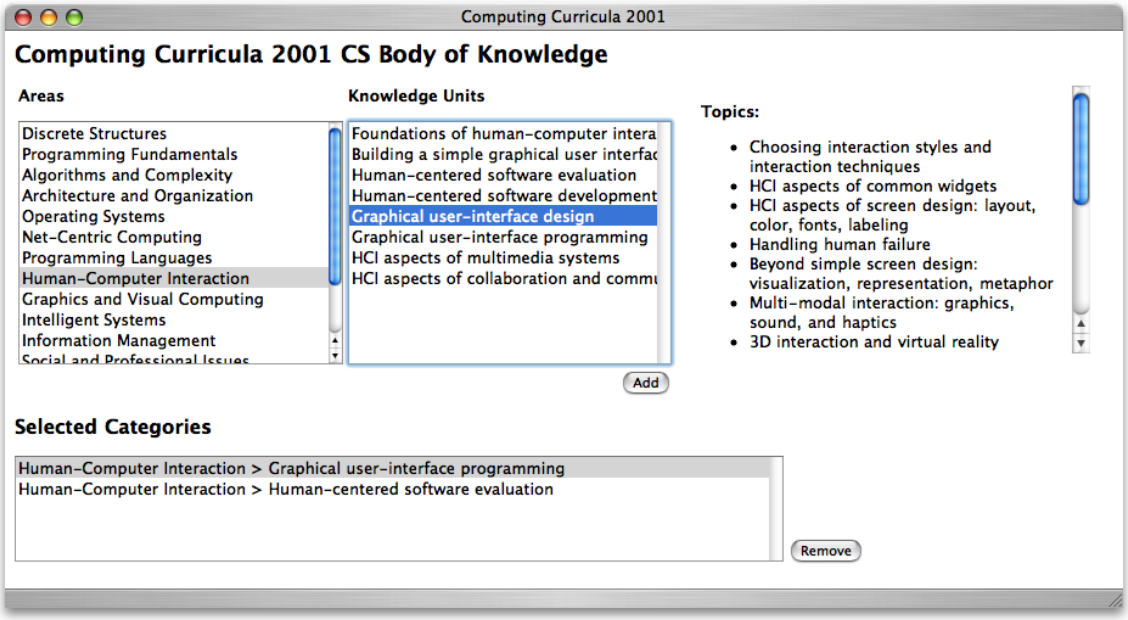

Figure 3: Adding Knowledge Units from CC 2001 to a Syllabus

We have created several tools that, together with the repository, represent a suite of tools and data to support the creation, modification, and use of the knowledge captured in the syllabus repository.

3.1 SyllabusMaker

One of our tools is the SyllabusMaker. This tool helps create a syllabus and supports the collection of meta-data which will automatically be shared with our central repository. Our initial version of SyllabusMaker is in the form of a plug-in to Moodle [5], an open-source course management system (CMS). With this plug-in, instructors can create their own syllabus and share it, if they so desire, with our syllabus repository. A screenshot of one of the dialogs within SyllabusMaker is shown in Figure 2.

By providing a tool like SyllabusMaker embedded in a CMS, we support the instructors in the creation of syllabi, we collect meta-data right at the point of creation, and it allows us to provide added value to the course creation process (more about this later in the paper). Initially, our tool is only available in Moodle, but we are planning to port it to Sakai [8], another open-source CMS. We also have plans to create a stand-alone syllabus creation tool that will be available on the Web.

3.2 Linking to the CC 2001 Body of Knowledge

What is unique about our syllabus tools is that they allow connections to curriculum schemas. In the case of Computer Science, an instructor can select a series of topics from the CC 2001 body of knowledge [7] that are covered in the course.

The Computing Curricula Project [7] (CC 2001) was undertaken to develop guidelines for computing curricula at an undergraduate level. In its final report, the authors present detailed coverage of the CS body of knowledge, core areas for undergraduate studies, learning objectives and curriculum models. In particular, the report recommends the number of hours to be devoted to particular knowledge units for courses with specified learning objectives as a guideline to instructors preparing a course. This data can be of use to instructors when creating a syllabus, either to design their course based entirely on CC 2001 recommendations, or to pick the right mix of knowledge units, given their unique goals for a particular course.

Our tool provides instructors the option to select knowledge units (KU) from CC 2001 to 'tag' a particular syllabus. This would allow instructors creating courses to use the content and knowledge included in the CC 2001 at course creation time. SyllabusMaker includes a browser for the CC 2001 knowledge units and brief descriptions for each unit to ease the creation of syllabi. Clicking the button labeled *'Add CC 2001 Knowledge Units'* in Figure 2 opens the KU browser, shown in Figure 3. From there, the instructor can select the set of KUs desired and include them in the syllabus.

3.3 Syllabus Comparison Tool

With a complete definition of syllabi available, it is easy to visualize the distribution of topics (or Knowledge Units from CC 2001) across a series of related courses. The topics included in a course can easily be used to compare two different courses and get a sense of if they are similar or not, based solely on the matching of KUs covered in them.

Figure 4 shows our Syllabus Comparison Tool with two syllabi loaded for comparison. The list of knowledge units down the left side of the screen is the union of all the knowledge units covered in the courses shown across the top. Our tool allows comparing an arbitrary number of courses, although practically it is most appropriate when comparing a small number of courses. In this example, the interface clearly shows how the two courses compare; the User Interface Software course covers more implementation issues (e.g., HC2, HC5, HC6) than the Introduction to HCI course. Though it currently works only with syllabi built with our SyllabusMaker, we plan to extend it to compare any syllabi from our repository.

This comparison mechanism can be used as a similarity score. Users of our tools could easily look for syllabi that are "similar" to their own courses. This will enable future tools to provide recommendations of other similar syllabi to instructors, as well as recommendations for textbooks, topics to discuss, etc. obtained from the set of other similar syllabi.

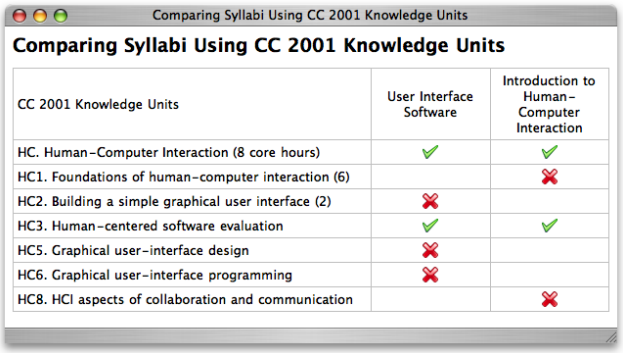

Figure 4: Comparing Two Syllabi Using CC 2001 Knowledge Units

3.4 Syllabus Search Engine

A syllabus repository can provide a course creator or student additional reference material from similar courses, but it may be difficult to find links between dissimilar courses. A full-text search on the contents of collected syllabi makes it easy to issue keyword queries that span the entire corpus, and thus locate hard-to-find references. Our search engine queries the text extracted from syllabi collected from the Web, as well as those entered via SyllabusMaker.

4. USES OF A SYLLABUS REPOSITORY

Availability of a syllabus repository opens up possibilities for many innovative applications. We describe a few of them in the remainder of this paper. We expect that the presence and easy access to a syllabus collection will almost certainly lead to new creative ways of using this information.

4.1 Assisting Instructors when Creating New Syllabi

When an instructor creates a new syllabus, there is a high likelihood of the end-product being similar to other syllabi (or perhaps a combination of two or more syllabi). Although the similarity between two syllabi at the same institution is likely to be minimal, the availability of syllabi from other institutions increases the chances of finding a match. Recommendations can be made to the instructor based on resources from similar syllabi to assist with one or more of the following tasks:

- Recommending the relative number of hours to be assigned to each knowledge unit, as per standardized syllabi classification schemes or simply based on what other courses do with the particular knowledge unit;
- Locating books, reading material, and other resources that similar syllabi have included;
- Enabling the instructor to import material from other syllabi (licensed appropriately) instead of having to recreate the entire syllabus involving extra effort (e.g., copy the full text reference from another syllabus);
- Identify possible contact points to discuss and share experiences in a particular course.

4.2 Comparing Computing Programs

There are bound to be subtle differences between syllabi for the same course offered at different universities. In case of multi-disciplinary topics such as Human-Computer Interaction (HCI), a particular school might offer an introductory course with more emphasis on usability engineering processes, while another one might devote more time to interface design principles, and a third one might devote more time to theories and models. We could use a collection of syllabi from a particular university and get a sense of the emphasis given at that institution to the different sub-areas within computing. This will be helpful to students for obtaining better information about specific programs in computer science before enrolling in one that fits their interests more closely. In a way, this functionality is analogous to our syllabus comparison tool, but instead using data at the program level. A simple visualization can show all the KUs covered in a department or school. Figure 5 shows an example diagram depicting this idea; comparing two curricula based on the knowledge units covered in their respective courses provides a way to see the focus of each curriculum.

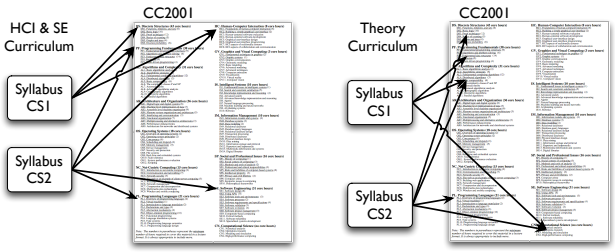

Figure 5: Comparing Two Computing Programs

4.3 Web Service API

To encourage more innovative applications from third-party developers, we plan to offer a web service application programming interface (API) which will let developers create tools to access and enter data into the repository.

5. PUTTING IT ALL TOGETHER

Given a syllabus collection, our next step is to extract important components from it such as the course title, course description, instructor, schedule and topics. A syllabus can be viewed as a collection of meta-data for a course. Therefore, machine learning approaches like support vector machines [11] and hidden Markov models [6] (HMM) can be applied to extract this meta-data. They have been used already to extract limited meta-data from syllabi [9]. The goal is to obtain similar meta-data as is produced by our SyllabusMaker, but automatically extracted from our 8000+ collection.

With the meta-data information available for course syllabi, we will be able to recommend resources to instructors and students automatically, based on the focus of the course. The recommendations will be done right at the CMS where instructors and students currently interact within the context of their courses. We plan to use techniques from genetic programming; ARRANGER (**A**utomatic **R**endering of **RAN**king functions by **GE**netic programming) [2] is a user modeling tool that approximates a user's ranking preference based on user feedback. In other words, given a set of documents along with their relevance information from a user, ARRANGER can automatically tune a ranking function based on syntactic and lexical evidence embedded in the documents and can discover a personalized ranking function that can be used to reorder information based on personal preferences. Combined with effective user profiling and/or user feedback, ARRANGER can deliver even higher quality information to end users.

It is likely that syllabi classified and parsed automatically will

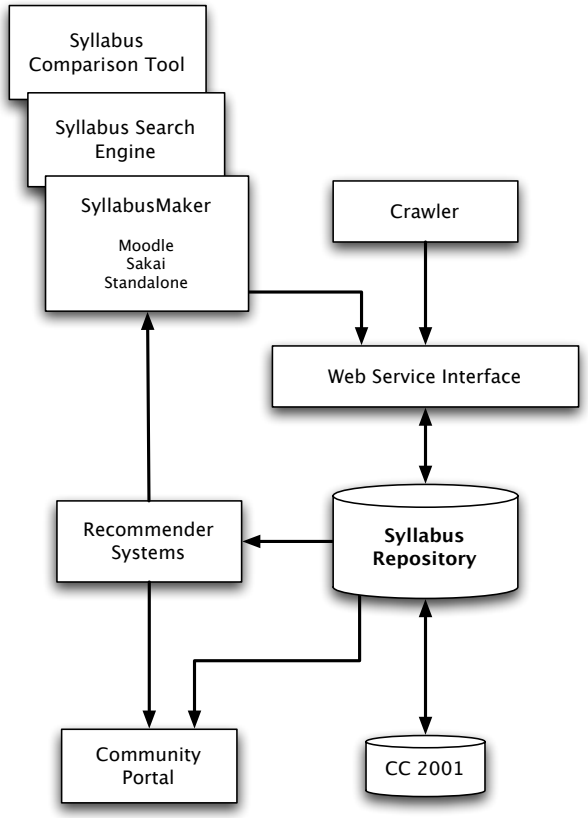

Figure 6: Overall Architecture of the Syllabus Repository System

have some errors. Most of these can be spotted by users of the system, and leveraging the assistance of the community would be helpful to maintain the quality of the repository. Success (to a high degree) has been achieved in open community-based systems in correcting structured information that was automatically parsed by autonomous agents [3] (referred to as Distributed Error Correction).

Finally, our efforts towards producing meta-data for syllabi from our tool and from our crawler have sparked interest in creating a standard representation for syllabi. No such standard exists today. Our group is leading the effort to define and use such a standard. The advantages of having a standard go beyond our initial set of tools and can have implications for other disciplines. More on the standard representation is discussed elsewhere [10].

6. CONCLUSIONS

In this paper we have presented our initial work on creating a collection of Computer Science course syllabi and our work to create tools to support the use of such a collection. Our field, more than most other sciences, changes rapidly. This brings in pressure from industry, students, and university administrators to update syllabi and curricula. For us to effectively manage our curricula and courses, we need tools like the ones we have presented here. They allow us to directly manipulate the knowledge embedded in course syllabi and to use and reuse them as a point of collaboration among the Computer Science faculty and other constituents.

A syllabus is an entry point to learning materials, presentations, readings, and other content. Having the connections from these learning objects to syllabi will allow us to better understand where and when supplemental materials can be used best. For example, an animation depicting sorting for an introductory course in CS needs to show different parameters than a similar animation for an advanced class in algorithms. Simply finding the animation is not sufficient to know if it is appropriate for an instructor's particular needs. But, finding a course syllabus that matches one's own, and through it seeing what other supplemental materials are used, can provide a significant benefit to the community of Computer Science professors.

7. ACKNOWLEDGMENTS

This work was funded by the National Science Foundation under grant DUE #0532825.

8. REFERENCES

[1] DSpace. DSpace.org. http://www.dspace.org/, Last Accessed: March 2006.

[2] W. Fan, E. A. Fox, P. Pathak, and H. Wu. The effects of fitness functions on genetic programming-based ranking discovery for web search: Research articles. *Journal of the American Society for Information Science and Technology*, 55(7):628–636, 2004.

[3] S. Lawrence, K. Bollacker, and C. L. Giles. Distributed error correction. In *DL '99: Proceedings of the Fourth ACM Conference on Digital Libraries*, page 232, New York, NY, USA, 1999. ACM Press.

[4] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[5] Moodle. Moodle - a free, open source course management system for online learning. http://www.moodle.org/, Last Accessed September 2006.

[6] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. pages 267–296, 1990.

[7] The Joint Task Force on Computing Curricula. Computing Curricula 2001. *Journal on Educational Resources in Computing (JERIC)*, 1(3es):1, 2001.

[8] The Sakai Project. Sakai: Collaboration and learning environment for education. http://www.sakaiproject.org/, Last Accessed September 2006.

[9] C. A. Thompson, J. Smarr, H. Nguyen, and C. Manning. Finding educational resources on the web: Exploiting automatic extraction of metadata. In *Proc. ECML Workshop on Adaptive Text Extraction and Mining*, 2003.

[10] M. Tungare, X. Yu, G. Teng, M. Pérez-Quiñones, E. Fox, W. Fan, and L. Cassel. Towards a standardized representation of syllabi to facilitate sharing and personalization of digital library content. In *Proceedings of the 4th International Workshop on Applications of Semantic Web Technologies for E-Learning (SW-EL)*, 2006.

[11] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.