CMPUT 701

Essay:

EduNuggets: An Intelligent Environment for Managing and Delivering Multimedia Education Content

By

Jari, Kavita
<kavita@cs.ualberta.ca>

Designated Readers:

Dr. Eleni Stroulia	<stroulia@cs.ualberta.ca>
Dr. Kenny Wong	<kenw@cs.ualberta.ca>

Expected Date of Completion: December 2002

TABLE OF CONTENTS

ABSTRACT....4

1. MOTIVATION....5

2. INTRODUCTION....7

3. RELATED WORK....8

4. ARCHITECTURAL DESCRIPTION OF THE EDUNUGGETS FRAMEWORK....11

4.1 THE REPOSITORY....12
4.2 THE DOMAIN MODEL AS A TOPIC MAP....13
4.2.1 DOWNLOADING CONTENT AND DOCUMENT PRE-PROCESSING....16
4.2.2 GENERATING NUGGETS....16
4.2.3 GENERATING TOPICS....17
4.2.4 GENERATING ASSOCIATIONS....18
4.2.5 BUILDING A TOPIC MAP....19
4.3 THE EDUNUGGETS FRAMEWORK....20
4.3.1 THE EDUNUGGETS INSTRUCTOR APPLICATION....20
4.3.2 THE EDUNUGGETS STUDENT APPLICATION....22
4.4 INFORMATION RETRIEVAL....26
4.4.1 LATENT SEMANTIC INDEXING....26
4.4.1.1 Definition....26
4.4.1.2 Process....26
4.4.1.3 Implementation....28
4.4.2 NAÏVE BAYESIAN CLASSIFIER....29
4.4.2.1 Definition....29
4.4.2.2 Process....29
4.4.2.3 Implementation....30
4.4.3 SEARCH PROCESS IN THE EDUNUGGETS STUDENT APPLICATION....33
4.4.4 USER FEEDBACK....35

5. MULTIMEDIA DEVELOPMENT IN EDUNUGGETS....36

6. EVALUATION....40

6.1 CASE STUDIES AND RESULTS FOR THE NAÏVE BAYESIAN CLASSIFIER....41
6.1.1 Nugget generation....43
6.1.2 Topic generation....48
6.1.3 General Observations....51
6.2 USABILITY EVALUATION AND ASPECT-ORIENTED PROGRAMMING....52
6.3 EXPERIMENTS AND RESULTS FOR THE SMIL EDITOR....52
6.2.1 Testing the third-party code that does PowerPoint slide to GIF....53
6.2.2 Playing the SMIL file on Unix machines....53
6.2.3 GIF image rendering....54

7. FUTURE WORK....55

8. CONCLUSION....57

9. ACKNOWLEDGEMENTS....58

10. REFERENCES....58

11. APPENDIX....60

TABLE OF FIGURES AND TABLES

FIGURE 1 - OVERALL ARCHITECTURE OF THE EDUNUGGETS FRAMEWORK 11

FIGURE 2 - A VIEW OF THE EDUNUGGETS REPOSITORY 12

FIGURE 3 - SAMPLE EDUNUGGETS TOPIC MAP 14

FIGURE 4 - SAMPLE VIEW OF THE NUGGETS' TABLE 17

FIGURE 5 - A SNAPSHOT OF THE TOPICS' VIEW 18

FIGURE 6 - A SNAPSHOT OF THE ASSOCIATIONS' VIEW 19

FIGURE 7 - THE EDUNUGGETS INSTRUCTOR APPLICATION 21

FIGURE 8 - THE EDUNUGGETS STUDENT APPLICATION 22

FIGURE 9 - INFORMATION RETRIEVAL IN THE EDUNUGGETS STUDENT APPLICATION 25

FIGURE 10 - ANNOTATED VIEW OF THE LSI ALGORITHM 28

FIGURE 11 - ANNOTATED VIEW OF THE NBC ALGORITHM 30

FIGURE 12 - SAMPLE RESULTS FROM THE NBC ALGORITHM FOR QUERY 'DESIGN' 32

FIGURE 13 - SAMPLE RESULTS FROM THE NBC ALGORITHM FOR QUERY 'COGNITION' 34

FIGURE 14 - THE SMIL EDITOR APPLICATION DISPLAYING A SMIL FILE 37

FIGURE 15 - BASIC STRUCTURE OF THE NAIVEBAYES.SMIL FILE 38

FIGURE 16 - A SMIL FILE CALLED "NAIVEBAYES.SMIL" PLAYING IN REALONE 39

TABLE 1 - CLASSIFIER RESULTS FOR 3 DATA SETS AND 2 XPATH EXPRESSIONS 42

TABLE 2 - CLASSIFIER RESULTS WITH FEWER TOPICS AND 1 XPATH EXPRESSION 48

TABLE 3 - CLASSIFIER RESULTS WITH RAINBOW TOPICS AND 1 XPATH EXPRESSION 50

FIGURE 17 - SAMPLE IMAGE LAYOUT USING THE FIT ATTRIBUTE IN SMIL 52

EduNuggets: An Intelligent Environment for Managing and Delivering Multimedia Education Content

Abstract

Today's teaching and learning practices are evolving to leverage the continuously increasing information available on the web for all conceivable subject matters. This change is clearly visible in the field of Web-based learning. Instructors use on-line information sources, in addition to their textbooks, to collect content for their teaching material. They also use a variety of tools to prepare their presentations of this material, and make them, in turn, available on the web. Students also use the web to find more information on the subject matter that they study. This wealth of information presents a great challenge: how to provide an integrated, authoritative, extendible and shareable information collection of related multimedia education materials. The main issues of concern become easy access, fast efficient retrieval and relevant representation of different content such as text, HTML, PowerPoint, audio, video etc. In this paper, the author describes EduNuggets, an intelligent repository for multimedia educational material. EduNuggets has been designed to support the semantic integration of multimedia content to be distributed over the web. The paper discusses the user interface, tailored to instructors who maintain the collection, and learners who may use several strategies to access this material. The paper also presents a discussion on the two Information Retrieval (IR) techniques implemented in EduNuggets for effective exploration of the multimedia content, namely, Latent Semantic Indexing (LSI) and Naïve Bayesian Classification (NBC). Case studies performed to determine the validity and effectiveness of the NBC algorithm are also elaborated upon. Finally, the author describes a technology called Synchronized Multimedia Integration Language (SMIL) for authoring multimedia applications. In this project, SMIL is used for developing interactive multimedia presentations that include resources such as text, HTML, PowerPoint (PPT) and images. These presentations can then be played on Windows or Unix platforms and can enhance the repertoire of material available to instructors and students. The paper concludes with plans for future research.

Keywords: Novel E-learning interfaces and interactions, Web-based education software, Information Retrieval, Naïve Bayesian Classifier, Latent Semantic Indexing, Multimedia, Synchronized Multimedia Integration Language, Intelligent information and knowledge management systems, Intelligent systems for multimedia presentation, Intelligent visualization tools, Interfaces for the semantic web, Support for collaboration in multi-user environments.

1. Motivation

Traditional practices of college-level teaching and learning are changing in response to the changing profile of learners, technological advances in networking infrastructure and the continuous increase of information available on the web. Learners are becoming increasingly diverse. Traditionally, learners attended courses for continuous periods of time on the campus of their home institution. Now, continuous learning is becoming common place. Learners may take courses from multiple institutions, and often need to adjust the course schedule to fit their personal and professional constraints. Given such a variety of non-converging requirements, being physically present on campus is becoming impractical, and distance education is emerging as a new mode of operation for higher-education institutions. At the same time, instructors increasingly use electronic materials, including lecture notes, software simulations and videos of relevant presentations, to enrich their lectures and to provide more supporting materials to their students for self study. The World Wide Web abounds with information on all conceivable subject matters in a variety of media. Instructors use this resource to identify relevant content and replenish it with new content they create. This content, therefore, presents a great opportunity for fulfilling the learners' need for distributed asynchronous education, and the powerful networking infrastructure makes this opportunity eminently realistic.

A substantial impediment to using the Web as an education resource is the fact that the information available on the Web is not organized in any meaningful scheme. Today, there is limited support for the instructors to maintain an authoritative and coherent view of this material so that they can reuse and extend it, and for the students to use this material creatively and effectively. For example, instructors have to maintain their own collection of materials (i.e., organize their lecture notes and review the validity of their on-line bookmarks). The students can read the provided lecture notes (either on-line or printed), and they may potentially browse the bookmarks or use a search engine to find additional relevant material to learn more about a topic of interest. Unfortunately, the terminology used by the different web documents is often inconsistent and there is no common overarching context for the available materials. Even worse, the various available documents offer inconsistent and even conflicting information, which, lacking an authoritative information evaluation, may cause confusion to the learner. Further, navigation through a large set of independent sources, with different content structure and presentation style, often creates in the learner a feeling of “being lost”, thus drastically limiting the usefulness of the learning material. Finally, since material may be available in a variety of media (HTML, text, PPT, audio, video) and in a variety of formats, students interested in accessing it are required to install and maintain a variety of pieces of software.

There are three key aspects to the overall problem of supporting on-line teaching and learning: (a) providing a means to an authority to evaluate and annotate the available information, (b) providing a coherent context for the organization of the available information, and (c) supporting the learners’ access of this material through various mechanisms appropriate for different learning styles. The objective of the EduNuggets project [1] is to develop a framework with an innovative set of features to address these impediments to the effective use of electronic course materials. EduNuggets is a step towards an intelligent education system where instructor's can capture content and model relevant concepts in the material (their own as well as reference) and student's can browse through the material by issuing queries or navigating through a user interface based on the student's style of learning.

2. Introduction

The EduNuggets framework provides a language for specifying the key concepts of the subject domain and their relations. Through the EduNuggets Instructor application, instructors can model the subject domain, by building and extending topic maps of the domain concepts and their inter-dependencies. These concepts are used to annotate (segments of) the various multimedia documents in the EduNuggets repository, thus providing a coherent overall organization framework for the course material. The repository contains documents developed and owned by the instructor and pointers to existing materials available on the web. The formats currently supported by the repository include text, HTML, PowerPoint, audio/video presentations and movies. To support the instructor's domain-modeling task, the EduNuggets Instructor application enables the automatic construction of a first-cut domain model given a corpus of course materials, using document-clustering methods. To support the learners in effectively accessing the repository information, the EduNuggets Student application provides multiple views of the subject topic maps, and also supports query-based information retrieval through the repository. Students can choose to search the material collection through a series of queries or by visually exploring the connections among the domain concepts and the various nuggets of multimedia information associated with them. At any point in the interaction process, the learners may provide feedback to the EduNuggets tool on the quality of the accessed information, which is used to adapt the tool's behavior. Thus, the learner's interface can be tuned to provide a personalized learning experience for each learner.

Presently, the EduNuggets environment "allows the instructors to (a) define the concepts and relations of interest in the subject domain, (b) add to the environment's repository multimedia (HTML, text, audio etc) content, relevant to the domain of interest, and (c) annotate (segments of) this content with the defined concepts" [1]. The EduNuggets Student application "enables a student to query the repository, view (or hear) specific documents retrieved in response to the query, and browse the repository following the conceptual links of the stored materials on a visual interface" [1].

The EduNuggets application currently provides support for Information Retrieval via the Latent Semantic Indexing algorithm. The primary goal of this project is to include the Naïve Bayesian Classifier as a parallel search engine in order to provide a strong, effective search capability within EduNuggets. The different techniques for Information Retrieval (namely LSI and NBC) are discussed in the context of the EduNuggets framework. LSI is discussed briefly and more attention is paid to the Naïve Bayesian Classifier as an effective approach to IR. Furthermore, different case studies to evaluate the performance of the NBC algorithm are also outlined.

The secondary goal of this project is the incorporation of the SMIL technology into EduNuggets through The SMIL Editor in order to develop interactive multimedia presentations that will become part of the corpus of material already present in the EduNuggets' repository. The SMIL Editor will allow instructors to reuse their Windows-only PowerPoint presentations by converting them into SMIL presentations for viewing on both the Windows and Unix platforms. These presentations can then be used as part of the learning material provided to students. Keeping the above objectives in mind, this document has been divided to support the two main goals, each of which are discussed at length in two main sections "Information Retrieval" and "Multimedia Development in EduNuggets". The rest of this paper is organized as follows. The motivation for this essay is presented in Section II. Related Work is discussed in Section III. The current architectural description of the EduNuggets system is described in Section IV (includes the discussion on Information Retrieval). Section V is devoted to Multimedia Development in EduNuggets. Case studies are discussed in Section VI. Future work is outlined in Section VII. The conclusion is provided in Section VIII and references in Section IX.

3. Related Work

The authors of [8] have developed a groupware application called NuggetMine that " collaborates with a workgroup to increase information nugget sharing among the group. NuggetMine and the workgroup work together to build, maintain, and utilize a repository

– or ‘mine’ – of information nuggets.” Instructors can use the EduNuggets Instructor application to provide the same sort of functionality as offered by NuggetMine. Essentially, instructors can generate nuggets by manually annotating content from their own notes or notes from external sources and develop nuggets out of them which are then stored in the repository. The instructors then can specify interesting relationships between these nuggets and important domain concepts (topics). Both the instructors and students can then view explore this “mine” of nugget information using the EduNuggets instructor and student applications respectively. They will also be able to view related topics.

The authors of [15] have presented work on synchronized multimedia presentations. Their main focus is the adaptation of WWW documents using several techniques and integration with other media components to create interesting presentations. For the adaptation process, the authors discuss different schemes for selecting content from WWW documents. This is similar to the bootstrapping process in EduNuggets where the focus is on the extraction of nuggets (segments relating to important concepts in a domain) and topics (the core concepts) and generating associations between these concepts. In [15], the authors take this selected content and integrate with different media components in time. In EduNuggets, the instructor can create independent presentations that contain text and PowerPoint image components. Future work in EduNuggets will focus on integrating these SMIL presentations with existing nuggets and adding them to the corpus of learning material.

[14] describes techniques for automatically constructing structured multimedia documents from live presentations. The media components include video, images and text. While live presentations are not captured in EduNuggets, the instructor can include previously generated video clips as part of the repository of domain-specific information. These video clips can be played in the SMIL-adaptable browser in the EduNuggets Student application. Furthermore, The SMIL Editor allows instructors to include PowerPoint presentation slides and annotated text in multimedia presentations, which are also provided to students as part of the learning material. [14] mainly focuses on issues of synchronization of captured data and automatic editing.

In [13], the authors have contrasted various teaching and learning styles such as web notes, presentations, lecture handouts etc. and augmented this captured information via ubiquitous computing technology such as PDAs, laptops, videos etc. The authors discuss various phases in their project namely pre-production, live recording and post-production. The pre-production and post-production phases are very similar to tasks carried out by the instructor and student in using the EduNuggets framework. The pre-production phase in [13] consists of converting documents into different formats, highlighting important piece of information etc. In EduNuggets, the instructor carries out a similar task where they highlight segments of documents to create nuggets (pre-processing is also performed on the documents). The post-production phase in [13] involves generating multimedia presentations using annotated presentation slides, web notes, video segments, audio clips etc. In EduNuggets, the instructor can use The SMIL Editor for developing interesting presentations with different media components.

[16] presents a list of the top 10 important issues in dealing with Information Retrieval. Among this list of important issues, [16] lists ‘Integrated Solutions’ as the most important. The author points out that developing an information retrieval system for a particular application requires different ‘retrieval’ components for different types of data. In EduNuggets, two different Information Retrieval algorithms have been included, namely, Latent Semantic Indexing and Naïve Bayesian Classifier. The LSI returns search results for a query by looking at the semantics of stored documents. The NBC on the other hand returns search results based on a probability model where the probability of the search query belonging to a set of documents is high. [16] puts ‘Efficient, Flexible Indexing and Retrieval’ at number 3 on the top 10 list. Keeping in time with the importance of efficient retrieval, different case studies have been setup in this project as well to determine the accuracy of the Naïve Bayesian classifier. [16] also focuses on ‘the wide variety of document formats’ that are available today and should be included when thinking about information retrieval systems. With this in mind, the EduNuggets framework currently supports text, HTML, audio, video and SMIL presentations.

4. Architectural description of the EduNuggets Framework

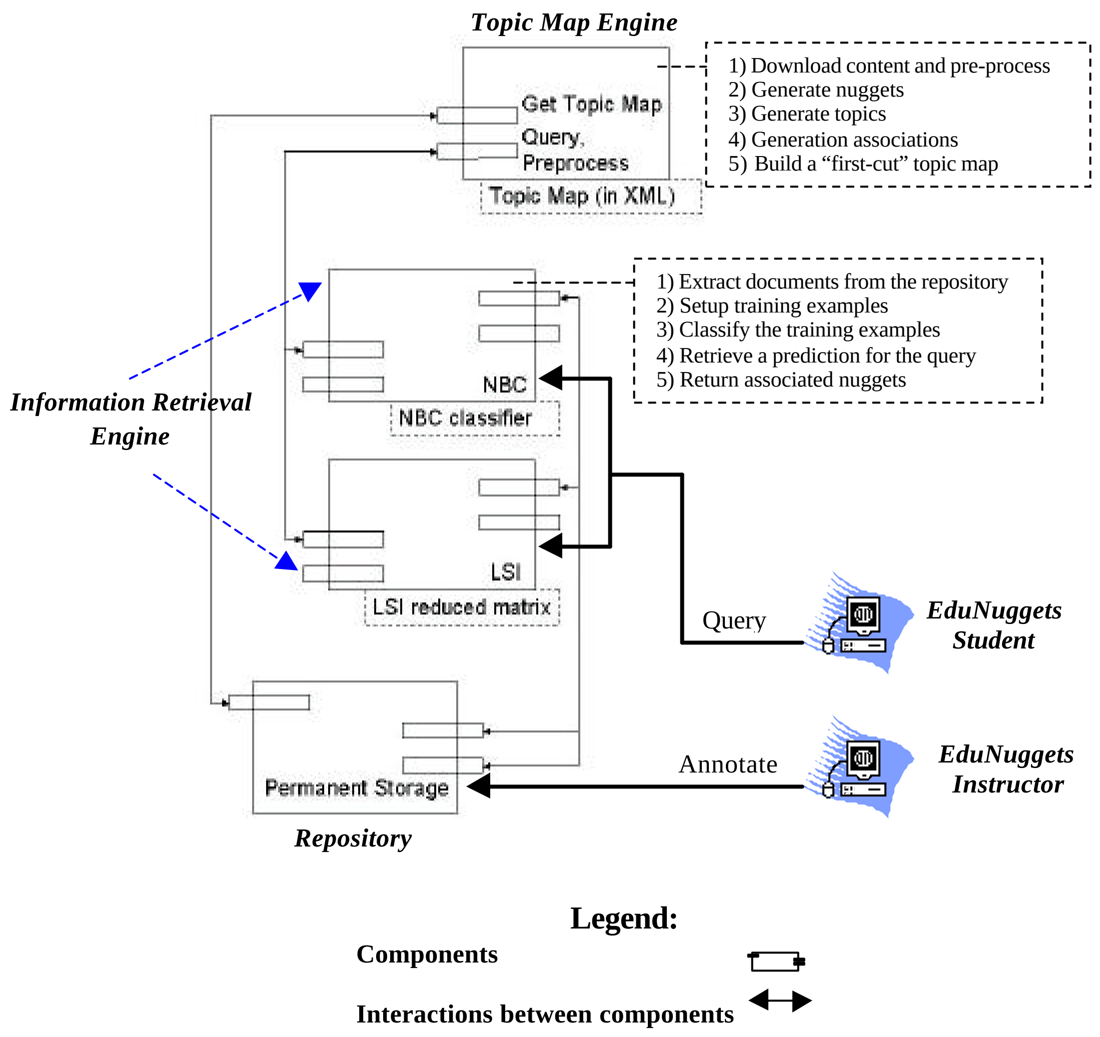

Topic Map Engine
1) Download content and pre-process
2) Generate nuggets
3) Generate topics
4) Generation associations
5) Build a "first-cut" topic map
Get Topic Map
Query,
Preprocess
Topic Map (in XML)
1) Extract documents from the repository
2) Setup training examples
3) Classify the training examples
4) Retrieve a prediction for the query
5) Return associated nuggets
Information Retrieval Engine
NBC
NBC classifier
LSI
LSI reduced matrix
Query
EduNuggets Student
Annotate
EduNuggets Instructor
Permanent Storage
Repository
Legend:
Components
Interactions between components

Figure 1 – Overall architecture of the EduNuggets framework

The architecture of the overall EduNuggets framework is depicted in Figure 1 and consists of the following components:

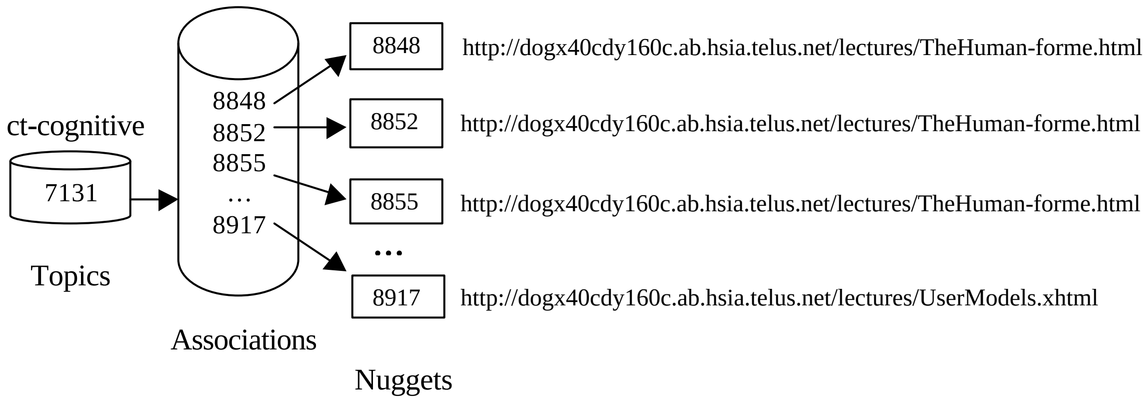
- The *repository* (database) containing the instructor's documents, URL pointers and *topic-map* model;
- The topic-map engine responsible for extracting a domain topic map given a document collection and for visualizing the topic map in the two client applications;
- Two thick-client applications for students and instructors, namely the EduNuggets Student application and the EduNuggets Instructor application, as part of the

EduNuggets framework;

- The *information-retrieval* engine, consisting of two components, one based on the Latent Semantic Indexing algorithm and a second based on the Naïve Bayesian Classification algorithm.

4.1 The Repository

The EduNuggets repository contains pointers to the documents that the instructor has identified as relevant to the domains he/she has defined. These pointers refer to the file system on which the repository resides, for the documents that the instructor has developed and to URLs for these documents that he/she has found on the Web. As has already been mentioned, the documents can be HTML, text, audio/video presentations and SMIL presentations. The repository also includes the topic map that the instructor has developed to model the domain of interest. The grounded topics include references to their Nuggets, i.e., to their corresponding document segments. Figure 2 shows a view of the EduNuggets repository with a sample topic and associated nuggets.



ct-cognitive
7131
Topics
8848
8852
8855
…
8917
Associations
8848
8852
8855
…
8917
Nuggets
http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html
http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html
http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html
http://dogx40cdy160c.ab.hsia.telus.net/lectures/UserModels.xhtml

Figure 2 – A view of the EduNuggets Repository

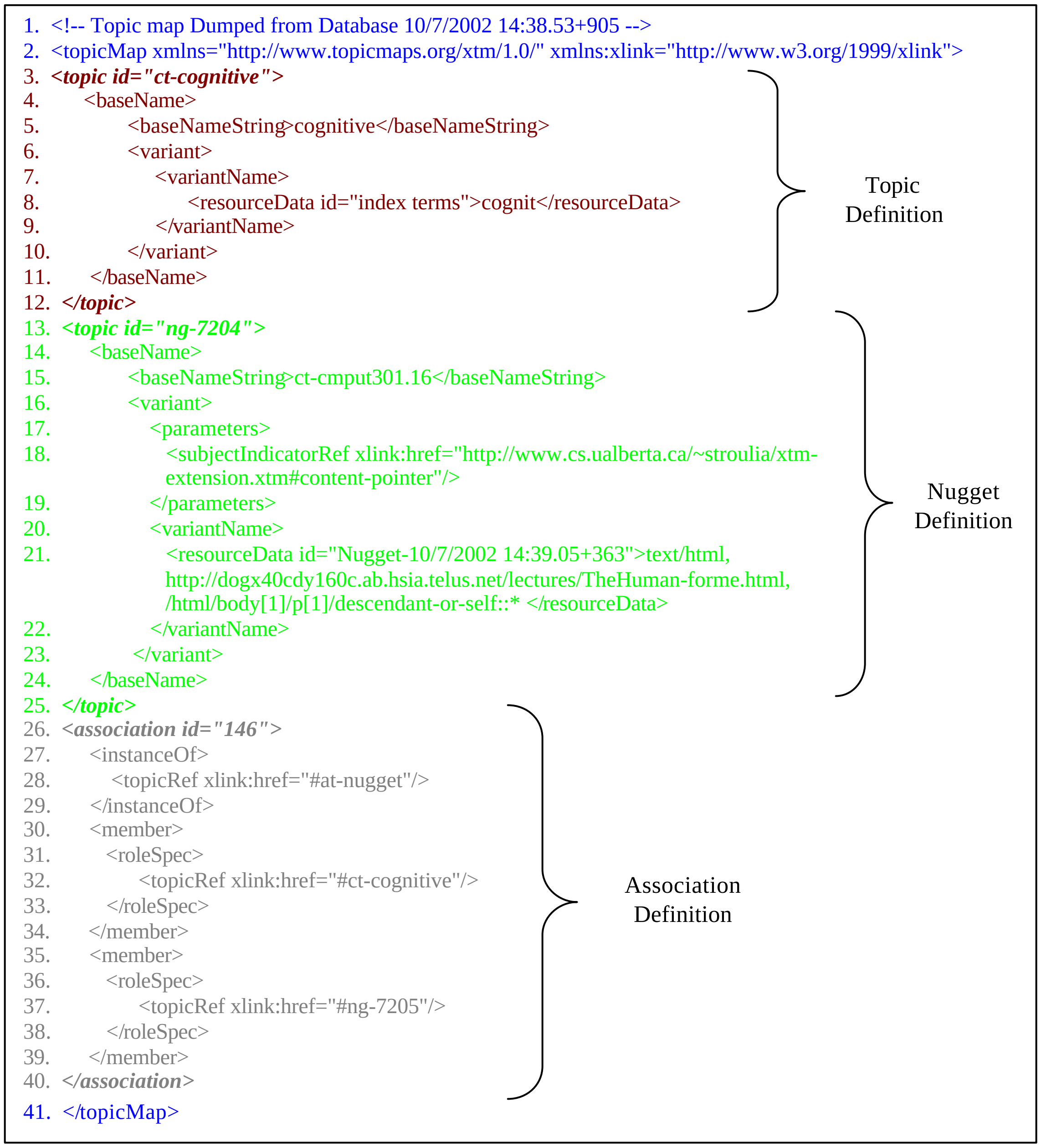
In Figure 2, the Topics and Associations tables are shown as cylinders. The associated nuggets are represented as boxes. 'ct-cognitive' is shown to be a topic in the Topics table with id as 7131. This topic is related to several nuggets via the Associations table. For e.g., 'ct-cognitive' is associated with nuggets 8848, 8852, 8855 and so on. Each of these nuggets in turn is a segment of a document which can be found at the corresponding URL. In the scenario where the URL is the same for a group of nuggets, the Nuggets table will hold specific information about that nugget such as the location of the segment within the document from which the nugget was formed etc. As such, a topic can be associated with different types of nuggets each of which can originate from the same document or from different documents.

Another example showing topic-nugget associations might have “Human Computer Interaction” as a topic and its subtopics as “the user”, “the computer”, and “Graphical User Interfaces”. Let us assume that the first of these subtopics, “the user” is grounded, i.e., it contains a reference to a segment of an HTML document, edited by the instructor discussing how user attributes should be considered in designing the human-computer interaction. The repository would then contain associations among the topics and corresponding nuggets, and these associations may represent relationships such as “specialization of”, “depends on”, “is composed of” etc. Furthermore, in our example domain, the topic of “the computer” might “be composed of” topics such as “the input device”, “the output device”, and the “memory storage” etc.

4.2 The Domain Model as a Topic Map

The model of the subject domain is defined in terms of *Topic Maps* [4]. Topic Maps describe conceptual structures, their associations and their mapping to each other and to resources, which are real-world media objects. In EduNuggets, topic maps are constructed using nuggets, topics and associations stored in the repository. The instructor’s own lecture notes as well as the related bookmarks that he/she may have identified on the Web become resources associated with concepts and associations in the subject Topic Map.

Topic Maps offer several advantages: they can be visualized to provide a visual map of the domain resources and their relations to the student. In this way, the student can navigate through the course-related materials and continuously perceive his/her navigation path without getting lost. Furthermore, since the topic maps schema is an XML-based representation, it is quite general and platform-independent. Figure 3 shows a sample EduNuggets topic map.



```
1.  <!-- Topic map Dumped from Database 10/7/2002 14:38.53+905 -->
2.  <topicMap xmlns="http://www.topicmaps.org/xtm/1.0/" xmlns:xlink="http://www.w3.org/1999/xlink">
3.  <topic id="ct-cognitive">
4.      <baseName>
5.            <baseNameString>cognitive</baseNameString>
6.            <variant>
7.                <variantName>
8.                     <resourceData id="index terms">cognit</resourceData>
9.                </variantName>
10.           </variant>
11.     </baseName>
12. </topic>
13. <topic id="ng-7204">
14.     <baseName>
15.           <baseNameString>ct-cmput301.16</baseNameString>
16.           <variant>
17.               <parameters>
18.                 <subjectIndicatorRef xlink:href="http://www.cs.ualberta.ca/~stroulia/xtm-
                    extension.xtm#content-pointer"/>
19.               </parameters>
20.               <variantName>
21.                 <resourceData id="Nugget-10/7/2002 14:39.05+363">text/html,
                    http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html,
                    /html/body[1]/p[1]/descendant-or-self::* </resourceData>
22.               </variantName>
23.            </variant>
24.     </baseName>
25. </topic>
26. <association id="146">
27.     <instanceOf>
28.        <topicRef xlink:href="#at-nugget"/>
29.     </instanceOf>
30.     <member>
31.       <roleSpec>
32.            <topicRef xlink:href="#ct-cognitive"/>
33.       </roleSpec>
34.     </member>
35.     <member>
36.       <roleSpec>
37.            <topicRef xlink:href="#ng-7205"/>
38.       </roleSpec>
39.     </member>
40. </association>
41. </topicMap>
```

Topic Definition
Nugget Definition
Association Definition

Figure 3 – Sample EduNuggets Topic map

From Figure 3 we can see that an EduNuggets topic map consists of definitions for nuggets, topics and associations. The topic map has several components [4]:

1. Statement 2 declares a default namespace for the elements in the topic map file using the xmlns attribute.

2. Statements 3 – 12 provide a definition for a topic. The <topic> tag gives the name and occurrence characteristics of a single topic. The name of the topic is specified using the <baseName> tag. Since the base name is a string, it is entered as part of the <baseNameString> tag. The <variantName> tag inside the <variant> gives an alternative name for the base name. In our example, the topic has base name ‘cognitive’ and the variant name is ‘cognit’, the stemmed form of ‘cognition’.
3. Statements 13 – 25 provide a definition for a nugget. In EduNuggets, nuggets are also stored as topics. Therefore, the nugget description is provided inside the <topic> tag. Just like the topic ‘cognitive’, the nugget’s name is described using the <baseName> tag. The nugget has additional information in the <parameters> tag. The <parameters> tag expresses appropriate processing context for the variant name of a topic. The sample nugget in Figure 3 is ‘ct-cmput301.16’.
4. Statements 26 – 40 provide a definition for an association. The <association> tag asserts a relationship among topics. The class to which an <association> belongs is specified by an <instanceOf> child element. The <member> element specifies all the topics that play a role in the association. In Figure 3, the <roleSpec> tag defines the roles played by the topics. The ‘at-nugget’ defines the association relationship between the topic and nugget.

The topic map is constructed by the instructor, through the EduNuggets Instructor client application. Knowledge-acquisition processes such as domain modeling of this type are notorious for their difficulty and the time investment they require on the part of domain experts. To simplify the domain-expert’s, i.e., the instructor’s, task, the EduNuggets environment contains a “domain model bootstrapping” component. The bootstrapping component is responsible for constructing a first-draft domain model, given the instructor’s document collection for this domain. Using document-analysis techniques, this component extracts the domain-model topics and their associations. The steps involved in the “domain model bootstrapping” process are described next.

4.2.1 Downloading content and document pre-processing

Initially, an instructor wishing to use the EduNuggets system specifies the material to be used as resources. A crawler crawls through the resources directory and downloads a copy of each file, if it is not already in the home file system. Any URLs contained in the documents (or the set of bookmarks) will also be fetched and the information from the corresponding web sites will be downloaded (one-step crawling).

Once crawling has completed, the collected documents are pre-processed: stop words and punctuation marks are removed and all words are stemmed to their base form without any affixes. The processed documents can then be considered as the base set of nuggets for the domain.

4.2.2 Generating Nuggets

All downloaded document files are saved in the repository as a set of nuggets. If a document is very large, it is better to construct nuggets from segments in the documents. EduNuggets uses a similar approach where each paragraph in an HTML or text file is created into a nugget. Since a nugget corresponds, as a whole, to a topic in the domain, it is important to process these documents in order to identify likely coherent segments. Figure 4 shows some sample fields for records in the nuggets' table that were created by the bootstrapping process and entered into the repository. Some of this information is visible to the student on the EduNuggets Student application when they perform a search query.

```
>> The Nuggets' table in the repository
Column               |       Type          |
---------------------+----------------------
 nugget_key          |  integer            |
 dt_key              |  integer            |
 nugget_dt_key       |  integer            |
 medium              |  character          |
 url                 |  text               |
 selection_specifier |  text               |
 selection_content   |  text               |
--------------------------------------------


>> Sample fields from the Nuggets' table. Note: some fields have been removed.
nugget_dt_key |  medium   |                         url                                                  |
selection_specifier                        |                  selection_content
-----------------------------------------------------------------------------------------------------------------------------
8848             | text/html | http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html       |
/html/body[1]/p[2]/descendant-or-self::* | understanding the user involves understanding
how the human works  which is the subject matter of cognitive psychology

8852             | text/html | http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html       |
/html/body[1]/p[6]/descendant-or-self::* | cognitive models
```

Figure 4 – Sample view of the Nuggets' table

Following our example from Figure 2, we can see that the nugget information provided in the picture above corresponds to the topic 'ct-cognitive'. In particular, nuggets' 8848 and 8852 are shown. Important pieces of information to note are the URLs (specified by the field url) and the segment of document (specified by the field selection_content) that comprises the nuggets. When a student searches for information on cognition, EduNuggets will return results that contain the above two nuggets. Furthermore, the student will be able to click on the returned results, and the document content will then be rendered with the nugget content highlighted (the section on the EduNuggets Student application explains this process in greater detail).

4.2.3 Generating Topics

After the domain nuggets have been generated, topics are created automatically: they are extracted from the headings in the documents. Intuitively, an instructor will specify the most important concepts of a document in a heading. Other policies can be used such as most frequently occurring words found in delimiters such as <p> tags, <hr> tags etc for HTML documents. Statistical keyword-extraction tools, such as Rainbow [2], have also been tested for topic generation. However, topic words generated by Rainbow are based on keyword frequency and often these words may not embody the true context of the document. Figure 5 gives a snapshot of the Topics view in the repository.

```
>> The Topics' view in the repository.
Column               |         Type          |
---------------------+-----------------------
 dt_key              | integer               |
 domain_name         | character             |
 topic_id            | character             |
 basename            | text                  |
 index_terms         | text                  |
 is_nugget           | boolean               |
-----------------------------------------------

>> A record from the Topics' view for topic 'ct-cognitive'.
dt_key | domain_name |   topic_id    | basename  | index_terms | is_nugget
--------+-----------------+-------------+-----------+---------------+-----------
7131    | cmput301       | ct-cognitive | cognitive  | cognit          | f
```

Figure 5 – A snapshot of the Topics' view

Since all relevant topic words are stemmed to their base form during topic generation, search queries such as 'cognition', 'cognitive' or 'cognitively' etc. will all be stemmed to 'cognit'. Therefore, only one topic matching the stemmed form is created during the bootstrapping process. As mentioned before, topic words are picked from document headings and verified against a stop-word list. The stop-word list contains common words such as a, an, the etc. and numbers. These words do not form meaningful topics and are pruned from the list of possible topics. In addition, references to URLs or email addresses occurring in document headings are also pruned after performing string comparison for the relevant patterns.

4.2.4 Generating Associations

The final step in the domain-model bootstrapping process is the association generation. An association is generated between two topics when there are nuggets for these topics based on the same underlying document. These are simple "related" associations, however, deeper associations such as specialization may also be extracted. Currently the draft domain model constructed by the bootstrapping process is flat, consisting of a layer of topics, their associated nuggets and their relations among them. Figure 6 shows records from the Associations' view in the repository.

```
>> The Associations' view in the repository.
Column                 |        Type           |
-----------------------+------------------------
 domain_name           | character             |
 ass_key               | integer               |
 type_dt_key           | integer               |
 from_dt_key           | integer               |
 to_dt_key             | integer               |
------------------------------------------------

>> A snapshot from the Associations' view for topic 'ct-cognitive'
domain_name | ass_key | type_dt_key | from_dt_key | to_dt_key
-----------------+----------+---------------+----------------+-----------
 cmput301    |  14104 |      6841     |      7131      |     8848
 cmput301    |  14112 |      6841     |      7131      |     8852
 cmput301    |  14114 |      6841     |      7131      |     8855
 cmput301    |  14130 |      6841     |      7131      |     8863
 cmput301    |  14137 |      6841     |      7131      |     8868
 cmput301    |  14155 |      6841     |      7131      |     8910
 cmput301    |  14161 |      6841     |      7131      |     8917
```

Figure 6 – A snapshot of the Associations' view

As can be seen in Figure 6, the topic 'ct-cognitive' with id 7131 has associations with 7 nuggets. This relationship is represented via the type_dt_key field. For example, the association relationship may be an "is-a" type relationship. The associations are established from topics to nuggets (i.e., from the from_dt_key to the to_dt_key in the above picture). The information presented in Figure 6 can be visualized from Figure 2 as well.

4.2.5 Building a Topic map

A topic map is then constructed from the nuggets, topics and associations found in the repository. This draft "Version 0" topic map can be further modified by the instructor. Using the EduNuggets Instructor thick-client application, the instructor may review the extracted topics, delete uninteresting ones, organize the remaining in specialization hierarchies, and define other domain-specific relations of interest.

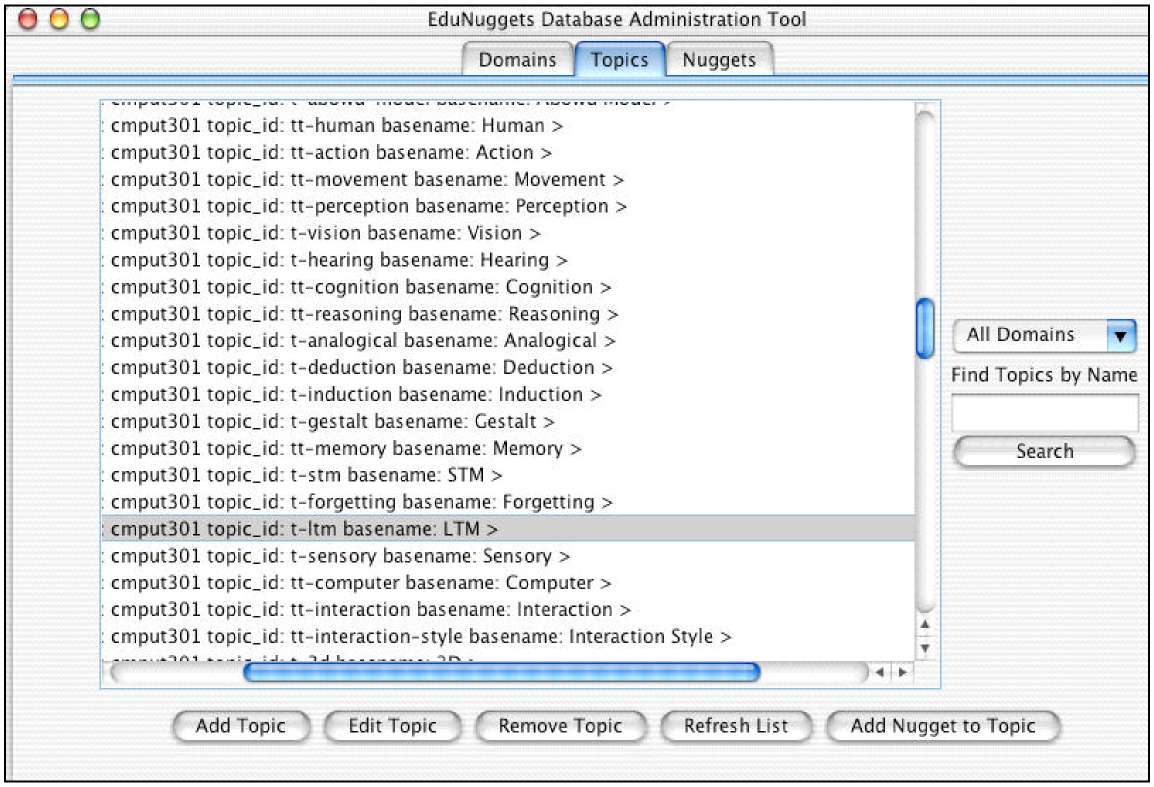
The EduNuggets Instructor and Student applications contain the same topic-map visualization component that displays a topic-map as a hypergraph, which can be focused on a user-selected node. In the context of the instructor application, this visualization component enables the instructor to review the current topic map in order to decide about desirable edits. In the context of the student application, the component (shown in panel D of the snapshot of Figure described later) enables the student to visually explore the domain model, navigating from one concept to another through attached relations and to browse the nuggets associated with them.

4.3 The EduNuggets Framework

The EduNuggets framework consists of two applications: the EduNuggets Instructor application and the EduNuggets Student application. The Instructor application enables the instructor to define the concepts and relations of interest in the subject domain, add to the repository new multimedia documents or pointers to URLs of documents (HTML, text, audio etc) relevant to the domain of interest, and annotate (segments of) this content with the defined concepts. The Student application enables a student to query the repository, view (and listen to) specific documents retrieved in response to the query, and browse the repository following the conceptual links of the repository materials on a visual interface.

4.3.1 The EduNuggets Instructor application

Figure 7 shows a snapshot of the EduNuggets Instructor user interface. The main purpose of this application is to enable the instructors to collect high-quality materials that they have developed or they have discovered on the Web, and to provide an overall organization context for them. Figure 7 presents a view of the Instructor user interface, as the instructor reviews the concepts (topics) and the nuggets of a domain, as they are currently stored in the repository. For example, the domain of interest in Figure 7 is “CMPUT 301” and several topics have been constructed for that domain such as t-vision (the “Vision” topic), t-STM (the “Short-Term Memory” topic) etc.



```
cmput301 topic_id: tt-human basename: Human >
cmput301 topic_id: tt-action basename: Action >
cmput301 topic_id: tt-movement basename: Movement >
cmput301 topic_id: tt-perception basename: Perception >
cmput301 topic_id: t-vision basename: Vision >
cmput301 topic_id: t-hearing basename: Hearing >
cmput301 topic_id: tt-cognition basename: Cognition >
cmput301 topic_id: tt-reasoning basename: Reasoning >
cmput301 topic_id: t-analogical basename: Analogical >
cmput301 topic_id: t-deduction basename: Deduction >
cmput301 topic_id: t-induction basename: Induction >
cmput301 topic_id: t-gestalt basename: Gestalt >
cmput301 topic_id: tt-memory basename: Memory >
cmput301 topic_id: t-stm basename: STM >
cmput301 topic_id: t-forgetting basename: Forgetting >
cmput301 topic_id: t-ltm basename: LTM >
cmput301 topic_id: t-sensory basename: Sensory >
cmput301 topic_id: tt-computer basename: Computer >
cmput301 topic_id: tt-interaction basename: Interaction >
cmput301 topic_id: tt-interaction-style basename: Interaction Style >
```

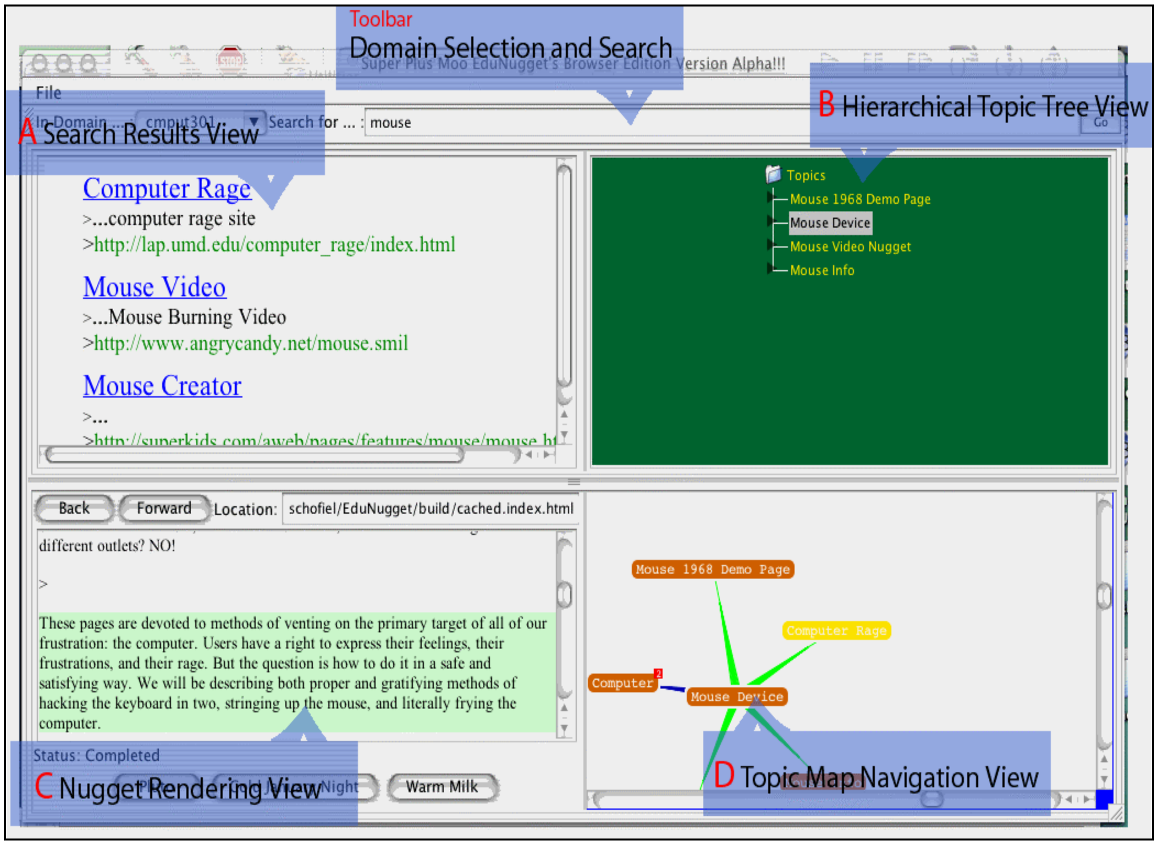
Figure 7 - The EduNuggets Instructor application

Using this application, instructors specify new subject domains. A subject domain is a knowledge area of interest to the instructor, for which he/she wishes to collect relevant electronic materials. Instructors may start with an empty domain and specify the domain's concepts and their relations. The underlying meta-model for the domain specification is topic maps [4], already described above. This meta-model is essentially a semantic-network representation, supporting generalization-specialization relationships among concepts as well as domain-specific associations. Furthermore, it supports concept "grounding" by association to documents, and it is this feature that enables the use of topic maps as an organization framework for a document collection.

Currently, the EduNuggets framework supports the collection of HTML, text, audio/video and SMIL documents in the repository. The Instructor application also includes a plug-in that enables the conversion of PowerPoint presentations to SMIL so that they can be accessible on all platforms (via the SMIL Editor). Documents of all these formats can provide grounding to the domain-model concepts, and, learners can subsequently access them through these concepts. Because documents are often big in size and cover more than a single concept of interest, the EduNuggets Instructor application enables the instructor to identify a small segment of the document as corresponding to a concept of interest. This segment constitutes a nugget. In EduNuggets, topics prefixed by a 't-' indicate a head topic. Topics prefixed by a 'tt-' indicate sub-topics or specializations of a head topic.

4.3.2 The EduNuggets Student application

Once the instructor has built a document collection and organized it using a domain model, students can access this content using the EduNuggets Student application. Figure 8 provides a view of the EduNuggets Student user interface.



Computer Rage
>...computer rage site
>http://lap.umd.edu/computer_rage/index.html

Mouse Video
>...Mouse Burning Video
>http://www.angrycandy.net/mouse.smil

Mouse Creator
>...
>http://superkids.com/aweb/pages/features/mouse/mouse.ht

Location: schofiel/EduNugget/build/cached.index.html

different outlets? NO!

>

These pages are devoted to methods of venting on the primary target of all of our frustration: the computer. Users have a right to express their feelings, their frustrations, and their rage. But the question is how to do it in a safe and satisfying way. We will be describing both proper and gratifying methods of hacking the keyboard in two, stringing up the mouse, and literally frying the computer.

Status: Completed

Figure 8 - The EduNuggets Student application

The main purpose of this application is to provide an integrated user interface to students through which to access the collected content. The interface provides multiple access mechanisms, and each student may select the mechanism more appropriate to his/her learning style. Furthermore, student can also provide feedback to the application regarding the quality of the documents they access. The EduNuggets Student application consists of a toolbar and four separate panels:

- The Toolbar at the top is used for Domain Selection and Searching.
- The top-left panel (A) is a view of the search results returned by EduNuggets for a specific user query.
- The top-right panel (B) provides a tree-view of the retrieved topic and its neighboring topics.
- The bottom-left panel (C) is a browser, where all retrieved resources are displayed e.g. text or HTML documents.
- The bottom-right panel (D) provides a graphical visualization of the domain topic map.

The different panels can be resized providing students with the ability to customize their application view.

Typically, a student selects the domain of interest from a drop-down menu and inputs a search query. The EduNuggets search engine retrieves the most relevant topics and displays them to the student via panel A. Panels B, C and D are also focused to this result, evaluated to likely be the most relevant to the query. The student may then choose to view the documents corresponding to the topic by clicking on it (either in panel A, B or D). The resource document will be rendered in panel C. Alternatively; the student might navigate to related topics, using the hierarchical view of panel B or the graph-based view of panel D.

Finally, the student might, at any point, issue a new query or provide feedback on the retrieved resource. The sample Student application in Figure 8 shows the feedback buttons as “Pluto” to indicate a poor result, “Cold Rainy Night” to indicate an average result and “Warm Milk” to indicate a good result (These metaphors will be revised soon).

If the student is satisfied with the search results, they can proceed to search for more information or exit the EduNuggets application. On the other hand, if the student is dissatisfied with the search results, he/she may provide some feedback via the feedback buttons on the Student application. This feedback will be considered the next time the student searches for the same query and a different set of results will be returned. The search results displayed in Panel A are a combination of results produced by the Latent Semantic Indexing algorithm and the Naïve Bayesian Classification algorithm.

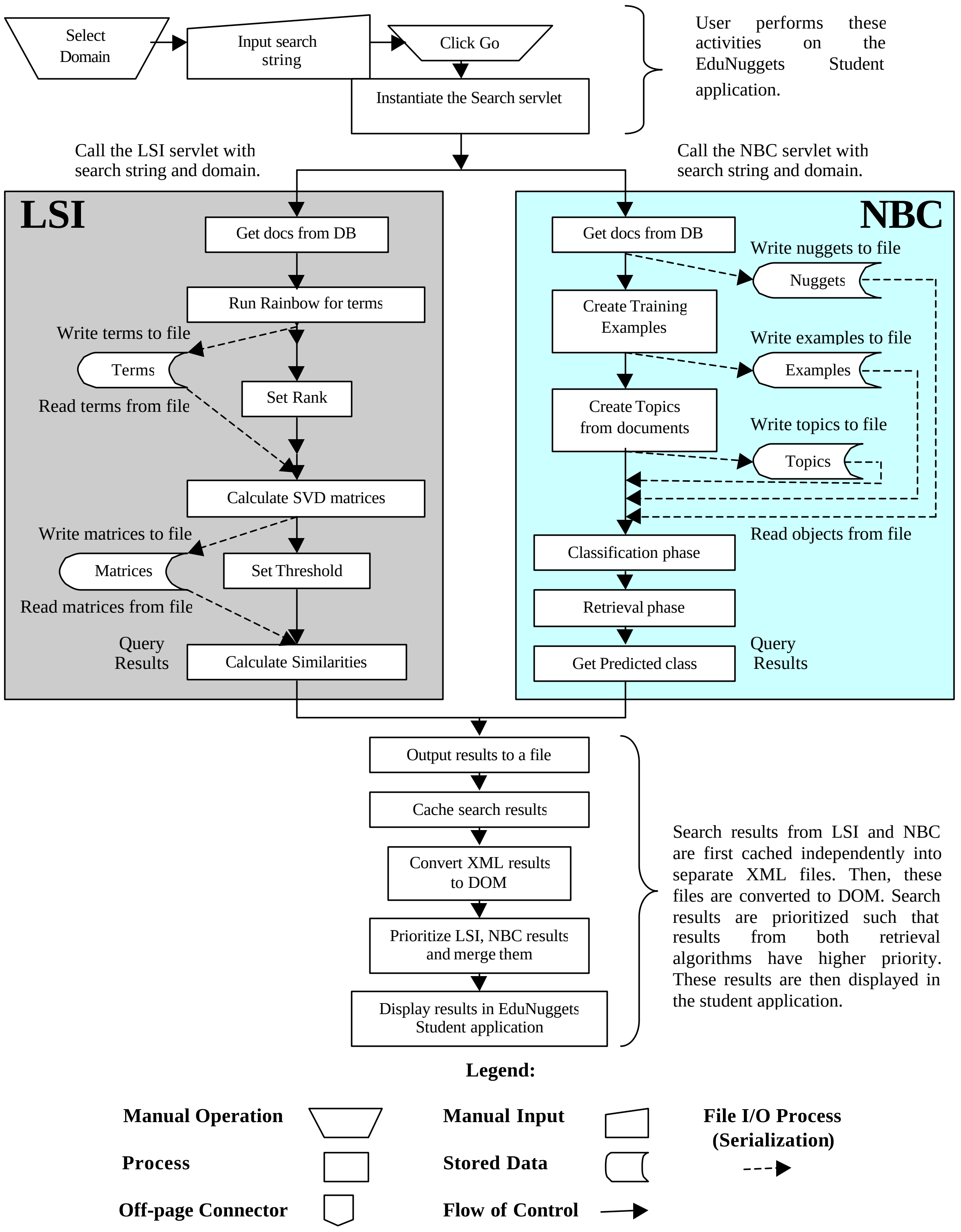

Select Domain
Input search string
Click Go
Instantiate the Search servlet
User performs these activities on the EduNuggets Student application.
Call the LSI servlet with search string and domain.
Call the NBC servlet with search string and domain.
LSI
Get docs from DB
Run Rainbow for terms
Write terms to file
Terms
Read terms from file
Set Rank
Calculate SVD matrices
Write matrices to file
Matrices
Set Threshold
Read matrices from file
Query Results
Calculate Similarities
NBC
Get docs from DB
Write nuggets to file
Nuggets
Create Training Examples
Write examples to file
Examples
Create Topics from documents
Write topics to file
Topics
Read objects from file
Classification phase
Retrieval phase
Query Results
Get Predicted class
Output results to a file
Cache search results
Convert XML results to DOM
Prioritize LSI, NBC results and merge them
Display results in EduNuggets Student application
Search results from LSI and NBC are first cached independently into separate XML files. Then, these files are converted to DOM. Search results are prioritized such that results from both retrieval algorithms have higher priority. These results are then displayed in the student application.
Legend:
Manual Operation
Manual Input
File I/O Process (Serialization)
Process
Stored Data
Off-page Connector
Flow of Control

Figure 9 – Information Retrieval in the EduNuggets Student application

4.4 Information Retrieval

In addition to exploring the repository knowledge through the visualized topic map, the EduNuggets student application enables learners to directly access information of interest through a query interface. The information-retrieval component consists of a Latent Semantic Indexing and a Naïve Bayes Classifier component. Figure 9 shows the Information Retrieval process in the EduNuggets Student application.

4.4.1 Latent Semantic Indexing

4.4.1.1 Definition

Latent Semantic Indexing is a conceptual-indexing technique that extracts the underlying or "latent" semantic structure of a word pattern using statistical techniques for matching against a given search query as opposed to traditional lexical matching algorithms. This enables the user to perform a retrieval query such as "computer" which returns results containing "computer" as well as "laptop" due to the latent semantic similarity in the word pattern. The underlying intuition is that if two terms are related they will co-exist in a large number of documents, and if a query refers to one of the terms, documents containing the related term are likely to be relevant, even if they do not contain the query term.

4.4.1.2 Process

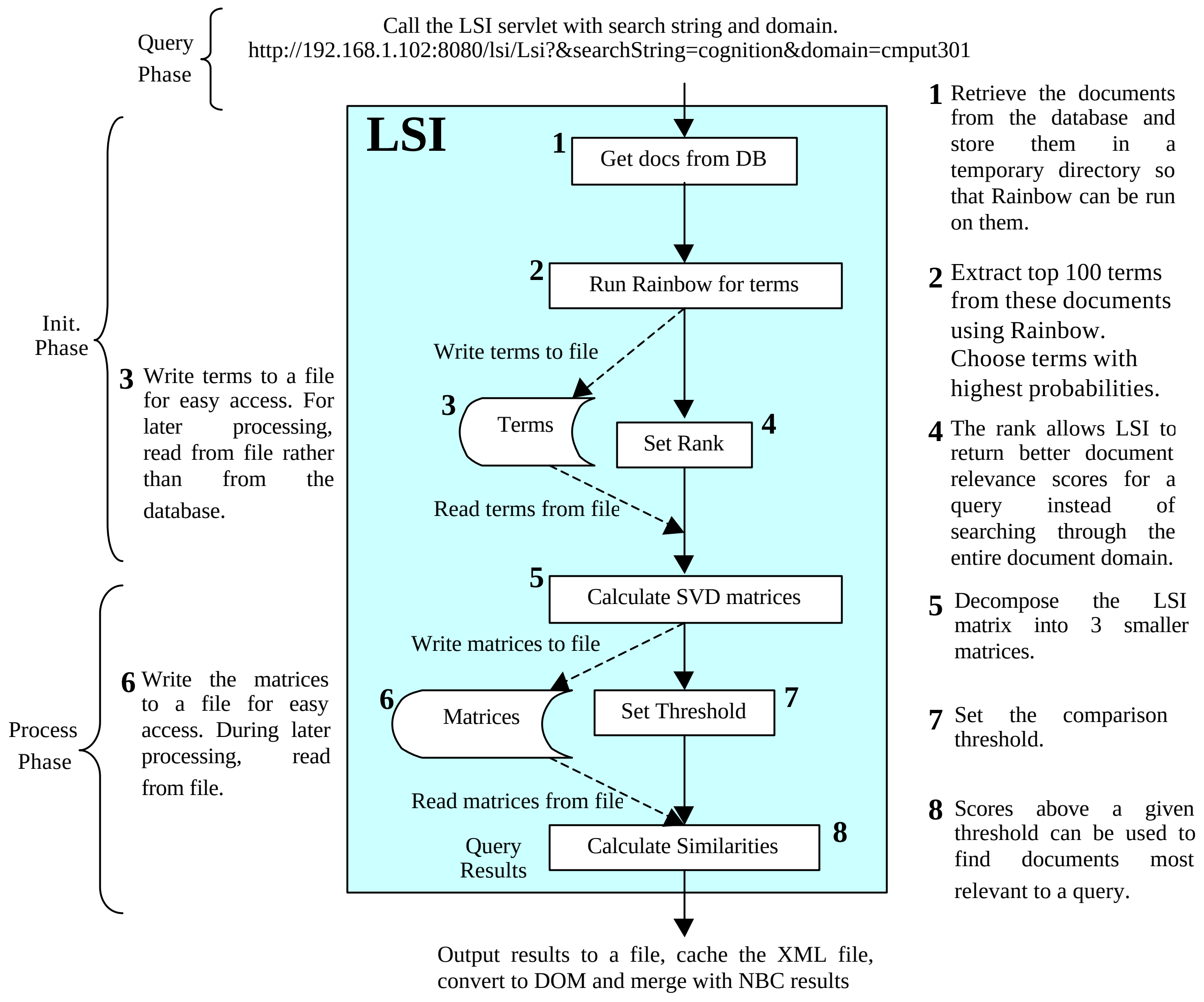
Each document is first represented as a vector of terms weighted by the frequency of each term in the document. Then, a term-by-document matrix is built to represent the entire text collection. LSI uses Singular Value Decomposition (SVD) to decompose the term-document matrix as the product of three smaller matrices. A user-defined parameter, the rank, specifies how much the original term-document matrix needs to be reduced. The three new matrices store all the information of the text collection and are used in the retrieval process. When the user issues a search query it is first converted into a vector of terms and then projected into the LSI semantic space for comparison against every document in the text collection. The algorithm returns the top N most documents similar to the query or all the documents with a degree of similarity above a pre-defined threshold.

Our LSI approach uses singular-value decomposition where a large term-document matrix is converted into a “semantic” space where terms and documents that are closely associated are placed near one another. Using SVD, the arrangement of space reflects the major associative patterns in the data and ignores the smaller, less important influences. When a user specifies a query, the terms in the query are used to identify a point in space and documents in its neighborhood are returned to the user.

In EduNuggets, the LSI algorithm is implemented via a servlet. Figure 10 (similar to the LSI box in Figure 9(a)) shows the processing of the LSI algorithm in the EduNuggets Student application. Figure 10 is annotated further with descriptions about the various steps involved in the querying, initialization and processing phases in LSI.

The LSI servlet takes a search string and a domain as inputs and retrieves relevant documents after stemming the input query and projecting it into the reduced matrix. The documents most similar to the query are selected as search results. The retrieved documents may or may not contain the query terms, and this is exactly the advantage of LSI-based retrieval over lexical-based retrieval.

4.4.1.3 *Implementation*



1 Retrieve the documents from the database and store them in a temporary directory so that Rainbow can be run on them.

2 Extract top 100 terms from these documents using Rainbow. Choose terms with highest probabilities.

3 Write terms to a file for easy access. For later processing, read from file rather than from the database.

4 The rank allows LSI to return better document relevance scores for a query instead of searching through the entire document domain.

5 Decompose the LSI matrix into 3 smaller matrices.

6 Write the matrices to a file for easy access. During later processing, read from file.

7 Set the comparison threshold.

8 Scores above a given threshold can be used to find documents most relevant to a query.

Figure 10 – Annotated view of the LSI algorithm

4.4.2 Naïve Bayesian Classifier

4.4.2.1 Definition

Naïve Bayesian classification is another generally used method for document classification and retrieval. The classification process involves defining functions, models and rules based on the attributes of known class labels in order to predict class membership for new examples. The underlying intuition is that, given a set of documents assigned to a class, the probability of the words contained in these documents being indicators of this class label can be calculated using the Naive Bayesian assumption. Consequently, the probability that new documents containing these words belong to the same class can be calculated.

More specifically, let us assume that a new document D_{new} containing the words (w_1, w_2, … , w_n) has to be classified, in a set of classes C. Furthermore, let C_i be one of the classes in C. The classifier aims at maximizing the probability $P(C_i| w_1, w_2, \dots, w_n)$ in order to classify D_{new} in C_i. The following theorem is used in predicting a class label (topic) for a new example - $P(C_i|x) = P(x| C_i) * P(C_i)$, where:

- $P(x | C_i)$ - conditional probability for attribute x in class label C_i where C_i belongs to a set of class labels C
- $P(C_i)$ – prior probability of class label C_i
- $P(C_i | x)$ – posterior probability of C_i given the attribute x.

4.4.2.2 Process

The Naive Bayesian classifier requires as input labeled, i.e., pre-classified examples. The topics in the EduNuggets repository form the class labels while the documents form the examples. Each nugget is built into a training example, possibly assigned multiple class labels. The class labels for these nuggets are also stored as part of the training information along with a term-frequency matrix (terms are keywords in the document).

4.4.2.3 Implementation

When a student inputs a query in the EduNuggets Student application, the Naive Bayesian classifier servlet is called in parallel with the LSI servlet, with the search query and domain. Figure 11 (similar to the NBC box in Figure 9(a)) shows the processing of the NBC algorithm in the EduNuggets Student application with descriptions for the querying, initialization and processing phases in NBC.

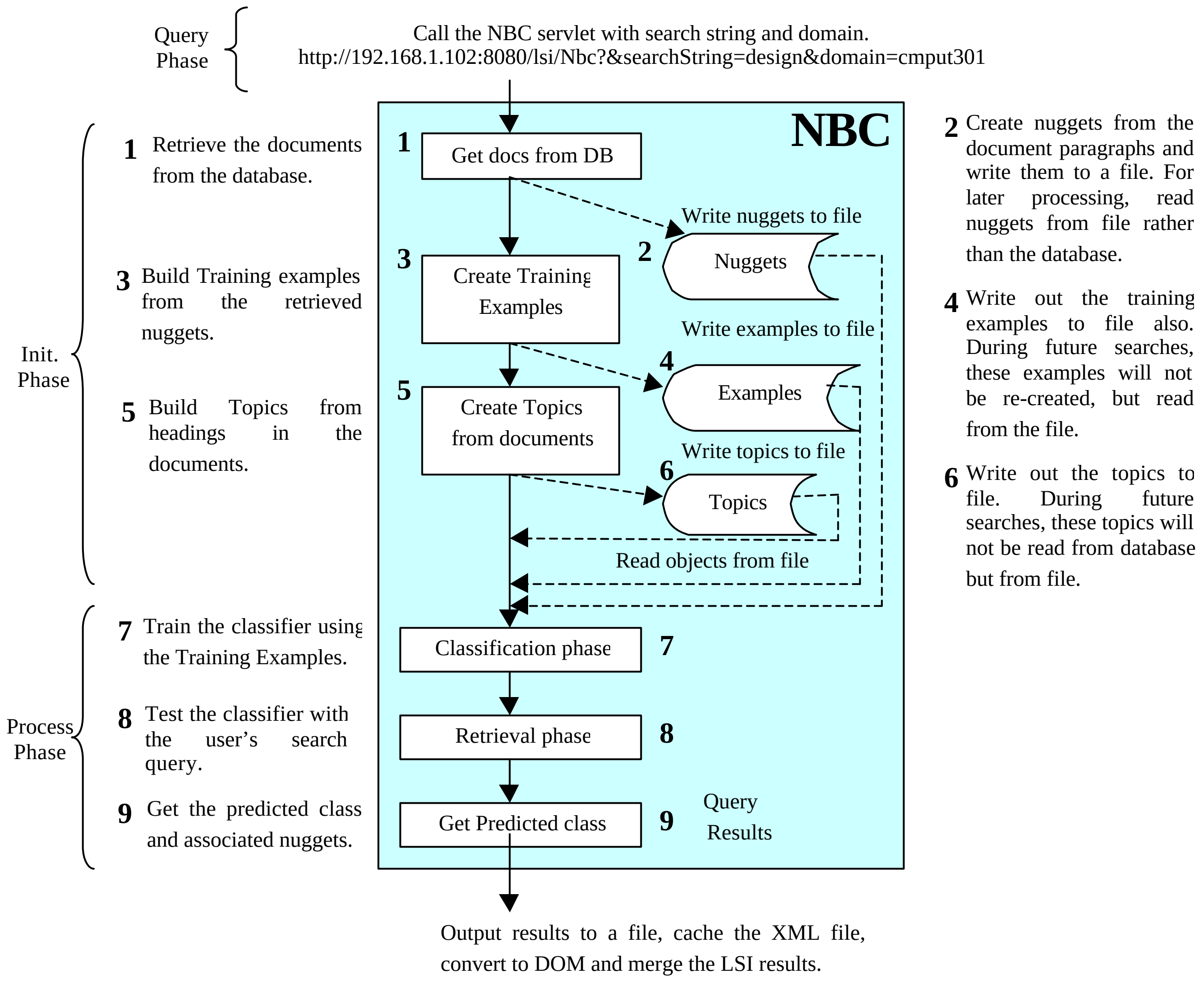

Query Phase
Call the NBC servlet with search string and domain.
http://192.168.1.102:8080/lsi/Nbc?&searchString=design&domain=cmput301
Init. Phase
1 Retrieve the documents from the database.
3 Build Training examples from the retrieved nuggets.
5 Build Topics from headings in the documents.
Process Phase
7 Train the classifier using the Training Examples.
8 Test the classifier with the user's search query.
9 Get the predicted class and associated nuggets.
NBC
1 Get docs from DB
Write nuggets to file
2 Nuggets
3 Create Training Examples
Write examples to file
4 Examples
5 Create Topics from documents
Write topics to file
6 Topics
Read objects from file
Classification phase 7
Retrieval phase 8
Get Predicted class 9
Query Results
2 Create nuggets from the document paragraphs and write them to a file. For later processing, read nuggets from file rather than the database.
4 Write out the training examples to file also. During future searches, these examples will not be re-created, but read from the file.
6 Write out the topics to file. During future searches, these topics will not be read from database but from file.
Output results to a file, cache the XML file, convert to DOM and merge the LSI results.

Figure 11 – Annotated view of the NBC algorithm

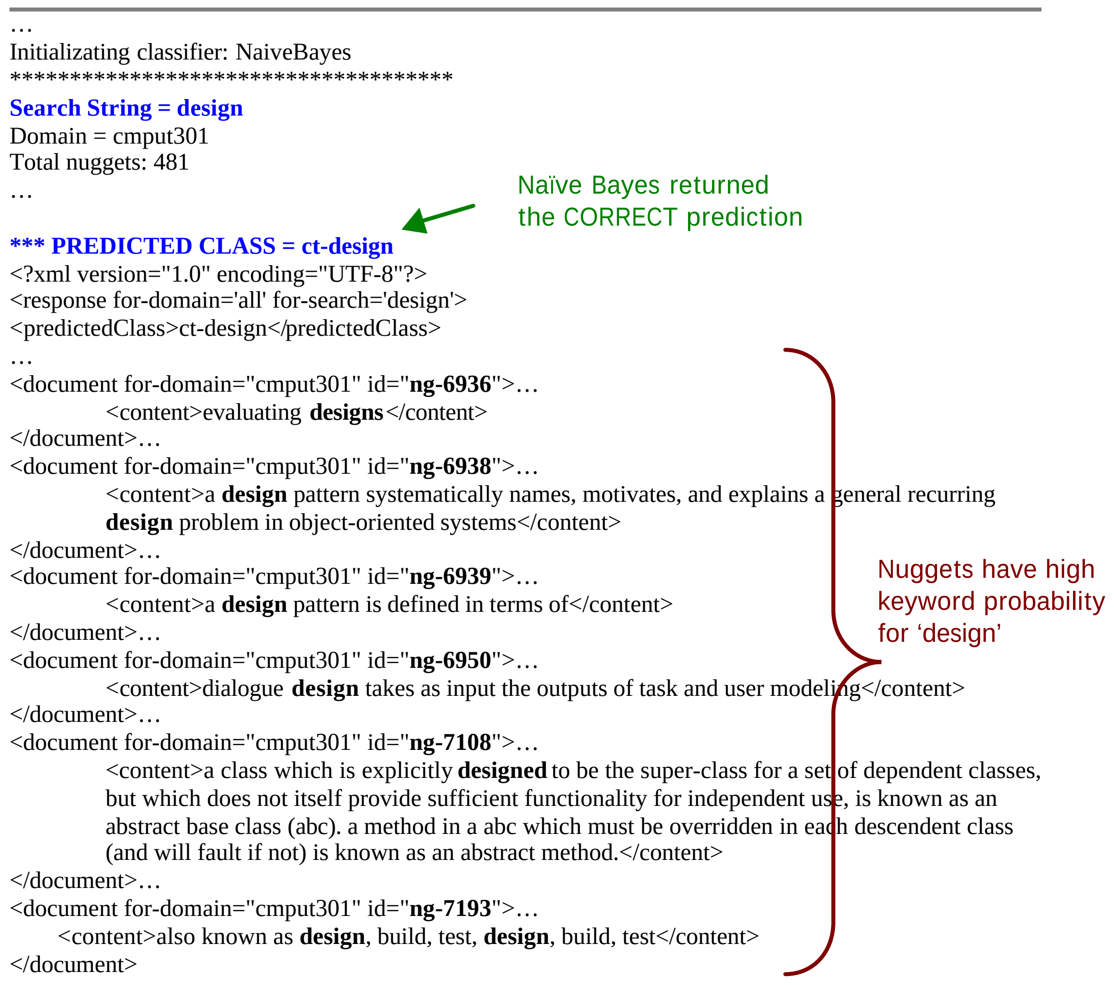
The training examples are then used in the classification phase. A training example is a vector containing the associated class labels for that example nugget, the nugget's name and nugget's term-frequency hash table:

Training Example t = <classLabelsVector, nuggetName, termFrequencyHash>

- classLabelsVector: Since each nugget can have multiple class labels, store all class labels with this example object for future use
- nuggetName: Name of the nugget, i.e. name of the document. E.g. Inheritance.html
- termFrequencyHash: The document is first parsed into individual terms. The terms along with the frequency of these terms are stored in a hash table.

The training example is classified under its most frequent class label. An instructor may also specify the class labels for each example but this is a relatively tedious task. The user's input query is built into a similar testing example. The Naive Bayesian classifier is run using the theorem described above and a class label is predicted for the new test example.

When a student inputs a query in the EduNuggets Student application, the Naive Bayesian classifier servlet is called in parallel with the LSI servlet, with the search query and domain as inputs. The classification phase begins after pre-processing tasks such as stemming, punctuation removal etc. have been performed on the documents and the user query acts as a test example for which a prediction has to be made. Once the classifier has made a prediction regarding the class in which the query most likely belongs, the nuggets of this class are returned.



```
…
Initializating classifier: NaiveBayes
*************************************
Search String = design
Domain = cmput301
Total nuggets: 481
…

*** PREDICTED CLASS = ct-design
<?xml version="1.0" encoding="UTF-8"?>
<response for-domain='all' for-search='design'>
<predictedClass>ct-design</predictedClass>
…
<document for-domain="cmput301" id="ng-6936">…
        <content>evaluating designs</content>
</document>…
<document for-domain="cmput301" id="ng-6938">…
        <content>a design pattern systematically names, motivates, and explains a general recurring
        design problem in object-oriented systems</content>
</document>…
<document for-domain="cmput301" id="ng-6939">…
        <content>a design pattern is defined in terms of</content>
</document>…
<document for-domain="cmput301" id="ng-6950">…
        <content>dialogue design takes as input the outputs of task and user modeling</content>
</document>…
<document for-domain="cmput301" id="ng-7108">…
        <content>a class which is explicitly designed to be the super-class for a set of dependent classes,
        but which does not itself provide sufficient functionality for independent use, is known as an
        abstract base class (abc). a method in a abc which must be overridden in each descendent class
        (and will fault if not) is known as an abstract method.</content>
</document>…
<document for-domain="cmput301" id="ng-7193">…
    <content>also known as design, build, test, design, build, test</content>
</document>
```

Naïve Bayes returned the CORRECT prediction
Nuggets have high keyword probability for 'design'

Figure 12 – Sample results from the NBC algorithm for query 'design'

This list is combined with the results of the LSI retrieval and all results are presented to the student in Panel A of Figure 8. Nuggets retrieved by both information-retrieval engines are given priority in this list. The panel D of the student application focuses on the top nugget of the list.

4.4.3 Search Process in the EduNuggets Student application

As has already been shown in Figure 9, the search process is initiated when a user selects a domain and enters a search query. At that point, the Search servlet is initialized, which in turn calls both the LSI and NBC servlets. Figure 12 shows the classifier's search results for the query 'design'. As can be seen from Figure 12, the classifier *correctly* predicted the class label for the test query (the database records 20 associations for the topic 'design')

The reason for the accurate prediction is that the classification phase contained a large number of examples under the topic 'design'. Furthermore, the test query was also 'design', so the classifier was able to predict with 100% accuracy. The evaluation section will show that there are a number of factors that can affect the accuracy of the classifier, namely – number of nuggets and topics, type of nuggets and topics etc. The complete list of search results returned by the LSI and NBC search engines and the final merged results are available in the appendix.

Figure 13, shows a sample of the classifier's search results for the query 'cognition'. In this case, the topic 'cognition' has only 7 associations recorded in the database (as opposed to 20 for the topic 'design'). The number of associations definitely affected the prediction of the classifier and this is visible from Figure 13.

From Figure 13, we can observe that two of the search results happen to contain forms of the word 'cognition'. These are by no means the entire set of results returned on querying the EduNuggets Student application with 'cognition', however, this portion of the search results were selected to illustrate a point. First, it is important to remember that the search results are a combination of results from the LSI and NBC algorithms. Second, each training example in the classifier is associated with multiple topics (class labels). In our scenario, the LSI algorithm did not return any results for 'cognition'. Therefore, the search results shown above are from NBC alone. The interesting thing to note is that NBC incorrectly predicted the class label for the query 'cognition' to be 'user'. This is because the Naïve Bayesian classifier performs predictions based on a probability model. As such, the probability of all the nuggets belonging to a particular class (having an association with a particular topic) is known. The classifier predicts class membership for a new example based on the attributes of known examples for a given class. In our case, very few nuggets were associated with the topic ‘cognition’ in the database. As a result, the topic ‘cognition’ had a low probability in the probability model. However, forms of the word ‘cognition’ were found in nuggets associated with the topic ‘user’. Also, the topic ‘user’ had a large number of nuggets associated with it resulting in a higher probability for this class and hence the incorrect prediction.

```
User: instructor
 Opening Connection, EduNuggetConnection: PostgreSQL version

Initializating classifier: NaiveBayes
************************************
Search String = cognition
Domain = cmput301
Total nuggets: 481
…
*** PREDICTED CLASS = ct-user
<?xml version="1.0" encoding="UTF-8"?>
<response for-domain='all' for-search='cognition'>
<predictedClass>ct-user</predictedClass>
…
<result id ="10">
<document for-domain="cmput301" id="ng-7205">
        <title>7205</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html</url>
        <content>understanding the user involves understanding  how the human works  which is the subject
matter of cognitive  psychology</content>
</document>
</result>
…
<result id ="15">
<document for-domain="cmput301" id="ng-7267">
        <title>7267</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/UserModels.html</url>
        <content>cognitive  models model aspects of user:</content>
</document>
</result>
<result id ="23">
<document for-domain="cmput301" id="ng-7381">
        <title>7381</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/pj1.html</url>
        <content>the amount of data available in the urls above is  huge ; you have to find a way to limit the
scope of your application data without limiting its functionality. …</content>
</document>
</result>
</response>
```

Naïve Bayes returned the WRONG prediction

These nuggets were returned in the search results because they contain the word ‘cognition’

Figure 13 – Sample results from the NBC algorithm for query ‘cognition’

4.4.4 User Feedback

After the information-retrieval engines have returned the query results, the student can proceed to review them by selecting them from either panel A, B or D of the EduNuggets Student application. The selected nugget is then rendered in panel C, which is a tailored SMIL browser. Now, the student can evaluate each retrieved nugget and provide negative feedback if he/she assesses it as irrelevant to the query. Feedback has not been incorporated in this project and will be considered as a future work activity.

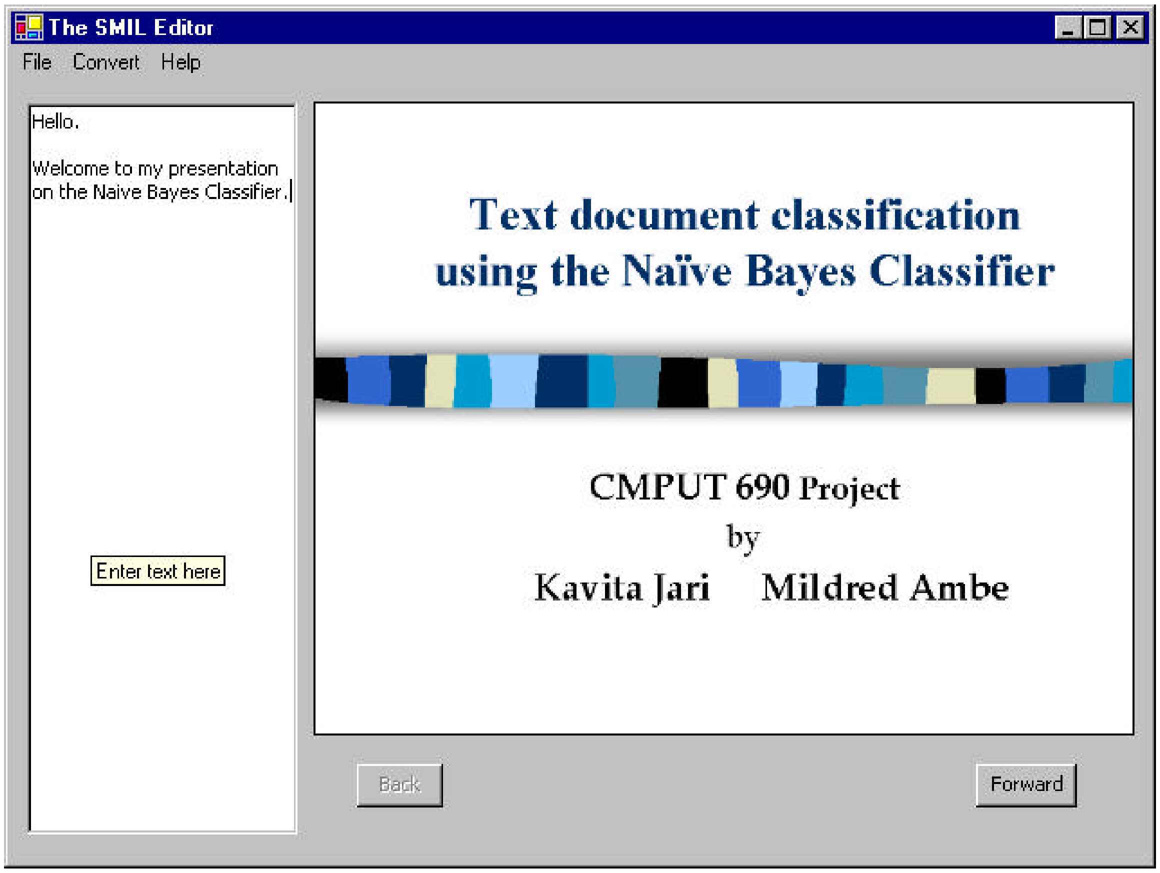
In order to include feedback into the EduNuggets Student application, it may be done in a manner similar to developing the NBC servlet. A feedback servlet may be created that will take the search string, domain and feedback type as input. This servlet will cause the classifier model to be rebuilt only if negative feedback is received via the student application feedback buttons. Generally, all training examples are classified under their best class label (based on keyword frequencies). Upon receiving negative feedback, this policy will be updated. The example that received the negative feedback will be built again with its next best class label. The old example will be discarded. At this point, there is no need to do anything else. The next time the student searches for the same query, the classification phase will run with the updated training example set. The retrieval/response procedure will continue as before. This process will ensure that the classifier no longer associates the example with the offending class label and thus this class label will not be chosen as a possible prediction.

The adaptation procedure is carried out differently in the NBC and LSI search engines. Presently, LSI does not have a policy for feedback. However, feedback can be incorporated in terms of “do not show” lists. On negative feedback, the LSI engine can record the negative association of the retrieved nugget with the input query and if the same query is issued later, this nugget will not be returned.

5. Multimedia Development in EduNuggets

The SMIL Editor is a Visual Studio .NET application (written in Visual C#) that allows instructors to create interesting and interactive SMIL presentations that can be accessible on multiple platforms, in particular Windows and Unix. SMIL or Synchronized Multimedia Integration Language is extensively used for integrating different types of resources such as discrete or continuous media into attractive multimedia presentations. Generally, the components of a SMIL presentation may include audio, video, HTML, text, animation etc.; however, in the EduNuggets framework, The SMIL Editor is used by the instructor for developing multimedia presentations containing text and PowerPoint slide image components.

Due to the proprietary nature of PowerPoint, presentations developed using the Microsoft Office PowerPoint utility can only be displayed on Windows. Often, students may not own personal computers and are forced to use machines in a University setting. Even though some machines may be equipped with Windows, more often that not, University machines have a Unix-like operating system (Unix or Linux). As a result, the wealth of information found in PowerPoint slides developed by the instructors is wasted. The SMIL Editor ensures that these slides can be reused by converting the slides into GIF images, including descriptive text for each slide and developing interesting presentations that have the look and feel of the original PowerPoint presentations (the editor offers a default presentation template at this time but future editions of the editor will include wizards). Figure 14 provides a snapshot of The SMIL Editor. It may be used as a stand-alone tool or may be incorporated into the EduNuggets instructor application. The right frame shows a GIF image (from a PowerPoint presentation). The left frame shows a textual explanation of the PowerPoint slide as input by the instructor.



Hello.

Welcome to my presentation on the Naive Bayes Classifier.

Text document classification using the Naïve Bayes Classifier

CMPUT 690 Project
by
Kavita Jari Mildred Ambe

Figure 14 - The SMIL Editor application displaying a SMIL file

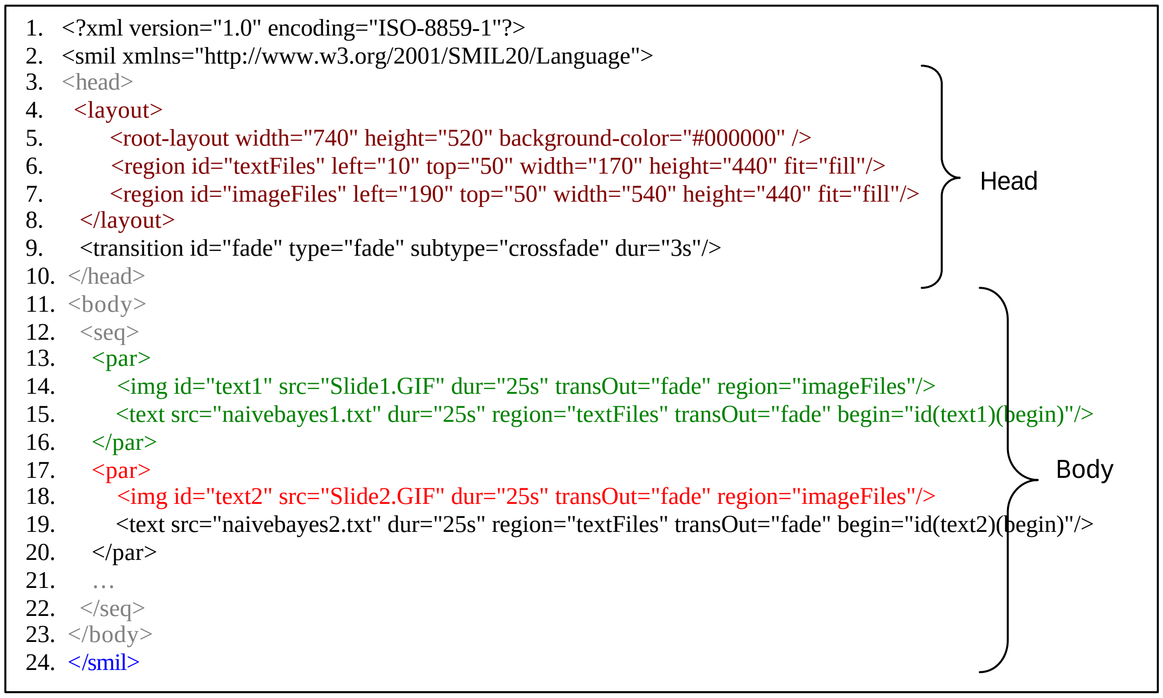
The SMIL Editor supports the following processes:

1. Converting PowerPoint presentations into a set of GIF images where each PowerPoint slide is stored as a separate image [22]. These images are stored at a location specified by the instructor and in a directory matching the presentation name. For example, a presentation named “Multimedia_Technologies.ppt” is stored in a directory called “Multimedia_Technologies”. Multiple presentations can also be converted one after another in the same session.
2. Creating text files explaining the contents of each PowerPoint slide. If the instructor provides some text for a slide, the text file is saved and stored in the directory containing the GIF images. If there is no associated text for a slide, no text file is created for that slide.
3. Integrating the PowerPoint GIF images and informative text files that explain the contents of the slide into a SMIL file with .smil extension. This SMIL file can also be

stored at a location specified by the instructor (preferably in the same directory that contains the GIF images). The instructor can specify the name of the SMIL file as well.

4. Developing the SMIL file with basic timing, synchronization and transitions (part of the SMIL 2.0 Language Profile modules). These features will allow the final SMIL presentation to be similar in style to the PowerPoint presentation (due to timing and default "fade out" transitions), and interesting and informative (due to unique synchronization of text and images).

Figure 15 gives a snapshot of a SMIL file. The SMIL syntax is similar to XML and this is visible from the structure of the file.



```
1.   <?xml version="1.0" encoding="ISO-8859-1"?>
2.   <smil xmlns="http://www.w3.org/2001/SMIL20/Language">
3.   <head>
4.     <layout>
5.          <root-layout width="740" height="520" background-color="#000000" />
6.          <region id="textFiles" left="10" top="50" width="170" height="440" fit="fill"/>
7.          <region id="imageFiles" left="190" top="50" width="540" height="440" fit="fill"/>
8.     </layout>
9.     <transition id="fade" type="fade" subtype="crossfade" dur="3s"/>
10.  </head>
11.  <body>
12.    <seq>
13.      <par>
14.         <img id="text1" src="Slide1.GIF" dur="25s" transOut="fade" region="imageFiles"/>
15.         <text src="naivebayes1.txt" dur="25s" region="textFiles" transOut="fade" begin="id(text1)(begin)"/>
16.      </par>
17.      <par>
18.         <img id="text2" src="Slide2.GIF" dur="25s" transOut="fade" region="imageFiles"/>
19.         <text src="naivebayes2.txt" dur="25s" region="textFiles" transOut="fade" begin="id(text2)(begin)"/>
20.      </par>
21.      …
22.    </seq>
23.  </body>
24.  </smil>
```

Figure 15 – Basic structure of the naivebayes.smil file

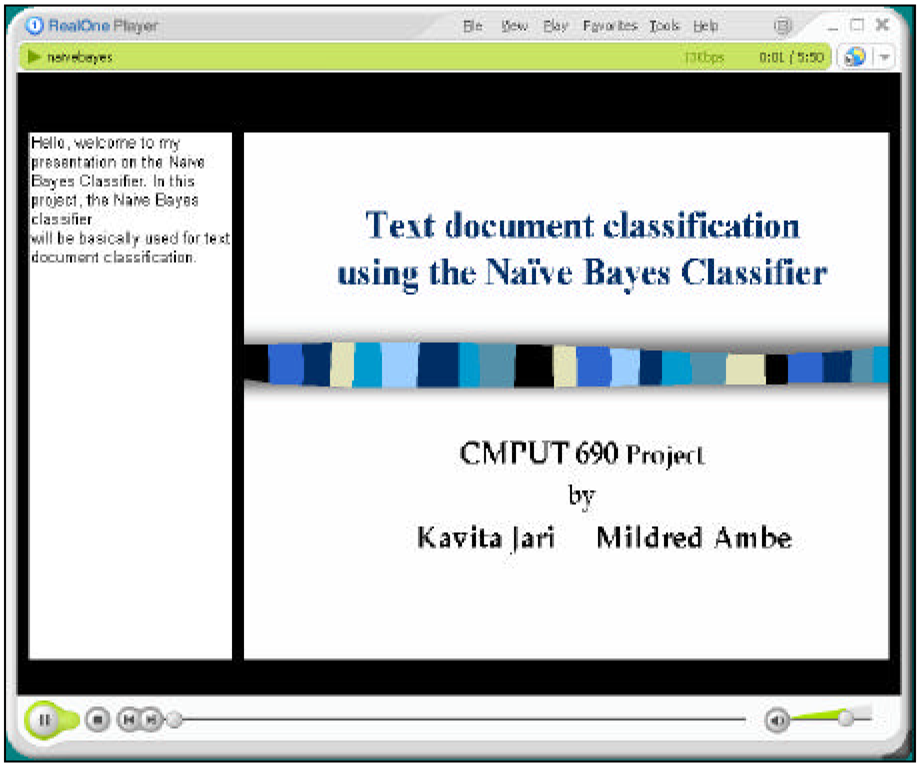
From Figure 15 we can see that a basic SMIL file has several components:

1. Statement 1 is a description of the XML version being used and the encoding standard.
2. Statement 2 declares a default namespace for the elements in the SMIL file using the W3C namespace [24] for the xmlns attribute.
3. Statements 3 – 10 include the head portion of the SMIL file. The layout for the

presentation is set up in statements 4 – 8. The <root-layout> element determines the layout of the window in which the SMIL presentation is rendered. The <region> element controls the position; size and scaling of media object elements [24].

4. Statements 11 – 23 describe the body of the SMIL file. The <seq> container allows media elements within it to be displayed one after another. The <par> container on the other hand allows multiple elements (media components) to play at the same time. The <img> and <text> media object elements define the image files and text files that are referenced by this presentation.

Figure 16 shows a sample SMIL file playing in the RealONE player by RealNetworks.



Hello, welcome to my presentation on the Naive Bayes Classifier. In this project, the Naive Bayes classifier
will be basically used for text document classification.

Text document classification
using the Naïve Bayes Classifier

CMPUT 690 Project
by
Kavita Jari   Mildred Ambe

Figure 16 - A SMIL file called "naivebayes.smil" playing in the RealONE player

As mentioned before, The SMIL Editor may also be integrated with the EduNuggets instructor application. Instructors can then develop their own SMIL presentations alongside developing the topic map for a particular domain. The EduNuggets student application already provides support for a SMIL browser in Panel C of Figure 8. SMIL files including audio and video content can currently be rendered in this browser.

As part of future work, the individual frames in the SMIL presentation (i.e., a text file and GIF image combination) can be developed as nuggets and stored in the database. For instance a particular slide may talk about “User Interface Design”. Using The SMIL Editor, the instructor may annotate this slide with their own information, which will be stored in the corresponding text file. Furthermore, the text notes in the original PowerPoint slide may also be extracted and developed into a nugget. The SMIL slide may then be referenced as part of the nugget content. Since the PowerPoint slide will not be visible on a Unix machine, the SMIL counterpart will be used. If a student then searches for information related to “User Interface Design” or “Interfaces” or “Design”, the SMIL slide along with slide content may be returned in the set of results. The remainder of the results might include segments of video clips, audio clips or HTML documents similar to the current implementation of EduNuggets.

6. Evaluation

The main feature of the EduNuggets student application is the automatic topic map generation. Instructors can define core concepts related to their resource material in the topic map and students can browse this map visually to gain knowledge about the related information. As a result, several case studies were performed to judge the “goodness” of the default topic map. The quality of nuggets and topics as well as the size of these components in turn influences the effectiveness of the classifier.

‘Accuracy’ was chosen as the primary measure for testing the validity of the classifier. Given a set of test examples, accuracy can be measured in terms of the number of correct predictions over the size of the sample set. The following formula can also be used:
Accuracy of the classifier = (Number of correct predictions / Test set size)

The results for the classifier under the different case studies are discussed in sub-section 1. The SMIL Editor was also tested under the Windows and Unix platforms. The experiments on the editor with results are presented in sub-section 2.

6.1 Case studies and results for the Naïve Bayesian Classifier

As has been described already, the Naïve Bayesian classifier takes as inputs some pre-defined training examples. When a new test example is provided to the classifier, it predicts class membership for this example based on the attributes of known classes namely:

1. Keyword probabilities of terms per documents
2. Number of documents classified under a particular class label.

The classification process aims at finding the class label that will represent the test example best based on the maximum probability of the example occurring under that class. In the EduNuggets application, the train and test sets are comprised of nuggets, class labels are represented by topics and associations specify the weight of a class, i.e., number of nuggets belonging to that topic.

Keeping the above points in mind, several case studies were run to evaluate the validity of the classifier. Three different datasets were used in testing the response of the classifier - a set of notes for Cmput 301 at the University of Alberta (39 documents), tutorial notes on learning the Java language from the Java Sun site (100 documents) and a set of soccer articles describing results from English Premiership League matches (30 documents). From Table 1, we can see that the classifier fared best with the soccer dataset. Even though the accuracy of the classifier is very low (9.3% with XPath 1 and 12.7% with XPath 2), we must observe that this accuracy was obtained using a very small dataset of 30 documents. We can reasonably predict that with a much larger dataset similar to the current soccer dataset, the accuracy of the classifier will indeed increase.

The accuracy of the classifier from the other two datasets is much worse and this is due to the nature of the documents in the dataset. In EduNuggets, nuggets are generated from HTML documents. As such, the classifier is deeply impacted by the nugget generation process - the type of nuggets, the size of nuggets and the number of nuggets. Furthermore, the number and type of topics also affects the accuracy of the classifier.

Dataset	# Topics	# Nuggets	# Assoc.	Training examples	Size of Train set	Size of Test set	Accuracy (%)
XPath = /html/body/p							
301 notes	90	482	436	339	181	20	**5**
Java notes	76	204	114	200	90	9	**2.22**
Soccer articles	26	159	359	159	151	15	**9.3**
	Min # correct	**Max # correct**					
301 notes	0	5					
Java notes	0	1					
Soccer articles	0	9					
XPath = /html/body/*[text()]							
301 notes	90	841	1220	800	483	53	**2.45**
Java notes	76	532	490	531	265	29	**6.2**
Soccer articles	26	188	463	188	162	18	**12.7**
	Min # correct	**Max # correct**					
301 notes	0	5					
Java notes	0	6					
Soccer articles	0	13					

Table 1 - Naive Bayes Classifier results for 3 data sets and 2 XPath expressions

Table 1 shows results for the Naïve Bayes Classifier under the 3 data sets. The table shows the # topics generated for each data set (# topics), the number of nuggets generated from the documents (# nuggets), the associations created between topics and nuggets (# assoc.). Furthermore, the table also shows the actual number of training examples provided to the classifier. The test set is taken to be 1/10th of the total training examples and 10-fold cross validation is applied during the case studies to determine the accuracy of the classifier. The number of training examples is different from the total number of nuggets because only nuggets that are associated with topics are chosen for the classifier. This seems reasonable as nuggets without associations (isolated nuggets) are not classified under any topic. As such, they are of no use in the prediction process. The instructor may manually add associations to these nuggets but this process is relatively tedious. The observations from the studies are analyzed and described below.

6.1.1 Nugget generation

(a) Type of nuggets

The type of nuggets greatly affects the accuracy of the classifier. This is because one of the key points that the classifier bases its predictions on is keyword probabilities or the weight of terms in a nugget. If the nuggets are sparse, i.e., have limited content, or if they are not meaningful, the document-term matrix generated for each nugget will have low probabilities. This will in turn result in the probability of a correct prediction being very low. This is visible from Table 1 under the results for the 301 notes' data set and the java files data set.

The nugget generation process in EduNuggets uses the XPath language. "XPath is a language for addressing parts of an XML document, designed to be used by both XSLT and XPointer" [5]. XPath expressions are applied to the downloaded HTML documents and nuggets are generated from text nodes that match these expressions. During the setup of the case studies, several XPath expressions were tried in order to determine which produced the best nuggets:

1. /html/body/p/

This XPath expression generates nuggets from all elements under <p> tags. In effect, this XPath provides the text from all paragraphs in an HTML document. Tests were performed on all three datasets using this expression.

2. /html/body/*[text()]

This XPath expression generates nuggets from all text nodes under the <body> tag. Since HTML documents contain only one body tag, all text nodes under the body tag were created into nuggets. Tests were done on all three datasets using this expression.

3. //p/descendant-or-self::*[text()]

This XPath expression generates nuggets from all text nodes under and including the <p> tag. The //p specifies that all elements in the document that match the criteria are selected. The descendant-or-self axis selects the context node and all its descendants.

4. //body/descendant-or-self::*[text()]

This XPath expression is similar to the one in 3 and generates nuggets from all text nodes under and including the <body> tag.

5. //hr/*

This XPath expression does not yield any nuggets since the <hr> tag does not have a corresponding closing </hr> tag. Thus, XPath cannot be used to extract text nodes from this expression.

After the different XPath expressions were used to generate nuggets for the three datasets, the following things were observed:

1. XPath expression 1 was the most favorable as it resulted in nuggets out of every <p> tag. However, the nuggets generated via this XPath were strongly affected by the style of the HTML document. For instance, certain writers encapsulated large chunks of text in <p> tags resulting in large meaningful nuggets. On the other hand, some writers tended to write abstract points in <p> tags or no text at all resulting in sparse useless nuggets. The lecture notes from 301 produced sparse nuggets due to the abstract nature of the text (5% accuracy). Nuggets from the java files dataset produced larger but still abstract nuggets because the files contained large pieces of code segments (2.22% accuracy). The soccer dataset fared best with this method of nugget generation as nuggets were moderately sized paragraphs and had lots of relevant content (9.3% accuracy). Refer to Table 1 for results with this XPath expression.
2. XPath expression 2 resulted in a larger number of nuggets with very fine content. In a few cases, a number of the nuggets comprised one line statements that were out of context (text nodes that were not inside <p> tags but were part of the <body> tag) while others were very short paragraphs. As can be expected, the classifier predicted poorly with datasets using this form of nugget generation. These results are again evident from Table 1. The classifier's accuracy on the 301 data set decreased to 2.45

% because of the structure of the HTML documents. A large number of nuggets in this scenario were one-line statements (due to abstract style of writing). Despite the large number of associations, the classifier predicted poorly because keyword probabilities were very low. On the other hand, the classifier's accuracy increased with the Java data set and this XPath expression. A visual check of the nuggets showed that they were far more meaningful that those created with the first XPath expression. The style of the Java files was such that a number of meaningful statements/paragraphs were included in the HTML document outside <p> tags. This style of nugget resulted in the classifier accuracy increasing to 6.2%. Finally, the soccer data set once again fared best and the accuracy of the classifier with this set jumped to 12.7%. The nuggets with this XPath expression included meaningful paragraphs outside <p> tags.

3. XPath expressions 3 and 4 resulted in an enormous number of nuggets that were completely unusable. These expressions constructed nuggets out of every text node, i.e., every statement inside a paragraph in the case of expression 3 and every statement in the <body> tag in the case of expression 4. A visual inspection of the nuggets showed that the content did not relate to anything meaningful and hence very few associations could be developed. These datasets were discarded.
4. XPath expression 5 could not be used as already mentioned above. However, it is important to point out that this test is highly subjective in nature since it depends on the style of the HTML document. All the documents in the 301 dataset contained this tag as the writer tended to separate meaningful content using <hr> tags. The other datasets were not structured in this way.

The major conclusion that can be drawn from the XPath nugget generation scheme is that the underlying structure of the HTML documents greatly affects the types of nuggets that are generated and in turn the accuracy of the classifier. Even though the XPath language is meant to "operate on the abstract, logical structure of the XML document rather than the surface structure" [5], its use in the EduNuggets application limits it to a "pattern-matching algorithm". The different XPath expressions merely look for elements that match a specific pattern and construct nuggets based on this pattern. While this is a good approach, the power of XPath can be greater utilized by using it in conjunction with the underlying DOM (Document Object Model) structure of the HTML documents [idea developed by Curtis Schofield].

A future version of EduNuggets will investigate the idea of developing nuggets out of large chunks of data in an intelligent manner rather than merely automating the process. The idea here is to analyze the tree structure of the HTML document as represented by the DOM [6] and search for all the text nodes present in the document, possibly after a <p> tag. The next step will be to search for text nodes at a distance x from a context node (i.e., path to all other nodes from the context node is no greater than x). The basic assumption here is that text nodes close to one another may be related. Using the DOM, try to find a node that encompasses all the related text nodes and provide a general XPath expression to that node (such as XPath_to_context_node/descendant-or-self::*[text()]). Now, the nuggets constructed using this XPath expression will contain relevant, meaningful content that is related.

In terms of the "goodness" of the final topic map, we can see that meaningful nuggets will be useful to students when they perform a search. Entire documents may be chosen as nuggets however this will take away from the "focused" approach being developed in EduNuggets. Even though the automatic nugget generation process is incomplete at present and requires further work in order to use it effectively, this process indeed helps the instructor and reduces their burden of creating relevant nuggets by automating the annotation process. It is possible to see that in a scenario where the instructor may have a lot of documents they wish to annotate, either their own or external, the automatic nugget generation process will save the instructor time and effort. With further investigation in the possible technology(ies) that can be used for nugget generation, EduNuggets will provide students with concise, informative and relevant material.

(b) Size of nuggets

The size of nuggets does not affect the accuracy of the classifier directly. However, the type of nugget (i.e., entire documents or paragraphs) may result in larger or smaller nuggets. If entire documents are chosen as nuggets, there will naturally be far more large nuggets in comparison to nuggets generated from every paragraph in a document. In both cases, the type of nugget will affect classifier results as mentioned in the previous section.

In the context of EduNuggets, large nuggets maybe very general, i.e., they may contain references to several concepts. This can detract from the goal of EduNuggets where the aim is to develop nuggets pertaining to specific concepts so that when students issue a query they get document segments that are relevant to the query (within a document, the relevant segment is highlighted). Furthermore, large nuggets may slow down processing as nuggets are being retrieved or accessed from the database. As a result, it is better to have a large number of moderately-sized nuggets with relevant content.

(c) Number of nuggets

The number of nuggets that are generated in the bootstrapping process also affects the accuracy of the classifier. This has already been hinted at in the previous sections. In particular, a large number of meaningful nuggets will aid in the classifier making accurate predictions. This is because of the two key factors that the classifier bases its predictions upon, namely, weight of keywords per document and number of documents associated with a specific class label. In the case of a large number of useful nuggets, the keyword probabilities for each nugget will be high. Furthermore, it is quite likely that a large number of nuggets will develop associations with specific topics. When a new nugget will need to be classified, the Naïve Bayesian classifier will reflect on its model of the pre-classified nuggets and produce good predictions. This can be observed from the test results produced from the Soccer dataset. Despite having a small number of nuggets, the classifier was able to predict well in comparison with the other two datasets.

During the case study, a large number of nuggets were generated with empty content in the 301 lecture notes and java files datasets. This is mainly due to the style in which these documents were written where some <p>…</p> tags were included with no text (only white space). These empty nuggets were removed from the database as they did not contribute to the classification process in anyway. Furthermore, there were also some nuggets that were not associated with any topics. As described earlier in the report, associations are generated between a nugget and a set of topics if the topic keywords exist in the nugget content. Nuggets that had no associations did not contain any core topic terms. Even after adding manual associations the accuracy of the classifier did not improve, as is expected. The ideal scenario is to increase the number of documents in the dataset. For example, this can be done with the Soccer dataset. Increasing the documents will not only increase the number of nuggets generated per document but also improve the overall accuracy of the classifier.

6.1.2 Topic generation

(a) Number of topics

If a large number of topics are used, the accuracy of the classifier will decline because there will be too many topics that can become potential candidates for a new nugget. On the contrary, a fewer number of "good" topics may potentially increase the accuracy of the classifier ("good" here refers to meaningful topics with large probability of occurring in the document). These results can be observed from Table 2. Ideally, we need a large number of useful nuggets associated with few core topics.

Dataset	# Topics	# Nuggets	# Assoc.	Training examples	Size of Train set	Size of Test set	Accuracy (%)
XPath = /html/body/p							
301 notes	30	482	196	127	115	12	**8.33**
Java notes	30	204	36	31	28	3	**43.3**
Soccer articles	10	159	218	136	123	13	**7.69**

Table 2 - Naive Bayes Classifier results with fewer topics and 1 XPath expression

Table 2 shows the results of the Naïve Bayes classifier with 3 data sets. The number of topics generated was reduced in each case (301 set from 90 to 30, Java set from 76 to 30 and Soccer set from 26 to 10). With a fewer number of good topics, the accuracy of the classifier improved in the case of the 301 and Java data set. However, it can be observed from Table 2 that in the case of the Java data set, decreasing the number of topics drastically reduced the number of associations that were generated. As a result, the number of training examples that were provided to the classifier was also reduced resulting in a very small test set (size 3). Given these inputs, the classifier predicted with 43.3% accuracy. These results can be explained in the following way – the 30 topics that were generated were the best out of all potential topics in terms of their frequency in the documents. Keeping the number of nuggets constant, fewer associations were generated because very few nuggets could be linked with these topics. Nuggets with associations were then provided to the classifier (31) and 10-fold cross validation was done to determine the accuracy of the classifier. Since the training examples were richly associated with the topics, the classifier was able to predict well in each fold of retrieval. In the case of the soccer data set, the accuracy of the classifier dropped because the number of associations also reduced with decrease in number of topics. These results show that with a good set of limited topics that are *richly* associated with meaningful nuggets, the Naïve Bayes classifier is a useful classification tool.

(b) Type of topics

Two different strategies were investigated for the type of topics. In the first strategy, Rainbow – a statistical tool for generating terms, was used to develop topics from the downloaded HTML documents. In this strategy, the topics that were produced were not very meaningful. For e.g., topics produced with the 301 dataset included words such as red, green etc. due to their repeated occurrence in the documents. As a result, the classifier accuracy was affected due to the poor choice of topics and dropped to 2% as shown in Table 3. However, in the case of the Soccer data set, Rainbow chose frequently occurring words such as Arsenal, Manchester, championship etc. that occurred in almost every document as topic words. The accuracy of the classifier increased with the same XPath expression (/html/body/p) in comparison with results in Table 1 (Note also that the number of topics is greater in this case). Table 3 summarizes these results. Results from the Java data set have not been included because Rainbow chose words such as public (as in public keyword in Java), int (as in the data type integer in Java), main (as in the main method in Java) as topic words. These topics were not meaningful and the accuracy of the classifier was much below 2%.

Dataset	**# Topics**	**# Nuggets**	**# Assoc.**	**Training examples**	**Size of Train set**	**Size of Test set**	**Accuracy (%)**
XPath = /html/body/p							
301 notes	50	482	73	55	50	5	**2**
Soccer articles	50	159	598	130	117	13	**12.3**

Table 3 - Naive Bayes Classifier results with Rainbow topics and 1 XPath expression

In the second strategy, topics were created from document headings. These results have already been presented in Table 1. This strategy seems to be an intuitive approach as the underlying assumption is that the writer of a document will provide the main idea behind the material in the heading of the document. As such, text within tags such as <b>, <strong>, <h1>...<h6> was used in order to extract topic keywords. However, as can be observed from the results in Table 1 and Table 3, this approach is once again strongly influenced by the style of the document writer. For instance, it is possible that an instructor may mistakenly include the entire document text within <b>…</b> tags. In this scenario, the topic generator program will pick words from this large amount of text, which is not desirable. This flaw is easily fixed but is to be considered nevertheless in cases where documents come from external sources. Secondly, it may be possible that the heading provided by the writer of the document may be meaningful but may use words that do not occur in the text of the document. This will definitely affect the predictions made by the classifier. In conclusion, the approach to topic generation also requires deeper investigation. Depending on the style of the HTML document, different techniques may need to be applied in order to develop meaningful nuggets and topics.

It is important to note that even after the automatic bootstrapping process, the instructor has the option to modify the automatically generated nuggets and topics via the EduNuggets Instructor application. The instructor can take this avenue in case they feel the nuggets and topics are not useful. Keeping in mind that the manual annotation task is extremely tedious and may require copyright permission when using external sources, the benefits of the automatic generation process are clearly visible. The automatic bootstrapping process helps instructors by doing their initial task for them, i.e., the process of extracting relevant topics and nuggets from a large mass of documents. This provides instructors with some material to work with, upon which they may choose to define their own requirements. Nonetheless, the amount of work to be done by the instructor is greatly reduced.

6.1.3 General Observations

The results of the case studies have shown that the Naïve Bayesian classifier fared poorly due to the nature of the dataset, in particular the types of nuggets that were generated in the bootstrapping process. The types of nuggets in turn were affected by the style of nugget extraction that was applied, for instance XPath versus DOM style extraction. However, it can be ascertained that the root of the problem lies in the structuring of the HTML documents from which these nuggets are generated. Documents that are available on the World Wide Web despite following the standard HTML syntax are still developed based on the style of the writer. Some writers prefer to clarify concepts by sectioning them off into large paragraphs, while others prefer abstract points or lists to illustrate ideas. There are a large number of writers that use the HTML tags incorrectly or sparingly. As a result, a single policy for nugget or topic extraction cannot work. In cases where the documents originate from the same source, it is possible to develop a reasonable policy for extraction; however, this is a near-sighted solution and cannot be applied to external sources. Nonetheless, the bootstrapping process has provided a way for instructors to extract useful information from a wide variety of material and provide it to students in a coherent, informative way.

6.2 Usability Evaluation and Aspect-Oriented Programming

A major goal of the EduNuggets framework is to provide student's with a convenient method for browsing search results. As has been explained in Section 4.3.2 and is visible from Figure 8, students can browse through search results in a graph visualization panel in the EduNuggets student application (Panel D). They may also explore search results in a tree-view in Panel B. Students can then click on hyper-linked results in Panel A and view the rendered HTML file in Panel C. A future task is to perform an experimental evaluation of EduNuggets to record student's navigation behavior. Aspect-Oriented Programming (AOP) has been used in this regard to track and record user movements' [7].

AOP provides constructs, "aspects", to track and monitor code as it is being executed. Aspects are written separately from regular code (i.e., as separate Java files distinct from the EduNuggets code in this case) and essentially work "above" the base code (EduNuggets code) to track the flow of control in the program. They add and remove functionality to code without interfering with the code itself. In the case of EduNuggets, aspect files have been created that monitor important points of execution in the EduNuggets code e.g., dumping of the topic map, initialization of the search servlet, activities of the mouse on the EduNuggets student application etc. The experimental evaluation will analyze log files that contain information about which user is using the EduNuggets student application, search queries inputted in the application, search results returned by the search engines, mouse activity on the client etc. These results will then be compared against work analyzing navigational properties of user interfaces [10].

6.3 Experiments and results for The SMIL Editor

The SMIL Editor was developed using Visual Studio .NET. The resulting SMIL files were tested under Windows and Unix platforms. While the .smil files played fine on Windows, some problems were encountered on Unix. These results and general observations on the editor are summarized below.

In order to deploy this application to other computers, the SMIL Editor source code first needs to be bundled into a deployment project. A wizard can also be created that will install the editor on other Windows machines. This activity will be implemented as part of future work and information provided in the following link will be used in creating deployment projects and installation wizards for Visual Studio .NET [23].

6.2.1 Testing the third-party code that does PowerPoint slide to GIF

The SMIL Editor employs third-party code for the conversion of Microsoft PowerPoint slides to GIF images. This code imports .dll (dynamic link library) files that are specific to the version of Microsoft PowerPoint installed on the Windows machine. The first experiment on The SMIL Editor involved the execution of this 'PowerPoint slide to GIF conversion program' on computers supporting Microsoft Office 97/98. It was found that this converter program did not run at all. Only Windows machines supporting Microsoft Office (in particular Microsoft PowerPoint) 2000 or greater could be used. As such, this restriction forces The SMIL Editor to be installed on computers with Windows 2000, Windows NT, Windows ME or Windows XP and their associated Microsoft Office suites only.

6.2.2 Playing the SMIL file on Unix machines

Presently, the SMIL file does not play well on Unix machines because Unix cannot render the <text> tag properly; however, images are displayed well. On Windows machines, the SMIL file can be played using any video player e.g., Windows Media player, RealONE player, QuickTime etc. Future activity will involve extending the SMIL file components to include HTML files. It is envisioned the HTML will be rendered well on Unix due to the nature of the document structure.

6.2.3 GIF image rendering

The final SMIL presentation includes GIF images. Furthermore, these images are rendered inside a 540 x 440 region as shown in Figure 15. While these images are viewable, they are not very clear. The SMIL specification only provides four values for the fit attribute, which governs how the media component is laid out in the region. These are:

- fill - allows the media component to fill the region
- meet – allows the media component to meet its region size
- slice – allows the media component to grow without distortion and fill the region
- scroll – allows the user to scroll through the media component (assuming the media component's size is bigger than the region size).

Because of the limited possibilities in displaying the GIF images, the fit attribute was set to 'fill'. This results in slightly hazy pictures if the region size is small. This is a bug in SMIL and future releases of SMIL may fix this problem. A possible solution may be to increase the overall size of the presentation layout. However, this does not seem as a viable option because computer users generally have monitors that are 800 x 600 and a very large presentation may not look attractive on these small screens. In EduNuggets, the SMIL browser available in Panel C of the EduNuggets student application will be modified to become detachable so that students may increase or decrease the size of the browser thereby setting the size of the presentation layout as they see fit. Figure 17 shows an image being displayed using the 4 values for the fit attribute [25].

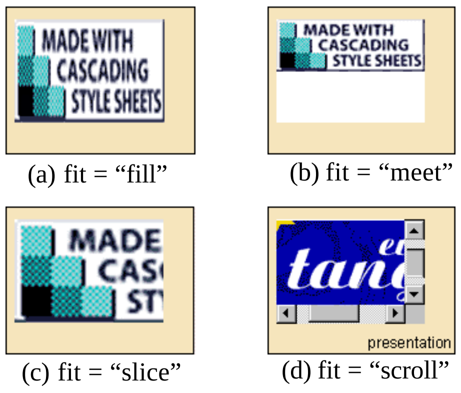


Figure 17 – Sample image layout using the fit attribute in SMIL

7. Future Work

EduNuggets provides an adaptive interface for using information available on the World Wide Web. Currently, the student application is a thick client; however, one of the primary future work activities is to re-architect the framework to make it a browser accessible thin client, thus enabling the use of the repository information from the web [11].

In EduNuggets, the instructor is the single authority responsible for the collection and annotation of study material via the domain-specific topic map. The students on the other hand have the crucial task of exploring through this collected information. Even though the automatic topic map generation feature within the framework attempts to reduce the workload on the instructor, the final topic map may still not be up to par. As such, further work needs to be done in the areas of generating meaningful nuggets and topics. A possible approach in generating topics may be to use the ACM dictionary of computing terms. Further strategies in developing nuggets (for example DOM-based XPath extraction) will also be explored.

The EduNuggets visualization of the topic-map is rich and informative; however, different types of associations are envisioned in future releases. As mentioned before, such associations may include specializations of topics and sub-topics. The search process will also be expanded to include search queries of type 'this and that', 'this or that' etc.

EduNuggets presently offers two alternative means of accessing the repository: a "direct access" mechanism through querying and information retrieval, and an "exploratory" mechanism through the topic-map visualization. One of the objectives of the upcoming experimental evaluation of EduNuggets is to explore the student's navigation behavior and to cross-reference the results with the substantial body of work analyzing the navigation properties of interfaces, especially for hypertext systems as described in [10].

One of EduNuggets significant features is its multimedia content and its fine-grained approach to accessing and displaying this type of content. Structured documents, such as HTML, are decomposed into smaller directly accessible segments through XPath expressions. Further work needs to be done in exploring effective XPath options for generating useful nuggets. Video and SMIL presentations may also be decomposed into short sequential segments and possibly annotated as in [9].

Another interesting property, enabled by the topic-map organization of the repository is the fact that it enables multiple views of the underlying documents [12]. A possible application of this capability is to support the instructor to construct documentation related to issues of interest that, although they are not covered explicitly by any single document in the repository, are addressed by multiple documents in the repository. In this way, instructors can reuse the effort they have invested in producing and collecting high-quality documents in the repository.

Finally, there has been quite a lot of effort invested recently in developing e-learning applications, such as the one described in [12] for learning to design user interfaces. As in that system, a major objective in EduNuggets is to enable learners to control their own learning process by providing multiple types of information –examples, definitions, discussions–, in different formats –audio/video, text– accessible in different ways.

A key feature to be incorporated into future releases of EduNuggets is student feedback. Students will have the choice of providing feedback (positive or negative) on documents returned as part of a search query. In the case of negative feedback, both the LSI and NBC search engines will be updated so that the tagged document is not returned in the next search. In the worst-case scenario, the user may decide to provide negative feedback on every retrieved document resulting in a sequence of classifier revisions. As a result, a "batch" update policy will also be implemented such that the search engines are not overworked in updating and modifying the existing models. Such policies should work well for both the NBC and LSI.

8. Conclusion

E-learning is becoming an increasingly important application area. Instructors are now posting their own notes online alongside using information that is already available for reference purposes. This information exists in a variety of formats such as HTML, audio, video, text, PowerPoint etc. Students can benefit from these resources but in most cases they have to go through the tedium of exploring multiple sites to gather information on a specific topic.

EduNuggets is an intelligent framework that enables the semantic organization of multimedia material so that students can access coherent, authoritative information on a subject area. By using the EduNuggets instructor application, instructors will have the ability to define interesting concepts and relations between these concepts in the subject domain. They will be able to annotate segments of interest in their own documents or documents extracted from external sources and provide these to students for furthering their knowledge. Instructors will also be able to generate interesting multimedia presentations that contain different media components working in collaboration, via the SMIL Editor. The vast store of information present in the repository will then be enriched with these interactive presentations, which will present students with a different and innovative approach to learning.

Using the EduNuggets student application, students will be able to pose queries; review documents pertaining to specific concepts retrieved in response to the query and navigate through the repository via a visual interface. Students will also benefit from efficient information retrieval engines that provide relevant information at the click of a mouse. It is envisioned that the EduNuggets framework will act as a portal that expands the horizons of distance-learning education integrated with a unique style of learning.

9. Acknowledgements

I would like to thank Dr. Eleni Stroulia and Curtis Schofield for their guidance in developing the bootstrapping process, the Naïve Bayesian Classifier and The SMIL Editor. Curtis has developed the EduNuggets framework and has provided the solution of DOM-based XPath extraction for nugget generation.

10. References

EduNuggets-related:

[1] EduNuggets Product Overview (EduNuggets Flyer): http://www.cs.ualberta.ca/~stroulia/EduNuggets/EduNuggets-Flyer.pdf

[2] Rainbow: http://www-2.cs.cmu.edu/~mccallum/bow/rainbow/

[3] Naive Bayesian Classifier source code: http://www.cs.utexas.edu/users/mooney/ir-course/

[4] Topic Maps: http://www.topicmaps.org/xtm/1.0

[5] XPath: http://www.w3.org/TR/xpath

[6] DOM: http://www.w3.org/DOM/

[7] Aspect-Oriented Programming: http://aspectj.org/servlets/AJSite

[8] Jeremy Goecks and Dan Cosley: *NuggetMine: Intelligent Groupware for Opportunistically Sharing Information Nuggets*, IUI 2002.

[9] Andrew S. Gordon, *Using annotated video as an information retrieval interface.* IUI 2000, 133-140

[10] Mark O. Riedl and Robert St. Amant, *Towards Automated Exploration of Interactive Systems*, IUI 2002.

[11] Jude W. Shavlik, Susan Calcari, Tina Eliassi-Rad, Jack Solock, *An Instructable, Adaptive Interface for Discovering and Monitoring Information on the World-Wide Web.* IUI 1999, 157-160.

[12] Robert L. Young, Elaine Kant, Larry A. Akers, *A knowledge-based electronic information and documentation system.* IUI 2000, 280-285

Multimedia Technologies / Multimedia and Web-based learning:

[13] Gregory D. Abowd, Christopher G. Atkeson, Ami Feinstein, Cindy Hmelo, Rob Kooper, Sue Long, Nitin Sawhney, Mikiya Tani, *Teaching and Learning as Multimedia Authoring: The Classroom 2000 Project*, ACM Multimedia '96, Boston, MA USA, 1996

[14] Sugata Mukhopadhyay, Brian Smith, *Passive Capture and Structuring of Lectures*, ACM Multimedia '99, Orlando, FL., USA, 10/99

[15] Franck Rousseau, J. Antonio Garcia-Macias, Jose Valdeni de Lima, Andrzej Duda, *User Adaptable Multimedia Presentations for WWW*, LSR-IMAG Laboratory, France

Information Retrieval:

[16] W. Bruce Croft, *What do people want from Information Retrieval?*, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst, Nov. 1995

[17] Bin Cheng, *Effective Information Retrieval using Latent Semantic Indexing*, University of Alberta, Sept. 2002

SMIL:

[18] Dick C. A. Bulterman, *SMIL 2.0: Overview, Concepts, and Structure*, IEEE Multimedia, October - December 2001

[19] Dick C. A. Bulterman, *SMIL 2.0: Examples and Comparisons*, IEEE Mutimedia

[20] Fabio Arciniegas A., *A Realist's SMIL Manifesto*, xml.com, May 29, 2002

[21] Fabio Arciniegas A., *A Realist's SMIL Manifesto Part II*, xml.com, July 17, 2002

The SMIL Editor:

[22] The Code Project Website:
http://www.codeproject.com/vb/net/litewait.asp

[23] Deploying a Windows Application in Visual C#:
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vsintro7/html/vbtskcreatinginstallerforyourapplication.asp

[24] W3C SMIL Specification:
http://www.w3.org/TR/smil20/

[25] Fit attribute in SMIL:
http://www.helio.org/products/smil/tutorial/chapter3/11.html

11. Appendix

(A) Search results for ‘design’ from the Naïve Bayesian Classifier algorithm

```
<?xml version="1.0" encoding="UTF-8"?>
<response for-domain='cmput301' for-search='design'>
<predictedClass>ct-design</predictedClass>
<result id ="0">
<document for-domain="cmput301" id="ng-6936">
        <title>6936</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignEvaluation.html</url>
        <content>evaluating designs</content>
</document>
</result>
<result id ="1">
<document for-domain="cmput301" id="ng-6938">
        <title>6938</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignPatterns.html</url>
        <content>a design pattern systematically names, motivates, and explains a general designa recurring design
        problem in object-oriented systems</content>
</document>
</result>
<result id ="2">
<document for-domain="cmput301" id="ng-6939">
        <title>6939</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignPatterns.html</url>
        <content>a design pattern is defined in terms of</content>
</document>
</result>
<result id ="3">
<document for-domain="cmput301" id="ng-6950">
        <title>6950</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html</url>
        <content>dialogue design takes as input the outputs of task and user modeling</content>
</document>
</result>
<result id ="4">
<document for-domain="cmput301" id="ng-7108">
        <title>7108</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/Inheritance2.html</url>
        <content>a class which is explicitly designed to be the super-class for a set of dependent classes, but
        which does not  itself provide sufficient functionality for independent use, is known as an abstract base
        class (abc). a method in a abc which must be overridden in each descendent class (and will fault if not)
        is known as an abstract method.</content>
</document>
</result>
<result id ="5">
<document for-domain="cmput301" id="ng-7111">
        <title>7111</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/Inheritance2.html</url>
        <content>a class which is explicitly designed to be the superclass for a set of dependent classes, but which does
        not itself provide sufficient functionality for independent use, is known as an abstract base class (abc).</content>
</document>
</result>
<result id ="6">
<document for-domain="cmput301" id="ng-7193">
        <title>7193</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheDesignProcess.html</url>
        <content>also known asdesign, build, test, design, build, test</content>
</document>
</result>
<result id ="7">
<document for-domain="cmput301" id="ng-7202">
        <title>7202</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheDesignProcess2.html</url>
        <content>process-oriented, basis for much of design rationale research</content>
</document>
</result>
<result id ="8">
<document for-domain="cmput301" id="ng-7204">
        <title>7204</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html</url>
        <content>user interfaces are designed to support the user to accomplish his/her tasks by using the system through
        the interface. to develop learnable and usable interfaces, we need to better understand the user.</content>
</document>
</result>
<result id ="9">
<document for-domain="cmput301" id="ng-7206">
        <title>7206</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html</url>
        <content>humans are limited in their capacity to process information. this has important implications for
        design.</content>
</document>
</result>
<result id ="10">
<document for-domain="cmput301" id="ng-7219">
        <title>7219</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman.html</url>
        <content>user interfaces are designed to support the user to accomplish his/her tasks by using the system through
        the interface. to develop learnable and usable interfaces, we need to better understand the user.</content>
</document>
</result>
<result id ="11">
<document for-domain="cmput301" id="ng-7221">
        <title>7221</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman.html</url>
        <content>humans are limited in their capacity to process information. this has important implications for
        design.</content>
</document>
</result>
<result id ="12">
<document for-domain="cmput301" id="ng-7317">
        <title>7317</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/cscw.html</url>
        <content>in design, management and research, we want to:</content>
</document>
</result>
<result id ="13">
<document for-domain="cmput301" id="ng-7339">
        <title>7339</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/hagissues4.html</url>
        <content>when designing the hypertext structure of your help file:</content>
</document>
</result>
<result id ="14">
<document for-domain="cmput301" id="ng-7341">
        <title>7341</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/help.html</url>
        <content>implementation and presentation both need to be considered in designing user support.</content>
</document>
</result>
<result id ="15">
<document for-domain="cmput301" id="ng-7364">
        <title>7364</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html</url>
        <content>the central theme in norman s book is that we need to make design more human-centred and that poor
        designs can make people look unnecessarily stupid. one of the features of user-centred design is that it reduces the
        amount or severity of human error either by making errors less likely or else by making any errors that do occur
        easier to recover from.</content>
</document>
```

```
</result>
<result id ="16">
<document for-domain="cmput301" id="ng-7365">
    <title>7365</title>
    <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html</url>
    <content>in norman s analysis, errors (slips or mistakes) occur when actions go wrong, either because the goal
    behind an action is wrong (a mistake) or else because the right goal is executed wrongly (a slip). in this lecture we
    review norman s theory of action, which has been very influential in the areas of human-machine systems design
    and human-computer interaction.</content>
</document>
</result>
<result id ="17">
<document for-domain="cmput301" id="ng-7383">
    <title>7383</title>
    <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/pj1.html</url>
    <content>your design document should contain most of the design components we have discussed in class, moving
    from the  problem statement to the high level design stages. as a reminder, we include the following short list of
    design elements:</content>
</document>
</result>
<result id ="18">
<document for-domain="cmput301" id="ng-7368">
    <title>7368</title>
    <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html</url>
    <content>for instance, a goal might be to make hot water. the action is switching a kettle on. the resulting event in
    the world is the heating of the water, and the feedback from the world is the whistling of the kettle when the water
    is boiling. note here that it is the design of the kettle and the electrical system that boils the water and provides the
    feedback.</content>
</document>
</result>
<result id ="19">
<document for-domain="cmput301" id="ng-7380">
    <title>7380</title>
    <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/pj1.html</url>
    <content>it is your task for this assignment to design the application s interface as well as its internal classes and
    their behaviors. to do that, you have</content>
</document>
</result>
</response>
```

(B) Search results for ‘design’ from the Latent Semantic Indexing algorithm

```
<?xml version="1.0" encoding="UTF-8"?>
<response for-domain='cmput301' for-search='design'>
<result id="0">
<document for-domain="cmput301" id="ng-6935">
	<title>6935</title>
	<url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignEvaluation.html</url>
	<content>goals of evaluation</content>
</document>
</result>
<result id="1">
<document for-domain="cmput301" id="ng-6950">
	<title>6950</title>
	<url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html</url>
	<content>dialogue design takes as input the outputs of task and user modeling</content>
</document>
</result>
<result id="2">
<document for-domain="cmput301" id="ng-7193">
	<title>7193</title>
	<url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheDesignProcess.html</url>
	<content>also known asdesign, build, test, design, build, test</content>
</document>
</result>
<result id="3">
<document for-domain="cmput301" id="ng-7206">
	<title>7206</title>
	<url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html</url>
	<content>humans are limited in their capacity to process information. this has important implications for
	design.</content>
</document>
</result>
<result id="4">
<document for-domain="cmput301" id="ng-7221">
	<title>7221</title>
	<url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman.html</url>
	<content>humans are limited in their capacity to process information. this has important implications for
	design.</content>
</document>
</result>
<result id="5">
<document for-domain="cmput301" id="ng-7317">
	<title>7317</title>
	<url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/cscw.html</url>
	<content>in design, management and research, we want to:</content>
</document>
</result>
<result id="6">
<document for-domain="cmput301" id="ng-7368">
	<title>7368</title>
	<url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html</url>
	<content>for instance, a goal might be to make hot water. the action is switching a kettle on. the resulting event in
	the world is the heating of the water, and the feedback from the world is the whistling of the kettle when the water
	is boiling. note here that it is the design of the kettle and the electrical system that boils the water and provides the
	feedback.</content>
</document>
</result>
<result id="7">
<document for-domain="cmput301" id="ng-7383">
	<title>7383</title>
	<url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/pj1.html</url>
	<content>your design document should contain most of the design components we have discussed in class, moving
	from the  problem statement to the high level design stages. as a reminder, we include the following short list of
	design elements:</content>
</document>
```

```
</result>
<result id="8">
<document for-domain="cmput301" id="ng-7365">
        <title>7365</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html</url>
        <content>in norman s analysis, errors (slips or mistakes) occur when actions go wrong, either because the goal
        behind an action is wrong (a mistake) or else because the right goal is executed wrongly (a slip). in this lecture we
        review norman s theory of action, which has been very influential in the areas of human-machine systems design
        and human-computer interaction .</content>
</document>
</result>
<result id="9">
<document for-domain="cmput301" id="ng-6947">
        <title>6947</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/Designing_A_OO_System.html</url>
        <content>the graphical editor presents the user with an initially clear drawing areaandwaits</content>
</document>
</result>
<result id="10">
<document for-domain="cmput301" id="ng-6961">
        <title>6961</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html</url>
        <content></content>
</document>
</result>
<result id="11">
<document for-domain="cmput301" id="ng-6962">
        <title>6962</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html</url>
        <content>petri nets are more powerful than finite automata because they can count any number of objects; e. g.
        1000 coke machines and 600 coins. a fsn is limited by its number of states.</content>
</document>
</result>
<result id="12">
<document for-domain="cmput301" id="ng-6945">
        <title>6945</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/Designing_A_OO_System.html</url>
        <content> </content>
</document>
</result>
<result id="13">
<document for-domain="cmput301" id="ng-6958">
        <title>6958</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html</url>
        <content>the stn is accurate, but it is difficult and un-intuitive to produce and it is difficult to understand</content>
</document>
</result>
<result id="14">
<document for-domain="cmput301" id="ng-6944">
        <title>6944</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/Designing_A_OO_System.html</url>
        <content></content>
</document>
</result>
<result id="15">
<document for-domain="cmput301" id="ng-7380">
        <title>7380</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/pj1.html</url>
        <content>it is your task for this assignment to design the application s interface as well as its internal classes and
        their behaviors. to do that, you have</content>
</document>
</result>
<result id="16">
<document for-domain="cmput301" id="ng-6937">
        <title>6937</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignEvaluation.html</url>
        <content>what is the point of evaluating an interface without users?</content>
```

```
</document>
</result>
<result id="17">
<document for-domain="cmput301" id="ng-7364">
        <title>7364</title>
        <url>http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html</url>
        <content>the central theme in norman s book is that we need to make design more human-centred and that poor
        designs can make people look unnecessarily stupid. one of the features of user-centred design is that it reduces the
        amount or severity of human error either by making errors less likely or else by making any errors that do occur
        easier to recover from.</content>
</document>
</result>
</response>
```

(C) Merged search results from NBC and LSI

```
<?xml version="1.0" encoding="UTF-8"?>
<response for-domain='cmput301' for-search='design'>
<result id="0">
<document for-domain="cmput301" id="ng-6950">
    <title>  6950  </title>
    <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html  </url>
    <content>  dialogue design takes as input the outputs of task and user modeling  </content>
</document>
</result>
<result id="1">
<document for-domain="cmput301" id="ng-7193">
    <title>  7193  </title>
    <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheDesignProcess.html  </url>
    <content>  also known asdesign, build, test, design, build, test  </content>
</document>
</result>
<result id="2">
<document for-domain="cmput301" id="ng-7206">
    <title>  7206  </title>
    <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html  </url>
    <content>  humans are limited in their capacity to process information. this has important implications for design.
    </content>
</document>
</result>
<result id="3">
<document for-domain="cmput301" id="ng-7221">
    <title>  7221  </title>
    <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman.html  </url>
    <content>  humans are limited in their capacity to process information. this has important implications for design.
    </content>
</document>
</result>
<result id="4">
<document for-domain="cmput301" id="ng-7317">
    <title>  7317  </title>
    <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/cscw.html  </url>
    <content>  in design, management and research, we want to:  </content>
</document>
</result>
<result id="5">
<document for-domain="cmput301" id="ng-7364">
    <title>  7364  </title>
    <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html  </url>
    <content>  the central theme in norman s book is that we need to make design more human-centred and that poor
    designs can make people look unnecessarily stupid. one of the features of user-centred design is that it reduces the
    amount or severity of human erroreither by making errors less likely or else by making any errors that do occur
    easier to recover from.  </content>
</document>
</result>
<result id="6">
<document for-domain="cmput301" id="ng-7365">
    <title>  7365  </title>
    <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html  </url>
    <content>  in norman s analysis, errors (slips or mistakes) occur when actions go wrong, either because the goal
    behind an action is wrong (a mistake) or else because the right goal is executed wrongly (a slip). in this lecture we
    review norman s theory of action, which has been very influential in the areas of human-machine systems design
    and human-computer interaction.  </content>
</document>
</result>
<result id="7">
<document for-domain="cmput301" id="ng-7383">
    <title>  7383  </title>
    <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/pj1.html  </url>
    <content>  your design document should contain most of the design components we have discussed in class,
        moving from the problem statement to the high level design stages. as a reminder, we include the following short
        list of design elements:  </content>
</document>
</result>
<result id="8">
<document for-domain="cmput301" id="ng-7368">
        <title>  7368  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/onNorman.html  </url>
        <content>  for instance, a goal might be to make hot water. the action is switching a kettle on. the resulting event
        in the world is the heating of the water, and the feedback from the world is the whistling of the kettle when the
        water is boiling. note here that it is the design of the kettle and the electrical system that boils the water and
        provides the feedback.  </content>
</document>
</result>
<result id="9">
<document for-domain="cmput301" id="ng-7380">
        <title>  7380  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/pj1.html  </url>
        <content>  it is your task for this assignment to design the application s interface as well as its internal classes and
        their behaviors. to do that, you have  </content>
</document>
</result>
<result id="10">
<document for-domain="cmput301" id="ng-6935">
        <title>  6935  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignEvaluation.html  </url>
        <content>  goals of evaluation  </content>
</document>
</result>
<result id="11">
<document for-domain="cmput301" id="ng-6947">
        <title>  6947  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/Designing_A_OO_System.html  </url>
        <content>  the graphical editor presents the user with an initially clear drawing area andwaits  </content>
</document>
</result>
<result id="12">
<document for-domain="cmput301" id="ng-6961">
        <title>  6961  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html  </url>
        <content></content>
</document>
</result>
<result id="13">
<document for-domain="cmput301" id="ng-6962">
        <title>  6962  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html  </url>
        <content>  petri nets are more powerful than finite automata because they can count any number of objects; e. g.
        1000 coke machines and 600 coins. a fsn is limited by its number of states.  </content>
</document>
</result>
<result id="14">
<document for-domain="cmput301" id="ng-6945">
        <title>  6945  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/Designing_A_OO_System.html  </url>
        <content>  </content>
</document>
</result>
<result id="15">
<document for-domain="cmput301" id="ng-6958">
        <title>  6958  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DialogNotations.html  </url>
        <content>  the stn is accurate, but it is difficult and un-intuitive to produce and it is difficult to
        understand</content>
</document>
</result>
<result id="16">
```

```
<document for-domain="cmput301" id="ng-6944">
        <title>  6944  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/Designing_A_OO_System.html  </url>
        <content></content>
</document>
</result>
<result id="17">
<document for-domain="cmput301" id="ng-6937">
        <title>  6937  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignEvaluation.html  </url>
        <content>  what is the point of evaluating an interface without users?  </content>
</document>
</result>
<result id="18">
<document for-domain="cmput301" id="ng-6936">
        <title>  6936  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignEvaluation.html  </url>
        <content>  evaluating designs  </content>
</document>
</result>
<result id="19">
<document for-domain="cmput301" id="ng-6938">
        <title>  6938  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignPatterns.html  </url>
        <content>  a design pattern systematically names, motivates, and explains a general designa recurring design
        problem in object-oriented systems  </content>
</document>
</result>
<result id="20">
<document for-domain="cmput301" id="ng-6939">
        <title>  6939  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/DesignPatterns.html  </url>
        <content>  a design pattern is defined in terms of  </content>
</document>
</result>
<result id="21">
<document for-domain="cmput301" id="ng-7108">
        <title>  7108  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/Inheritance2.html  </url>
        <content>  a class which is explicitly designed to be the super-class for a set of dependent classes, but which does
        not itself provide sufficient functionality for independent use, is known as an abstract base class (abc). a method in a
        abc which must be overridden in each descendent class (and will fault if not) is known as an abstract method.
        </content>
</document>
</result>
<result id="22">
<document for-domain="cmput301" id="ng-7111">
        <title>  7111  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/Inheritance2.html  </url>
        <content>  a class which is explicitly designed to be the superclass for a set of dependent classes, but which does
        not itself provide sufficient functionality for independent use, is known as an abstract base class (abc).  </content>
</document>
</result>
<result id="23">
<document for-domain="cmput301" id="ng-7202">
        <title>  7202  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheDesignProcess2.html  </url>
        <content>  process-oriented, basis for much of design rationale research  </content>
</document>
</result>
<result id="24">
<document for-domain="cmput301" id="ng-7204">
        <title>  7204  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman-forme.html  </url>
        <content>  user interfaces are designed to support the user to accomplish his/her tasks by using the system
        through the interface. to develop learnable and usable interfaces, we need to better understand the user.
        </content>
```

```
</document>
</result>
<result id="25">
<document for-domain="cmput301" id="ng-7219">
        <title>  7219  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/TheHuman.html  </url>
        <content>  user interfaces are designed to support the user to accomplish his/her tasks by using the system
        through the interface. to develop learnable and usable interfaces, we need to better understand the user.
        </content>
</document>
</result>
<result id="26">
<document for-domain="cmput301" id="ng-7339">
        <title>  7339  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/hagissues4.html  </url>
        <content>  when designing the hypertext structure of your help file:  </content>
</document>
</result>
<result id="27">
<document for-domain="cmput301" id="ng-7341">
        <title>  7341  </title>
        <url>  http://dogx40cdy160c.ab.hsia.telus.net/lectures/help.html  </url>
        <content>  implementation and presentation both need to be considered in designing user support.  </content>
</document>
</result>
</response>
```