Chapter Five

Technical Details and Implementation

5.1 Introduction

This chapter presents deeper technical details and implementation specifications of aspects of the proposed Adaptive e-Learning Models and highlights some of the intelligent services. Upon the start to build the proposed model, some challenges became clear. Challenges include:

- Lack of ability to access Internet in certain occasions. Internet is not available to all students all the time.
- Large size of some learning resources, mainly video lectures.
- LOs copyrights prevent us from uploading them online.
- There is a need to provide meanings of learning to students all the time.

Besides, one of the proposed solutions to cheating problem was conducting exams through a desktop application that provides more secure environment than the Web. Hence, the idea of presenting a Desktop Learning Environment that provides learning features while offline, and integrates with the Proposed Model Adaptive LMS when the application comes online became a clear need.

Technical Implementation consists of three complementary parts:

1. **Student Desktop Application:** available to registered students to download, including instructor recommended LOs for course topics based on student registration information.
2. **Student e-Learning Environment and Adaptive Features:** Adaptive LMS web site where student registers, download the application, update profile and learning preferences, and connects the application so the students' usage and learning data is synchronized automatically, and the

desktop application itself where students attends the Adaptive e-Learning Model and use the different services and features.

3. **Instructor Portal:** the Adaptive LMS administration where instructors manage students, courses, topics, learning objects, instructors' data, and other system configurations that affects the learning process.

SOA design pattern presents system and system components as collection of reusable services with standard interface that can be used within and among different applications, and allows shareability and integration between different systems. In chapter three different adaptive model components were presented, and in chapter four different intelligent services specifications are available. Technical details and implementation of selected services are presented here.

Services are needed to support proposed Adaptive e-Learning model. System Services are services that present one or more of the adaptive and intelligent functionalities proposed by the model, and include services that present functionalities needed to achieve required tasks. Services include: Students Manager Service, Intelligent Student Tracker Service, Learning Objects Manager Service, Intelligent LOs Recommender Service, Intelligent Meeting Manager for Suspended Students Service, and Intelligent Document Classifier.

5.2 Intelligent Student Tracking Service

It is the service that tracks students' status during the learning journey. Figure 5.1 displays student transition state diagram which is one of the states:

- **Complete Profile:** This is the state that a student is in when s/he fills all the learning preferences required fields.
- **Currently Studying:** Incase student has completed learning preferences profile, and is not suspended.
- **Partial Exam Completion:** Student doesn't have a due exam yet.

- **Exam Due:** Student is scheduled to have an exam.

This service provides a standard interface that can be queried to define the state of the student. Input parameters are: authentication token of the calling service, and student ID, and the service response is a number from 1 : 5 indicating the student state. Intelligent Student Tracker Service utilizes Students Manager Service Helper.

Figure 5.2 presents the loading screen students see at running the application. Figure 5.3 highlights the Intelligent Student Tracker in action for new students, when the system identifies that student has not completed the learning profile. Intelligent Student Tracker directs student to complete learning profile.

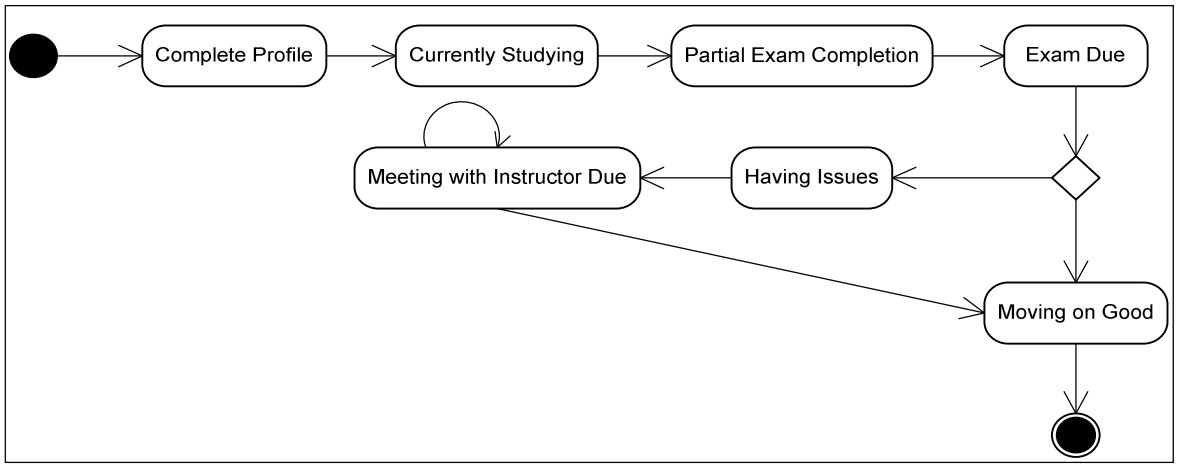


Figure 5.1: Student State Transition Diagram

Figure 5.2: Application Loading Screen

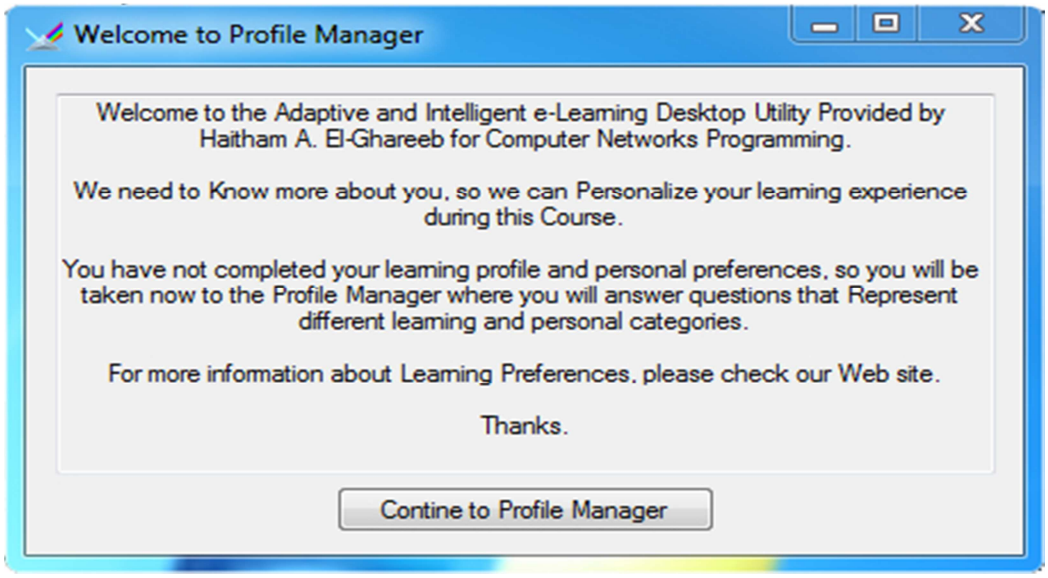


Figure 5.3: Welcome Message indicating the Need to Build Learning Profile

Intelligent Student Tracker service needs two services: Students Manager Service and Students Usage Data Manager Service. Students Manager Service tracks students' data themselves while Students Usage Data Manager Service tracks students' different usage data of the system.

5.3 Students Manager Service

Students Manager Service enables different systems to manage students' data. Students' data include learning preferences, learning profiles, time table, and students' usage data. Students Manager Service include three inner services, they are: Students General Data Manager, Students Learning Profile Manager, and Students Usage Data Manager. While Students General Data Manager handles basic Create, Retrieve, Update, and Delete (CRUD) operations for basic information like username, password, email, and other general data, Students Learning Profile Manager, and Students Usage Data Manager are more of pedagogical importance to the proposed model. Figure 5.4 presents the required database tables to support Students Learning Profile Manager Service.

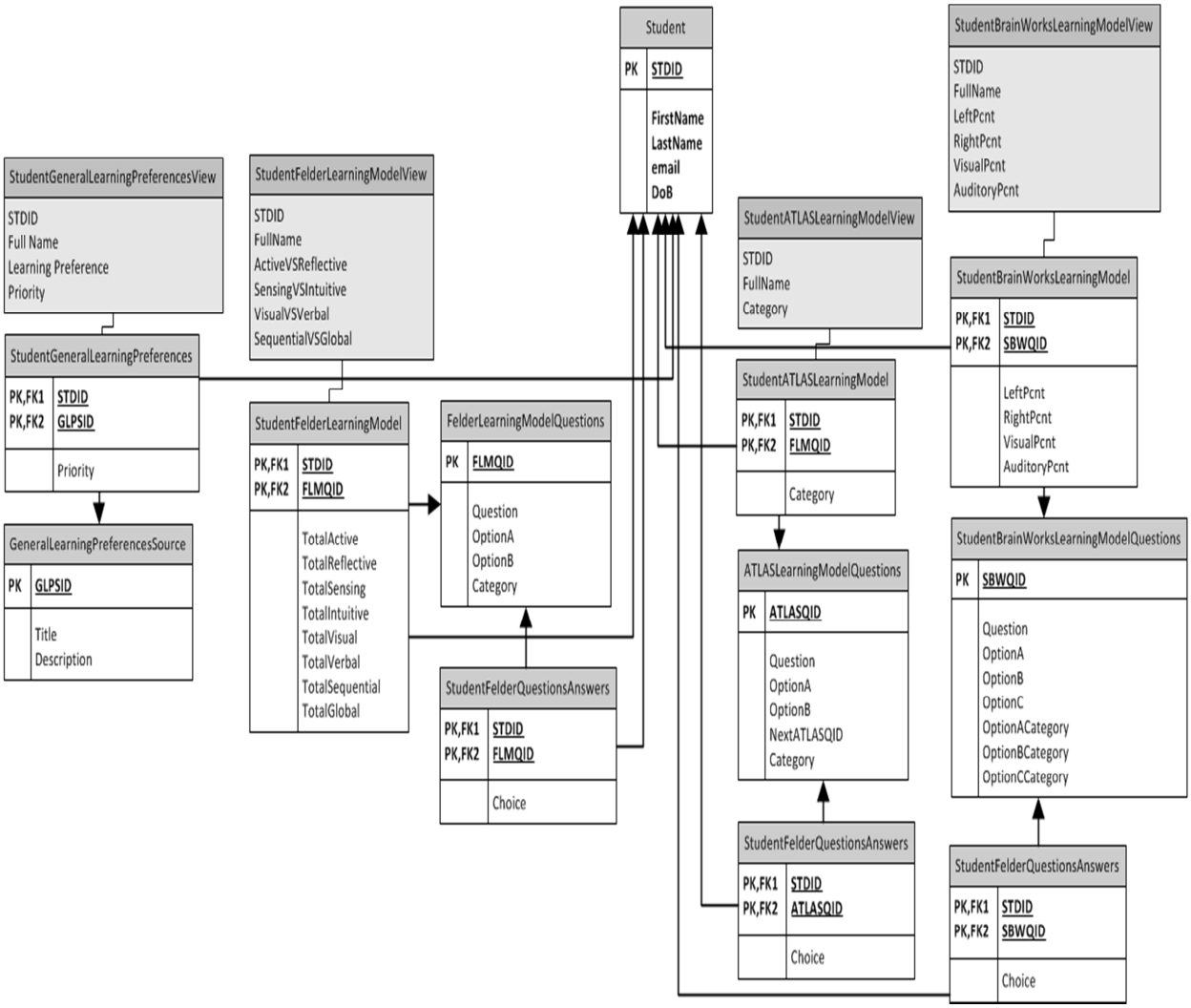


Figure 5.4: Students Learning Profile Manager Database Tables

Four different learning profiles are available for each student: General, Felder, ATLAS, and Brain Works. Figure 5.5 presents the General Profile Manager screen. Figure 5.6 presents Felder Learning Profile Manager and displays sample of Felder Questions. Felder questions sample include:

1. I understand something better after I:

   try it out. think it through.

2. I would rather be considered

   realistic. innovative.

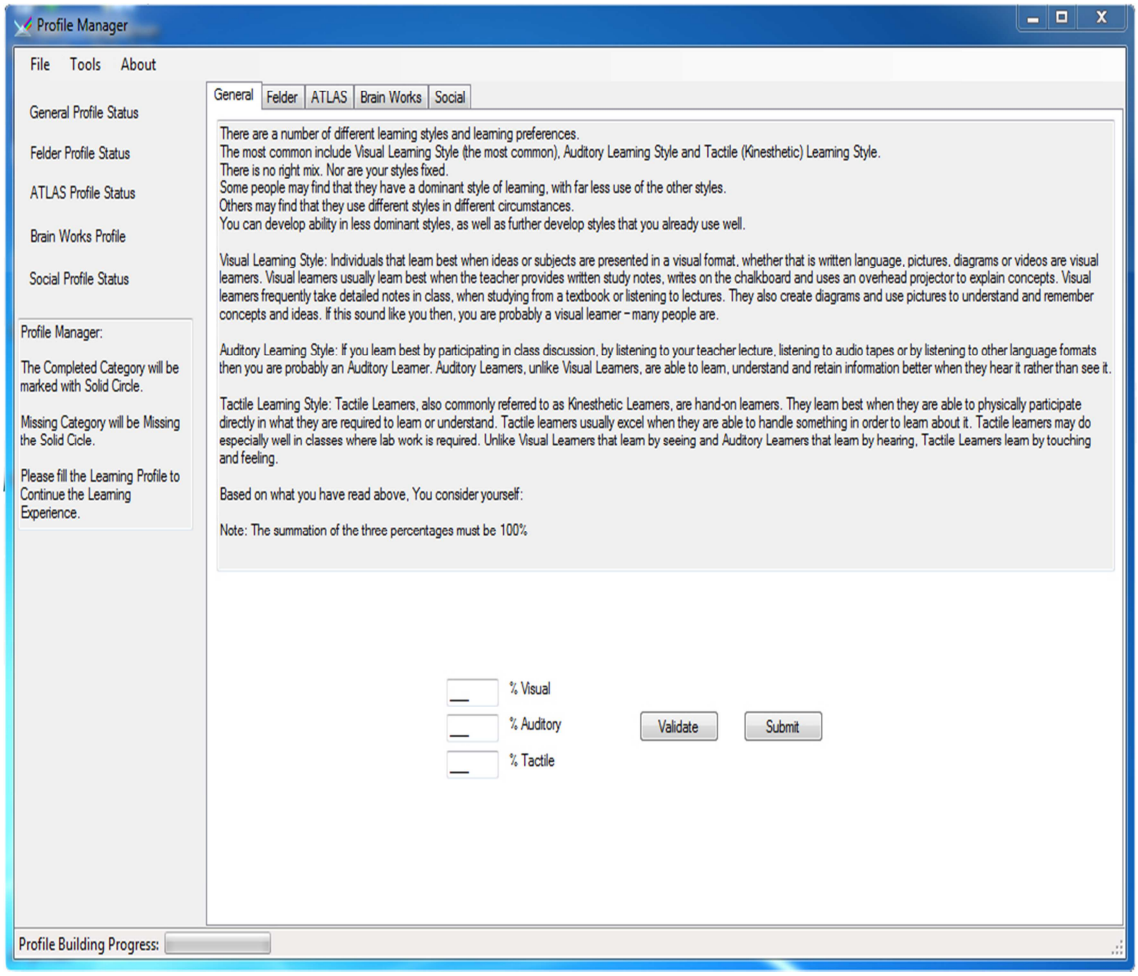


Figure 5.5: Student Learning Profile Manager Screen - General Learning Profile

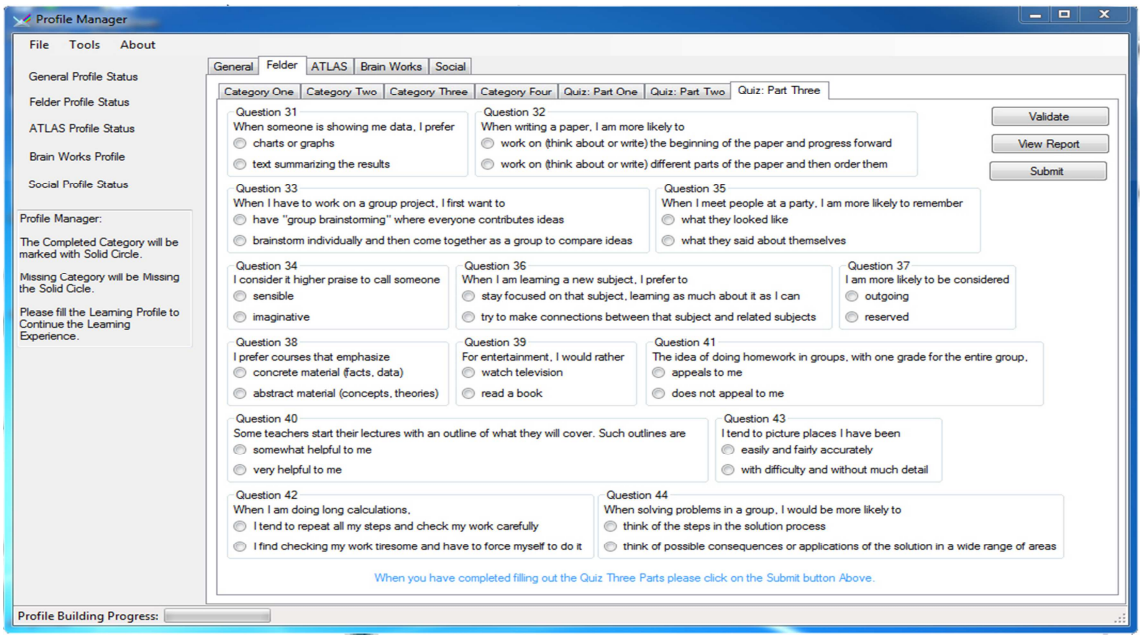


Figure 5.6: Student Learning Profile Manager Screen - Felder Learning Profile

Figure 5.7 presents Felder Learning Style Report. Felder proposes calculation method to identify to which category each student belongs. Figure 5.7 presents an instance of the Felder report.

Figure 5.8 presents ATLAS Learning Style profile manager. ATLAS learning style doesn't rely on questions heavily as Felder does. However, it is a step-by-step series of questions that are capable of identifying students' learning style at the end. Figure 5.9 displays BrainWorks questions sample, and figure 5.10 displays the BrainWorks report. BrainWorks learning style identifies two aspects for each student: auditory or visual, and left or right brain hemisphere directed.

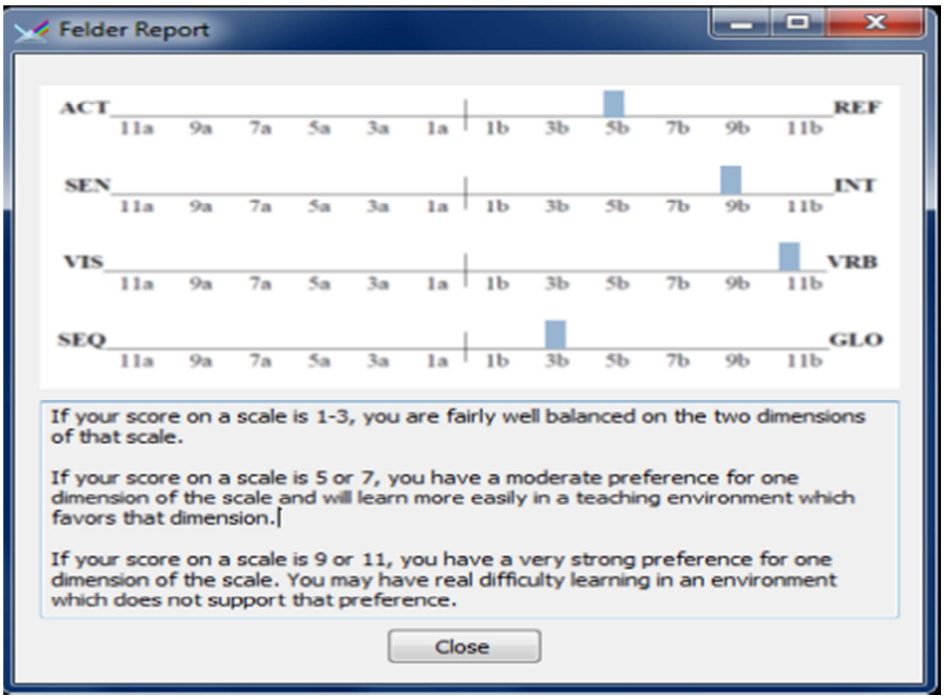


Figure 5.7: Student Learning Profile Manager - Felder Report

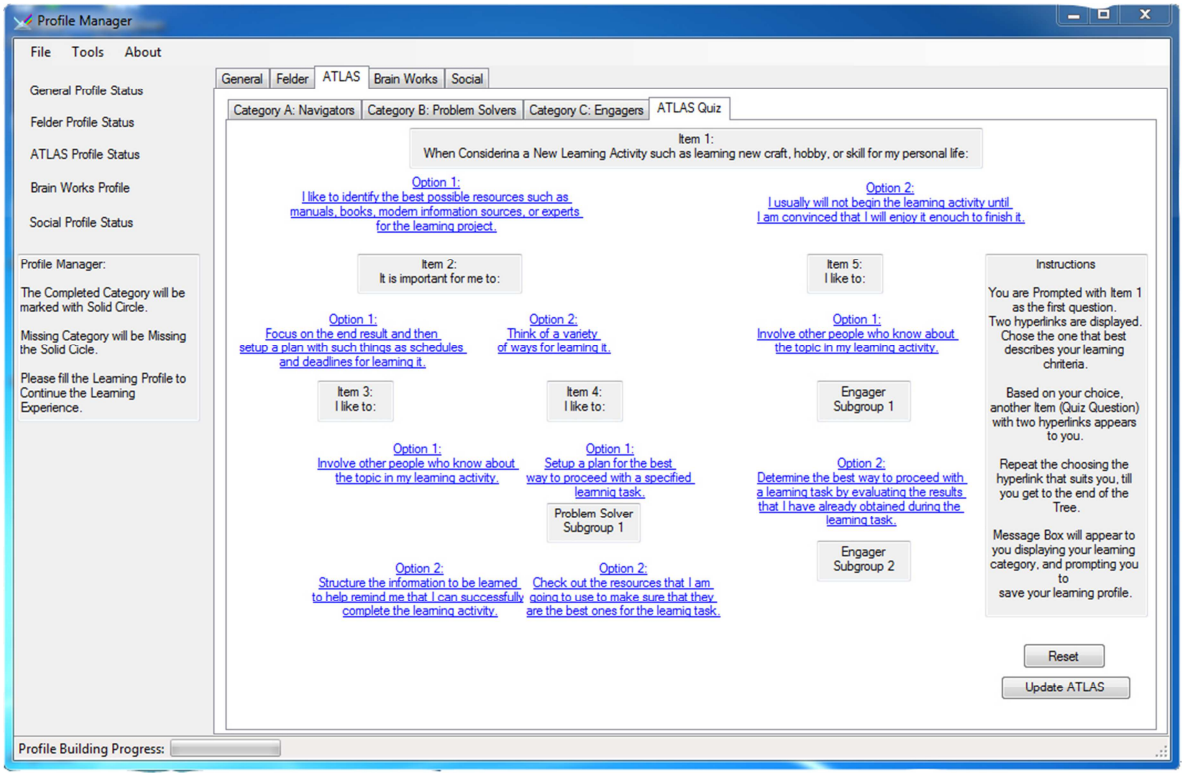


Figure 5.8: Student Learning Profile Manager Screen - ATLAS Learning Profile

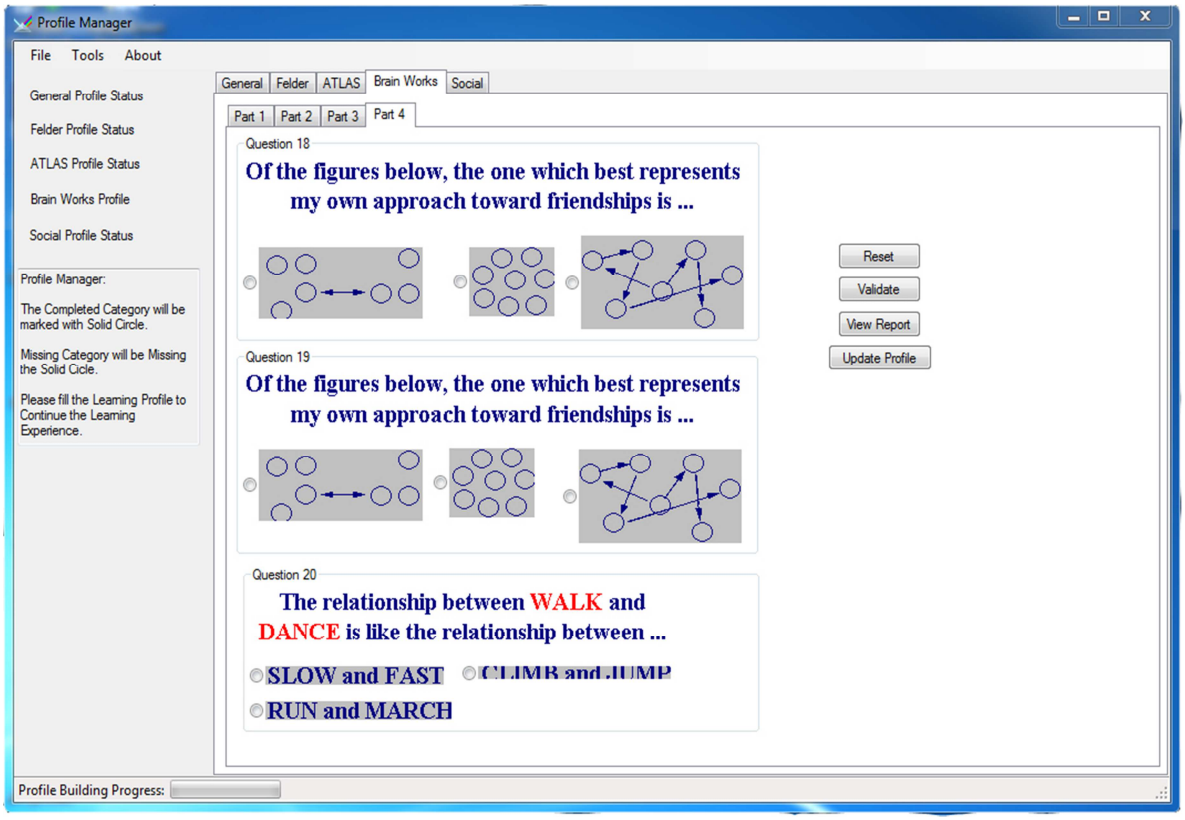


Figure 5.9: Student Learning Profile Manager Screen – Brain Works Learning Profile

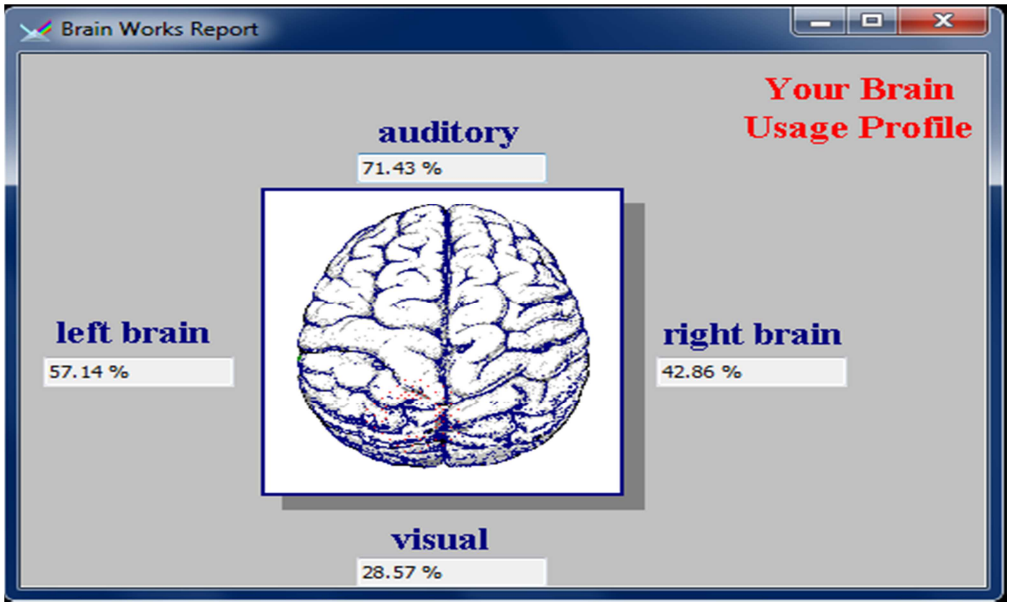


Figure 5.10: Student Learning Profile Manager - Brainworks Report

5.4 Students Usage Data Manager

Student Usage Data Manager keeps track of three student's behavior that are used to adjust the overall adaptive model performance and behavior. Figure 5.11 presents database tables to support this service. Usage data include:

1. **Browsing Behavior:** Intelligent LOs Recommender depends on students browsing through different LOs for initiating relation between different LOs. Tracking Referrer and Target URL helps the Intelligent LOs Recommender based on students browsing relates LOs together.
2. **LOs Study Time:** Among specifications that instructors associate with LOs, they identify the time needed to study it. However, that time might differ from student to another based on their personal differences. Recording the time taken by student to study certain LO is so important for the system. It helps both student and the system.
3. **LOs Ranking:** Students can express their thoughts about each LO in any of the three ways: Like/Not Like, Starring the LO from 1 to 5, or leaving comment/feedback about it.

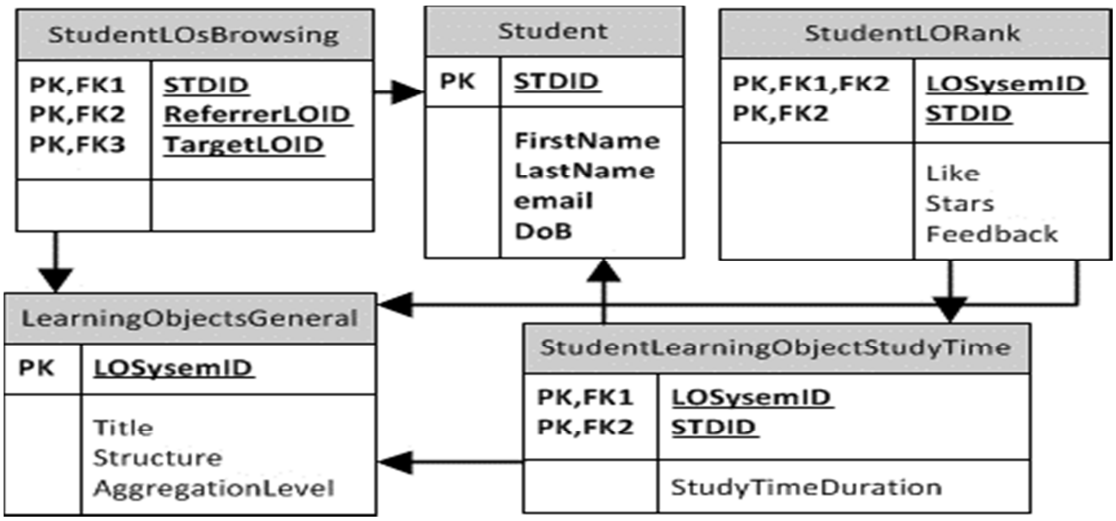


Figure 5.11: Students Usage Manager Database Tables

5.5 Learning Objects Manager Service

LOs metadata needed to support the adaptive e-Learning models is presented in details in chapter three. Figure 5.12 presents an instructor's screen from the instructor portal that is used to edit the metadata of LOs. Figure 5.13 highlights the search functionality available for instructors. Figure 5.14 in two parts presents the needed database tables to support LOs metadata.

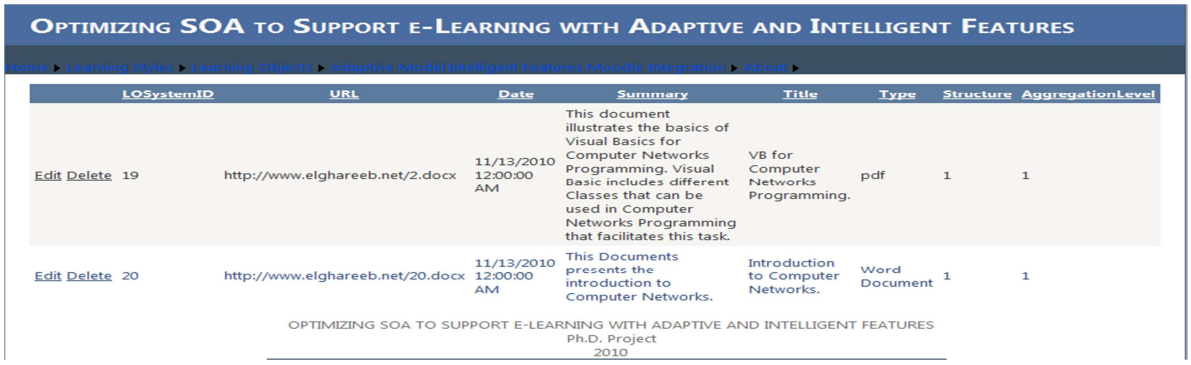


Figure 5.12: LOs Manager from Instructor's Portal

Figure 5.13: LOs Search from Instructor's Portal

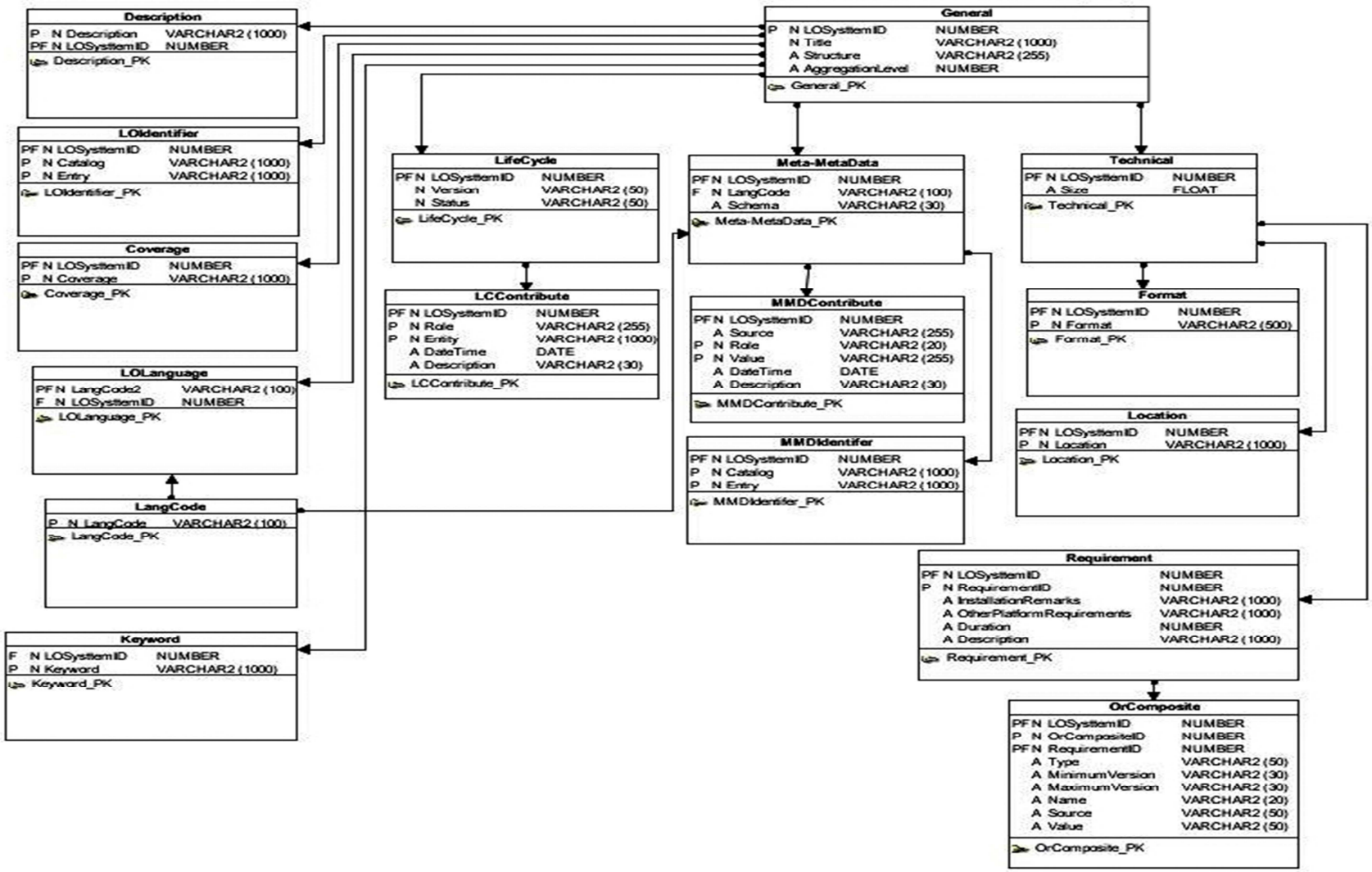


Figure 5.14: LOs Metadata Tables – Part I

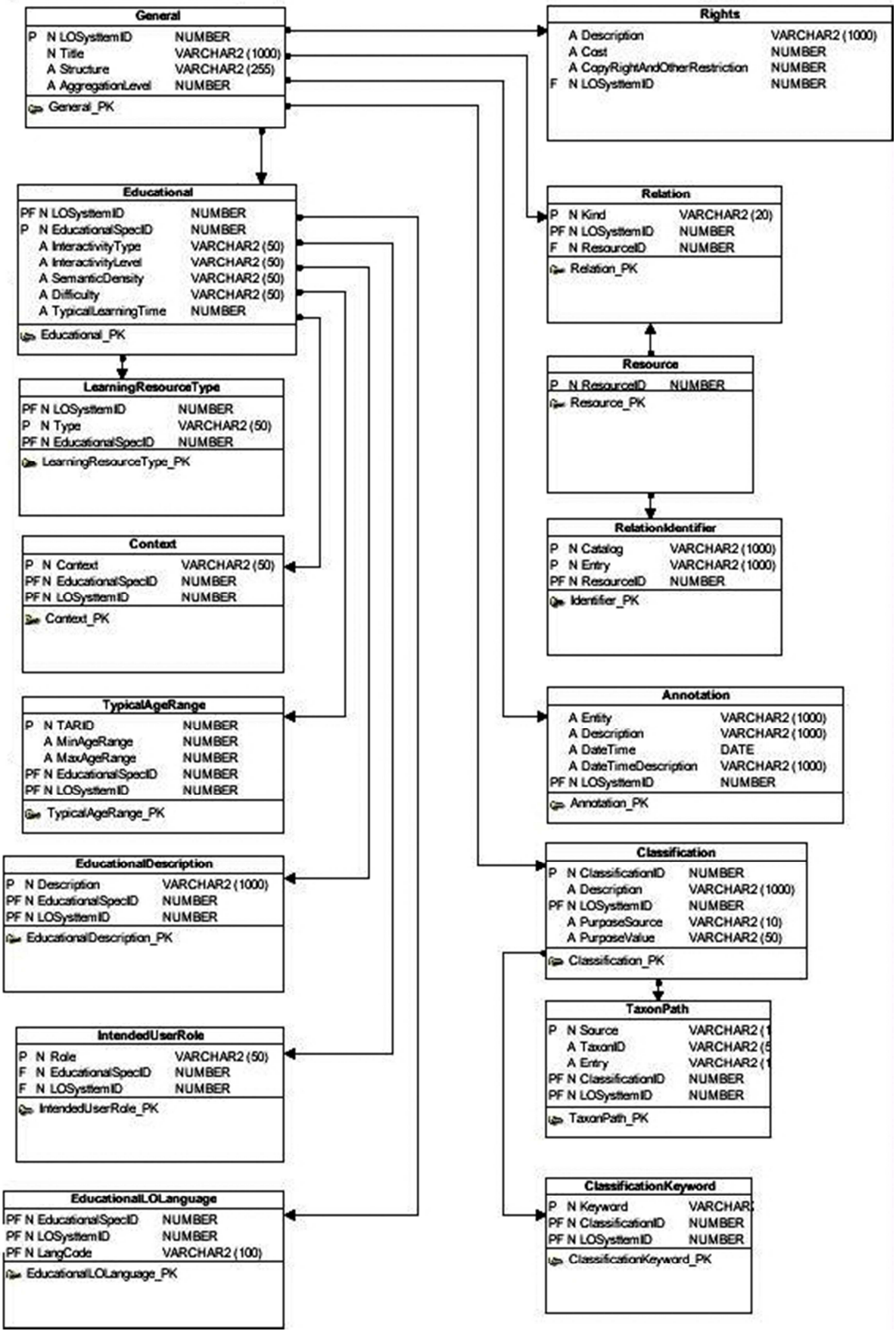


Figure 5.14: LOs Metadata Tables - Part II

5.6 Intelligent LOs Recommender

With the availability of unique 221 different learning objects varying from word documents (.txt, .doc, .docx, .pdf) and presentations (.ppt, .pptx), a service that reads the contents of digital libraries in the previous formats and extracts the contents to be further processed. Processing those 221 files yielded 1623 high quality keywords when compared to those generated from the Web pages. Intelligent LOs Recommender include different modules and services, they are: Pending LOs for Recommendation Manager Module, Crawler Module, and Document Processor Service.

5.6.1 Pending LOs for Recommendation Manager Module

Pending LOs represent LOs that crawler collected information about through querying online search engines, but haven't been processed yet. They are marked as pending waiting for processing either to be stored or deleted. Figure 5.15 presents the Pending LOs for Recommendation Manager database tables that temporarily store collected information about the two LOs types: files, and online resources.

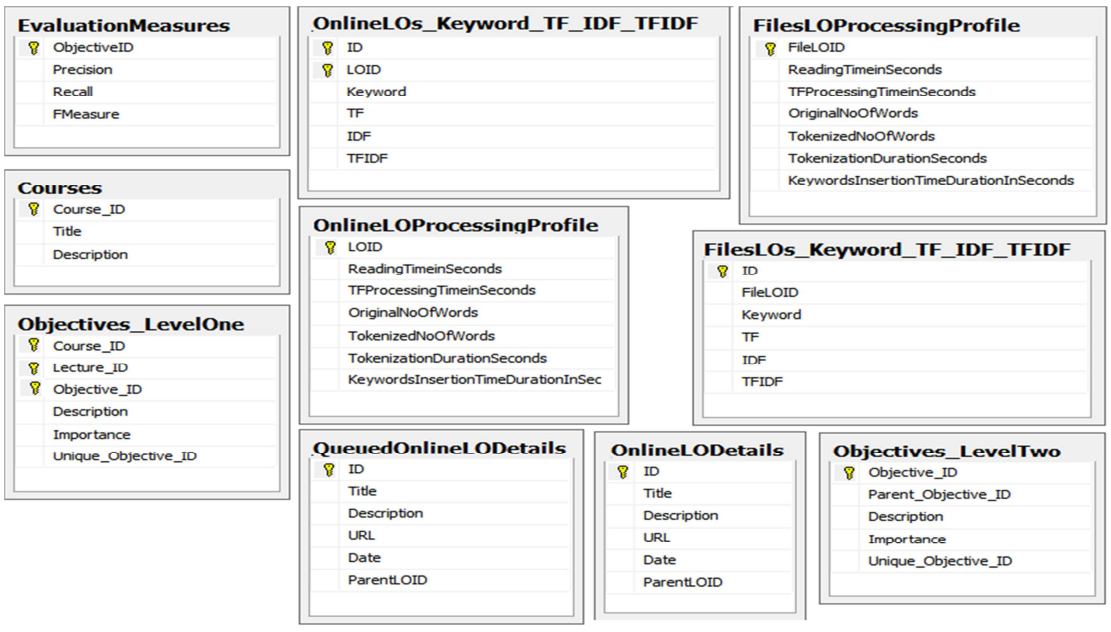


Figure 5.15: Learning Objects Recommendation Process Database Tables

5.6.2 Crawler Module

Crawler is a program that searches for information on the Web, and they are widely used by Web search engines to index all the pages on a site by following the links from page to page. Based on the crawler model, we developed a Web based java crawler that reads Google Search results via an open source library; that is Selenium. Working on the knowledge domain of information systems analysis and design, we ran the search for queries including different combinations of the keywords: System + Analysis + Design + Tutorial. Crawler extracts three main pieces of information for each search result: Title, URL, and Description. Database includes the summary of 2236 records, yielding 161 unique stemmed keywords. Figure 5.16 presents Crawler module package diagram. Figure 5.17 presents Crawler module source code needed to execute. Challenges faced this process include:

- List of search results needed further cleansing before processing because there were list of web sites that yields fake results.
- Search results can't be relied on due to intensive commercial contents.
- Stemmed keywords are sometimes meaningless. Automatically generated keywords will need revision and enhancement.

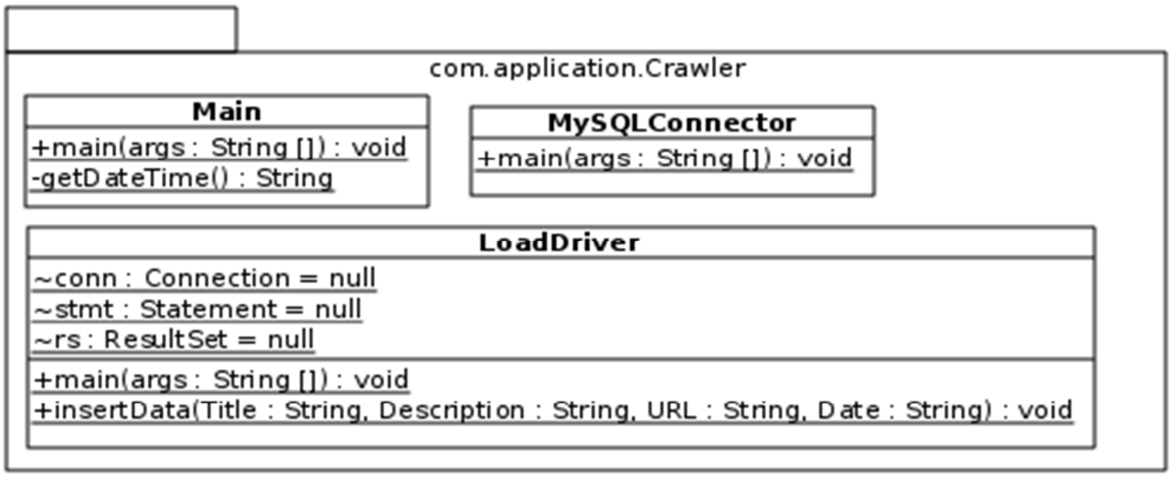


Figure 5.16: Crawler Package Diagram

```java
/*...*/
package crawler;
import java.util.List;
import org.openqa.selenium.By;
import org.openqa.selenium.RenderedWebElement;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
/**
 *
 * @author Haitham A. El-Ghareeb
 */
public class Main {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        String searchKeyWord = "selenium";
        // number of visited page * 10    Ex: 5 page * 10 = 50
        int visitedPage = 50;
        for (int j = 0; j < visitedPage; j=j+10) {
            String url = "http://www.google.com/search?q="+searchKeyWord+"&start="+j+"&sa=N";
            driver.get(url);
            WebElement element = driver.findElement(By.name("q"));
            element.submit();
            long end = System.currentTimeMillis() + 20000;
            while (System.currentTimeMillis() < end) {
                // Browsers which render content (such as Firefox and IE) return "RenderedWebElements"
                RenderedWebElement resultsDiv = (RenderedWebElement) driver.findElement(By.className("r"));
                // If results have been returned, the results are displayed in a drop down.
                if (resultsDiv.isDisplayed()) {
                    break;
                }
            }
            List  allDescription = driver.findElements(By.xpath("//li[@class='g']"));
            List  allTitle = driver.findElements(By.xpath("//h3[@class='r']"));
            for (int i = 0; i < allDescription.size(); i++) {
               try{
                   WebElement title = (WebElement) allTitle.get(i);
                   WebElement description_link = (WebElement) allDescription.get(i);
                   System.out.println("Title :" + title.getText());
                   String des_link = description_link.getText();
                   String des_link_array[] = des_link.split("\n");
                   System.out.println("Description :" + des_link_array[0]);
                   String link = des_link_array[des_link_array.length - 1];
                   // make url from www.google.com/ - Cached - Similar --> www.google.com/
                   link = link.substring(0, link.length() - 18);
                   System.out.println("Link :" + link);
               }catch(Exception e){}
        }
      }
    }
}
```

Figure 5.17: Crawler Source Code

5.6.3 Document Processor Service

Document Processor Service includes phases presented at figure 5.18. Phases are:

- **Tokenization:** A document is treated as a string, and then partitioned into a list of tokens.
- **Removing stop words:** Stop words such as “the”, “a”, “and” are frequently occurring, so the insignificant words need to be removed.
- **Stemming:** Applying stemming algorithm that converts different word forms into similar canonical form. This step is the process of conflating tokens to their root form, e.g. connection to connect, computing to compute.

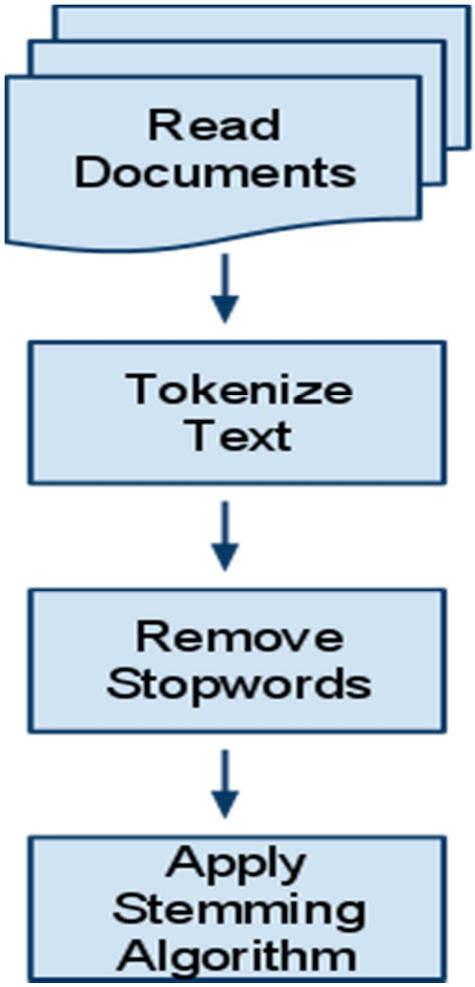


Figure 5.18: Document Processor Phases

5.6.3.1 Tokenizer Module

Document Processor Service tokenizes sentences into words based on any of the following symbols: dot, comma, space, semicolon, and colon. It takes the String to tokenize as an input and returns list of tokenized words. Those words will be the input to the next module: Removing Stop words.

5.6.3.2 Removing Stop words Module

Stop words are words which are filtered out after Tokenizer module processing of text. There is not one definite list of stop words which all tools use. Observations lead to a customized list of stop words that includes the following list as a sample. Course Objectives usually include words like the ones included in the following list.

stopwords = {"details", "course", "lecture", "explains", "discusses", "outside", "inside", "curriculum", "syllabus", "explain", "discuss"}

5.6.3.3 Stemmer Module

Figure 5.19 illustrates the Stemmer Module pseudo code and presents the conducted activities. Stemmer module activities include:

- Split the input using delimiter
- For every processed word:
    - Declare Global Variables.
    - Split the word into two stems.
    - Search for Patterns, and replace them.
    - Return Stemmed word.

Step 1: Split input using Delimiter.

Step 2: For Every word in the Splitted Query, Do:

Step 3: Declare Global Variables

```
Vowels = {"a", "e", "i", "o", "u", "y"}
```

Step 4: Generate Stem (Word) as Stem

Step 4.1: Split the Word into two Stems at any of the characters of in the Vowels Array. Word is splitted to R1 and R2. R1 is the region after the first non-vowel following a vowel, or is the null region at the end of the word if there is no such non-vowel. R2 is the other region.

Step 4.2: Search for the longest among the following suffixes, and, if found and in R1, perform the action indicated.

- tional: replace by tion
- li+: delete if preceded by a valid li-ending

```
Friend Sub ReplaceEndingStep2()
        If EndsWithAndInR1(Indicated Value) Then
            Stem = Stem.Substring(0, Stem.Length –Length) & Replacement Value
            Return
        End If
```

Step 4.3: Search for the longest among the following suffixes, and, if found and in R1, perform the action indicated.

- tional+: replace by tion
- ative*: delete if in R2

```
Friend Sub ReplaceEndingStep3()
        'ational+:   replace by ate
        If EndsWithAndInR1(Indicated Value) Then
            Stem = Stem.Substring(0, Stem.Length – Length) & Replacement Value
            Return
        End If
End Sub
```

Step 4.4: Search for the longest among the following suffixes, and, if found and in R2, perform the action indicated. “al ance ence er ic able ible ant ement ment ent ism ate iti ous ive ize delete “ ion delete if preceded by s or t

```
Friend Sub StripSuffixStep4()
        If EndsWithAndInR2(Any Previous Indicated Value) Then
            Stem = Stem.Remove(Stem.Length – Length of Indicated Value)
            Return
        End If
    End Sub
```

Step 4.5: Search for the following suffixes, and, if found, perform the action indicated. ‘e’ delete if in R2, or in R1 and not preceded by a short syllable, l delete if in R2 and preceded by ‘l’

```
Friend Sub StripSuffixStep5()
        If EndsWithAndInR2("e") OrElse (EndsWithAndInR1("e") AndAlso
IsShortSyllable(Stem.Length - 3) = False) Then
            Stem = Stem.Remove(Stem.Length - 1)
            Return
        End If
        If EndsWithAndInR2("l") AndAlso Stem.EndsWith("ll") Then
            Stem = Stem.Remove(Stem.Length - 1)
            Return
        End If
    End Sub
```

Step 5: Return Joined Stemmed Words

Figure 5.19: Stemmer Module Pseudo Code

5.6.3.4 Query Expansion Module

WordNet is a large lexical database of English used to find synonyms of stemmed keywords generated by Stemmer Module to be further used in the search and recommendation process in later services / modules. Figure 5.20 presents WordNet relational database tables implementation to provide Query Expansion capabilities to the Adaptive e-Learning Model. Each set of synonyms (synset), has a unique index and shares its properties, such as a dictionary definition (lemma).

Query Expansion takes place as follows:

1. Take input (the term to expand).
2. Search WordNet synsets for the term. When Found: Return Synonyms to Expand Query.

Figure 5.21 presents Intelligent LOs Recommender Class Diagram.

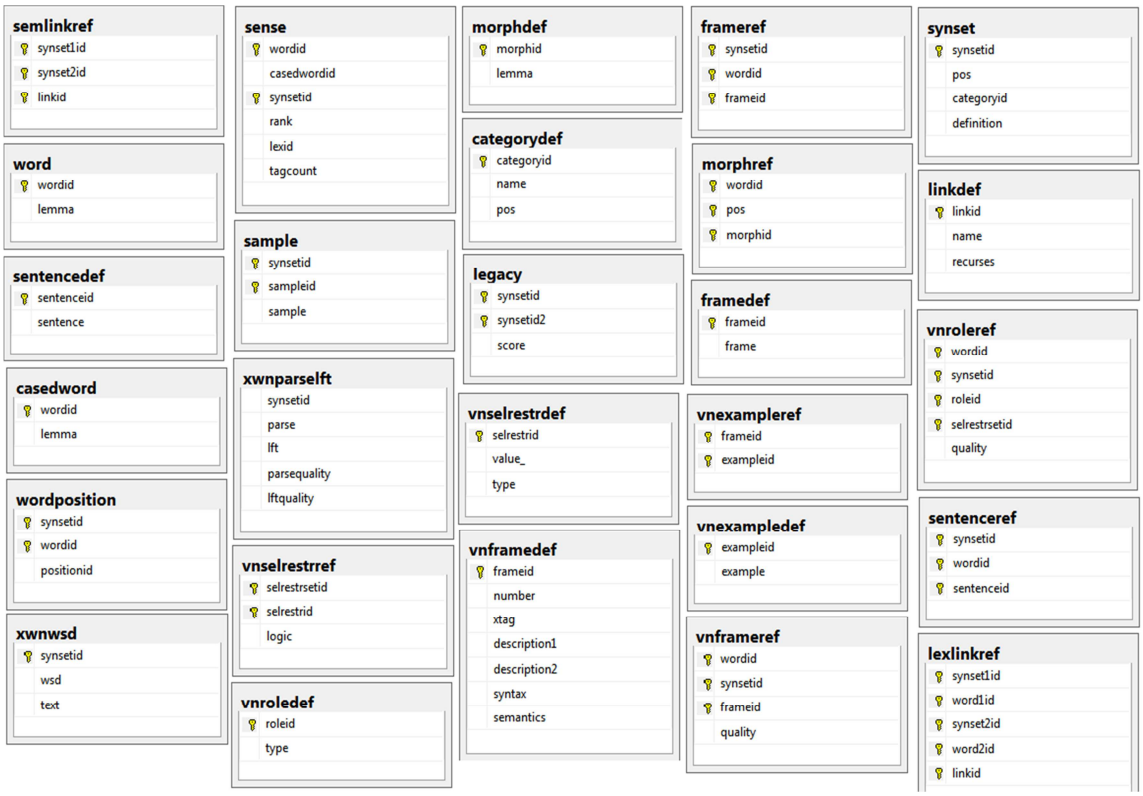


Figure 5.20: WordNet Relational Database Tables

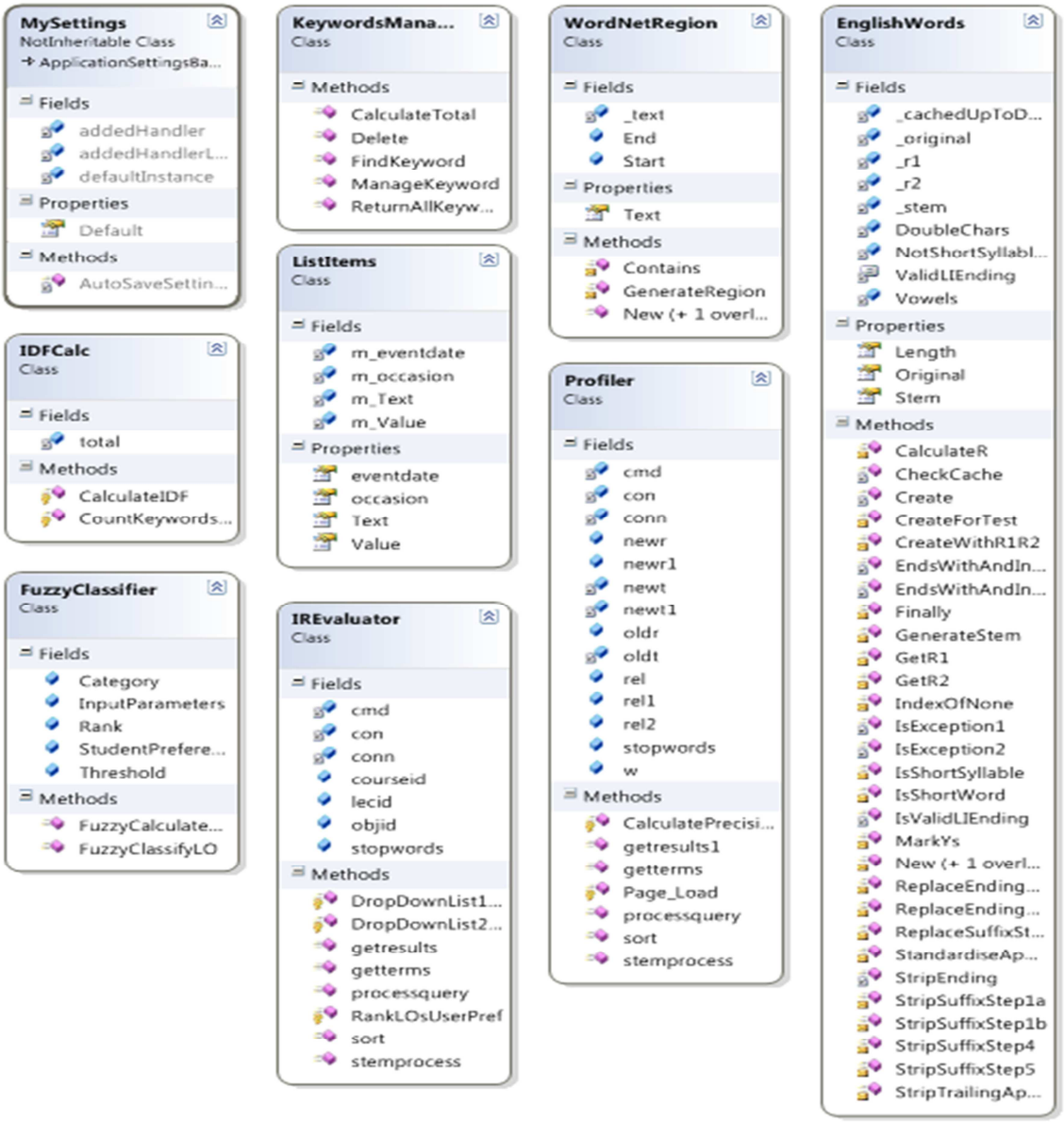


Figure 5.21: Intelligent LOs Recommender Class Diagram

Figure 5.22 presents a screen shot from the instructor's portal that manages extracted keywords from LOs and calculates each document's Term Frequency (TF). Figure 5.23 presents a LOs recommended list from instructor's portal. Instructor can edit and delete recommended LOs list. This functionality is used to fine tune the automatic recommendation process. Figure 5.24 presents LOs recommender in action from student's portal. Figure 5.25 presents Evaluation results of LOs recommender. Evaluation is discussed in details in the next chapter.

Please Choose an LO to Display Keywords TF:

Proposed Adaptive e-Learning Model.pdf

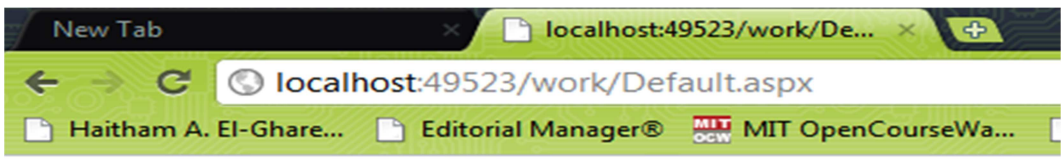
	ID	FileID	Keyword	TF
Delete	1	1	propos	0.0128786072024803
Delete	2	1	adapt	0.0221798235153828
Delete	3	1	e	0.0157405199141426
Delete	4	1	learn	0.0591461960410207
Delete	5	1	model	0.0140710708323396
Delete	6	1	inform	0.0062008108752683
Delete	7	1	system	0.01669449081803
Delete	8	1	technolog	0.00739327450512759
Delete	9	1	process	0.0062008108752683
Delete	10	1	featur	0.00548533269735273

1 2 3

Figure 5.22: Extracted Keywords and TF from Instructor's Portal

	ID	Name	FullName	Extension	Length	CreationTime
Edit Delete	1	Proposed Adaptive e-Learning Model.pdf	http://www.haitham-online.net/SundayMeeting/Site/aquatic/Proposed Adaptive e-Learning Model.pdf	.pdf	2630435	16/10/2010 02:48:12 م
Edit Delete	2	p105-bollacker.pdf	http://www.haitham-online.net/SundayMeeting/Site/aquatic/p105-bollacker.pdf	.pdf	123522	16/10/2010 02:48:21 م
Edit Delete	3	Important Student Modelling-case study.pdf	http://www.haitham-online.net/SundayMeeting/Site/aquatic/Important Student Modelling-case study.pdf	.pdf	292533	16/10/2010 02:48:29 م
Edit Delete	4	Student Model Components.pdf	http://www.haitham-online.net/SundayMeeting/Site/aquatic/Student Model Components.pdf	.pdf	113495	16/10/2010 02:48:30 م

Figure 5.23: LO's Recommendation List from Instructor's Portal



Objective ID	LO URL	Rank
1	http://en.wikipedia.org/wiki/Computer_net	0.03744496
1	http://www.bramjnet.com/vb3/forumdispla	0.04021487
2	http://compnetworking.about.com/	0.04253591
3	http://www.networktutorials.info/	0.0398329

Figure 5.24: Intelligent LOs Recommender in Action

Figure 5.26 presents the Search for Keywords within LOs, LOS URL returned, TF and IDF Calculation results. Figure 5.27 presents the tokenization process, defining tokenized words, and query expansion process.

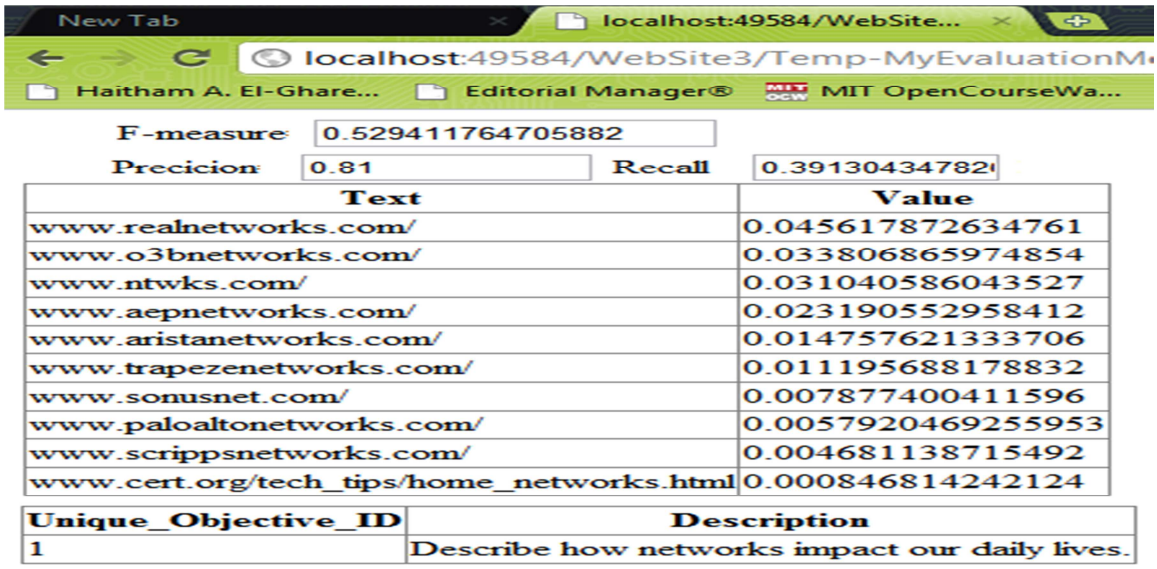


Figure 5.25: Information Retrieval Evaluation Results for Intelligent LOs Recommender

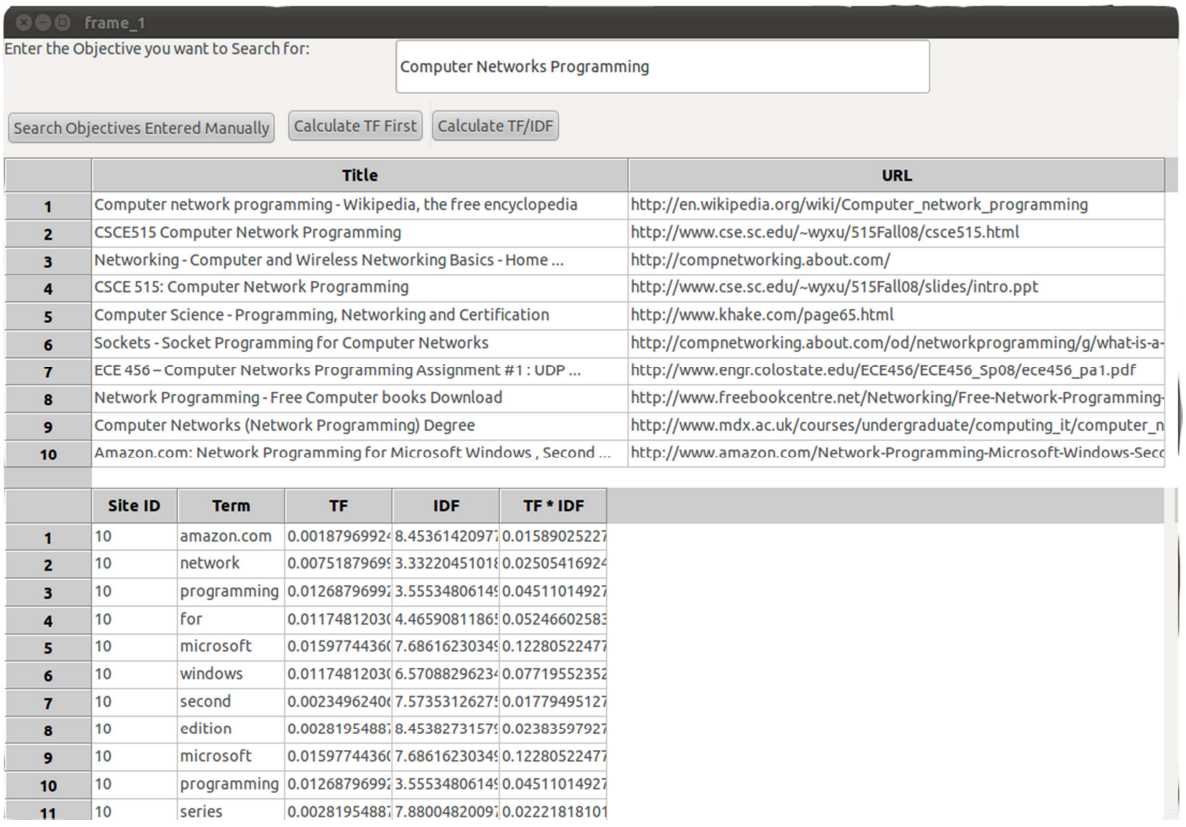


Figure 5.26: Keywords based LOs Search, TF-IDF Calculation for LOs Search Results

frame_1

Enter Text to Tokenize:

Computer Networks Programming and Information Systems

Tokenize Text

	Word	Synsets	Definition
1	Computer	Synset('computer.n.01')	a machine for performing calculations automatically
2		Synset('calculator.n.01')	an expert at calculation (or at operating calculating machines)
3	Networks	Synset('network.n.01')	an interconnected system of things or people
4		Synset('network.n.02')	(broadcasting) a communication system consisting of a group of broadcasting s
5		Synset('net.n.06')	an open fabric of string or rope or wire woven together at regular intervals
6		Synset('network.n.04')	a system of intersecting lines or channels
7		Synset('network.n.05')	(electronics) a system of interconnected electronic components or circuits
8		Synset('network.v.01')	communicate with and within a group
9	Programming	Synset('scheduling.n.01')	setting an order and time for planned events
10		Synset('programming.n.02')	creating a sequence of instructions to enable the computer to do something
11		Synset('program.v.01')	arrange a program of or for
12		Synset('program.v.02')	write a computer program
13	Information	Synset('information.n.01')	a message received and understood
14		Synset('information.n.02')	knowledge acquired through study or experience or instruction
15		Synset('information.n.03')	formal accusation of a crime
16		Synset('data.n.01')	a collection of facts from which conclusions may be drawn
17		Synset('information.n.05')	(communication theory) a numerical measure of the uncertainty of an outcome
18	Systems	Synset('system.n.01')	instrumentality that combines interrelated interacting artifacts designed to wo
19		Synset('system.n.02')	a group of independent but interrelated elements comprising a unified whole
20		Synset('system.n.03')	(physical chemistry) a sample of matter in which substances in different phases
21		Synset('system.n.04')	a complex of methods or rules governing behavior
22		Synset('arrangement.n.03')	an organized structure for arranging or classifying
23		Synset('system.n.06')	a group of physiologically or anatomically related organs or parts
24		Synset('system.n.07')	a procedure or process for obtaining an objective
25		Synset('system.n.08')	the living body considered as made up of interdependent components forming
26		Synset('organization.n.05')	an ordered manner; orderliness by virtue of being methodical and well organize

Figure 5.27: Tokenization and Query Expansion Process in action

5.7 Intelligent Meeting Manager for Suspended Students Service

Intelligent Meeting Manager for Suspended Students is responsible for defining meetings to students that didn't successfully pass the same exam for three times. This is an indication that student needs meeting. The intelligent need is addressed in finding the most appropriate time for both instructor and student from different available time slots. Figure 5.28 presents the required database table to support this service and Figure 5.29 presents the class diagram. Figure 5.30 presents Data Access Layer class diagram.

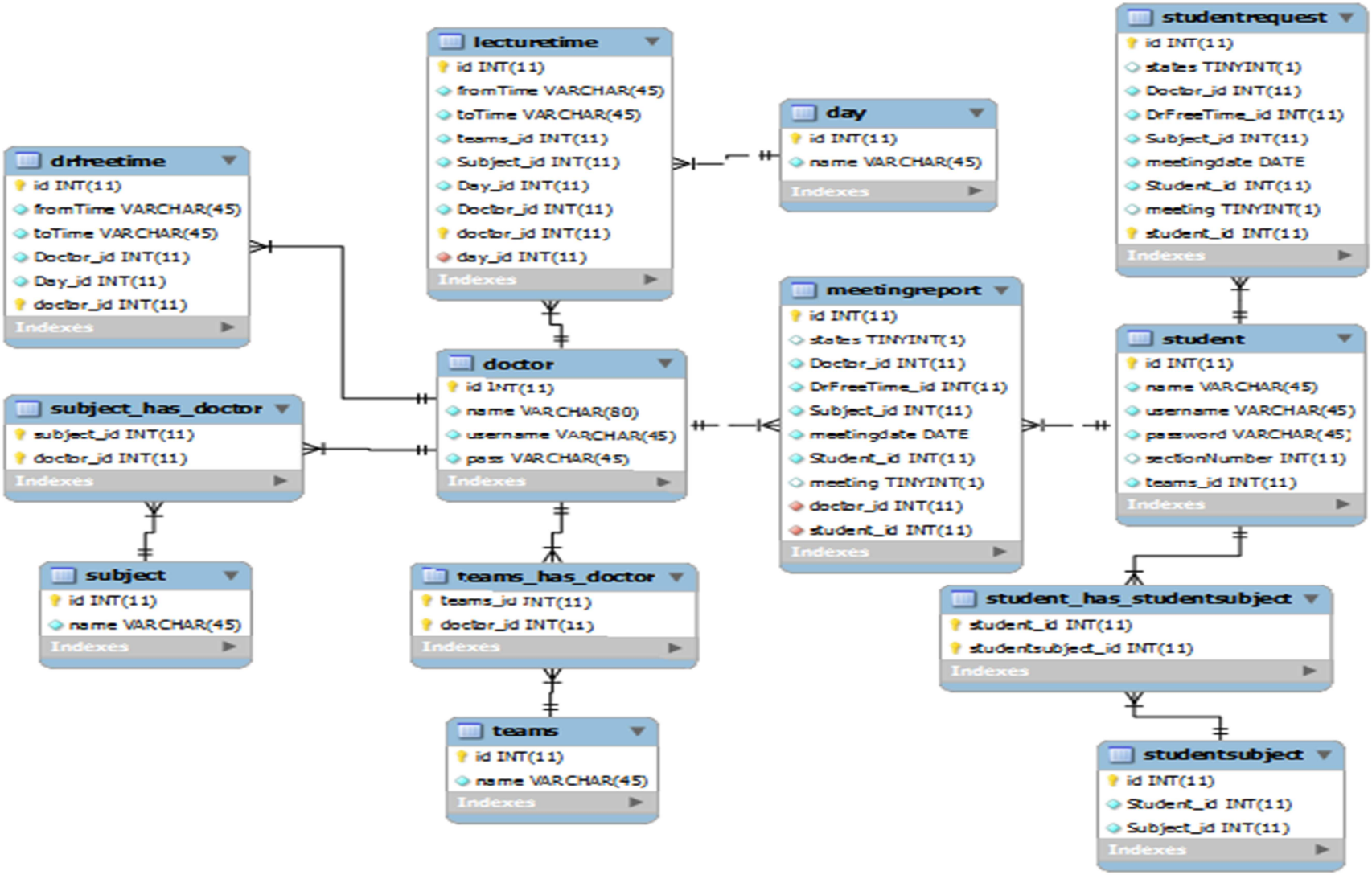


Figure 5.28: Intelligent Meeting Manager for Suspended Students Database Tables

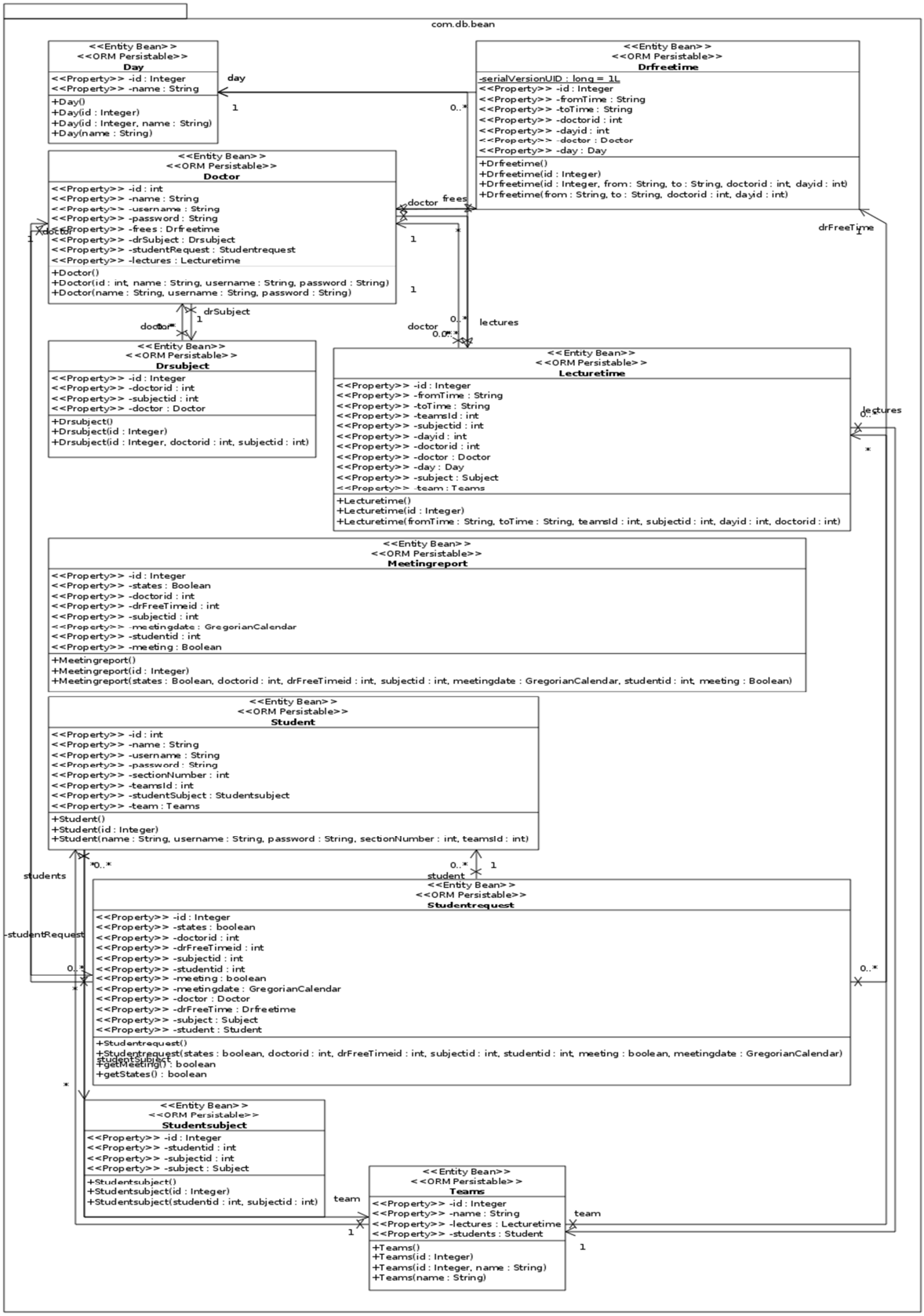


Figure 5.29: Intelligent Meeting Manager for Suspended Students Class Diagram

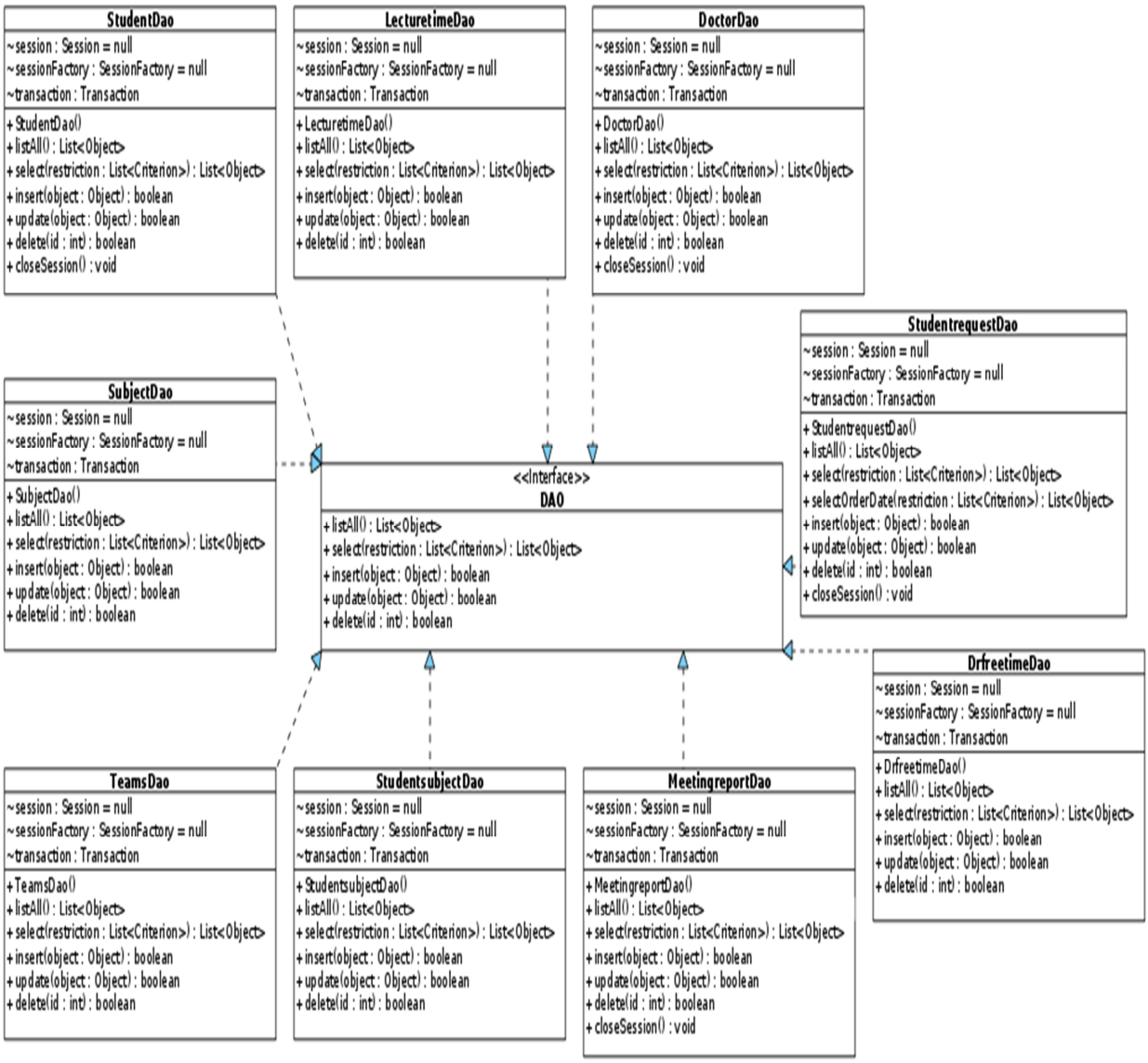


Figure 5.30: Intelligent Meeting Manager for Suspended Students Data Access Layer Class Diagram

5.8 Intelligent Document Classifier Service

Presented document classifier implements two of the Supervised Document Classification algorithms: Naive Bayes Classifier, and Term Frequency-Inverse Document Frequency (TF-IDF). Supervised learning algorithms refer to the service need to be trained first by taking the learning class and some identifying documents as an input, and then it can identify documents classes for non-classified documents. Figure 5.31 presents Intelligent Document Classifier Class Diagram.

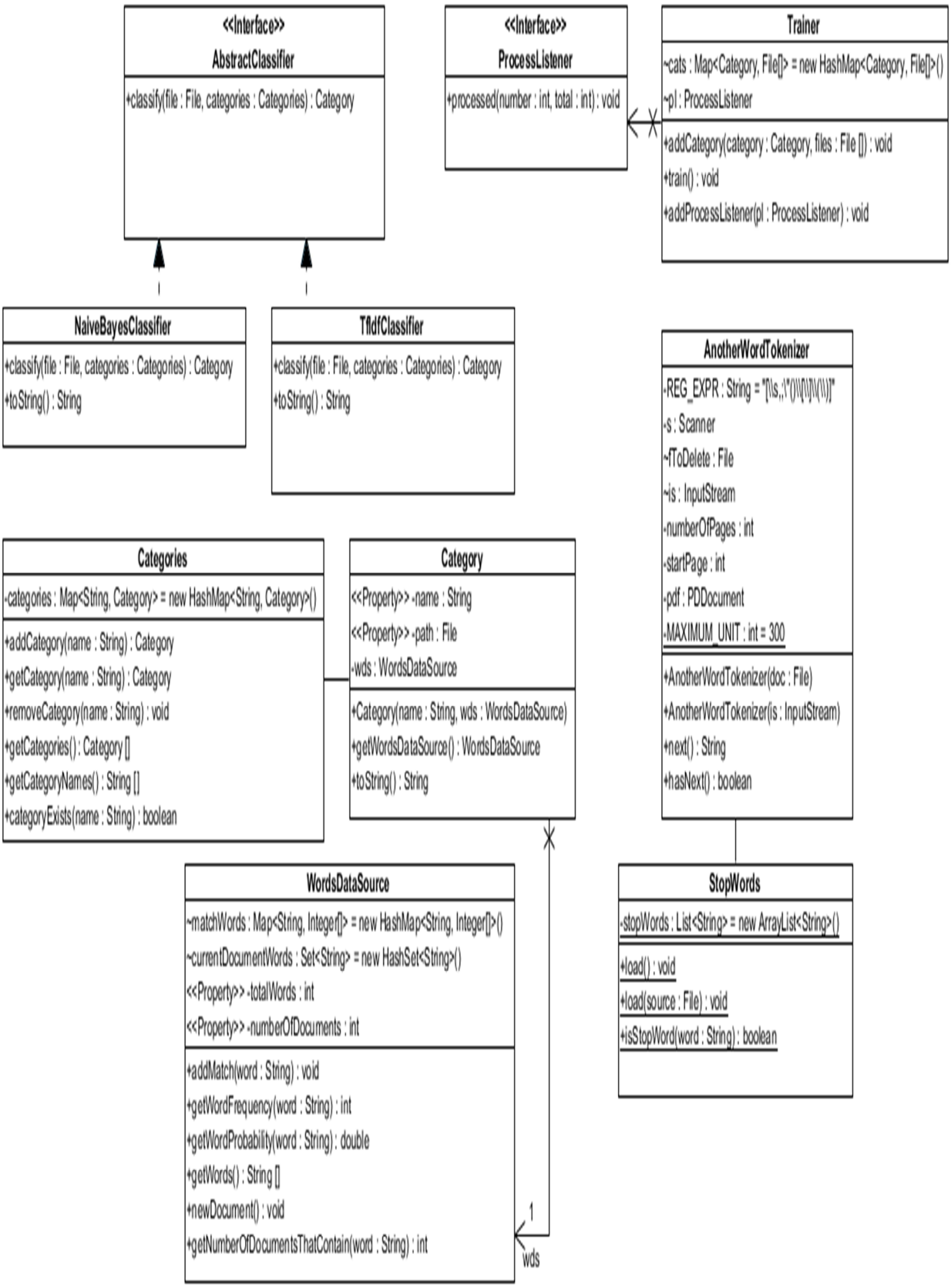


Figure 5.31: Document Classifier Class Diagram

5.9 Student e-Learning Environment and Adaptive Features

Figure 5.32 presents the students e-Learning environment presented through the desktop application. Figure 5.32 highlights different aspects presented at different parts of the application screen:

- Left Column “Top-Down”:
  - Available Topics: Displays list of available topics to browse based on the student learning profile.
  - Student Calendar: highlighting next events, and giving student to add personal meetings.
  - Student Tracker Data: presenting to the student some useful information about student’s habits and time consumed studying this module, total modules, and other tracking data.
  - Lecture PowerPoint Presentation: presents a link to the lecture PowerPoint presentation in case it is available by the instructor for the student to read and study.
- Middle Column “Top-Down”:
  - Video Display Area: Displays recorded online lectures and can be used for further recommended videos section.
  - Chapter Study Area: Displays instructor recommended LOs for this topic.
- Right Column “Top-Down”:
  - Displays recommended LOs for student based on topics keywords and students’ preferences.

Figure 5.33 displays Learn Via Question “LVQ” Main Screen. LVQ is one of the adaptive features presented through the Adaptive e-Learning Model to give students opportunity to practice final exams before attending real exams. Immediate feedback and simulation of the exam environment is presented at this phase. Figure 34 presents LVQ sample test.

5.10 Instructor Portal

Instructor Portal is the central are where instructor can handle and manage all the functionalities related to the system from courses, topics, keywords, students, registration, and all other activities. Figure 5.35 presents an instructor screen of the instructor's portal.

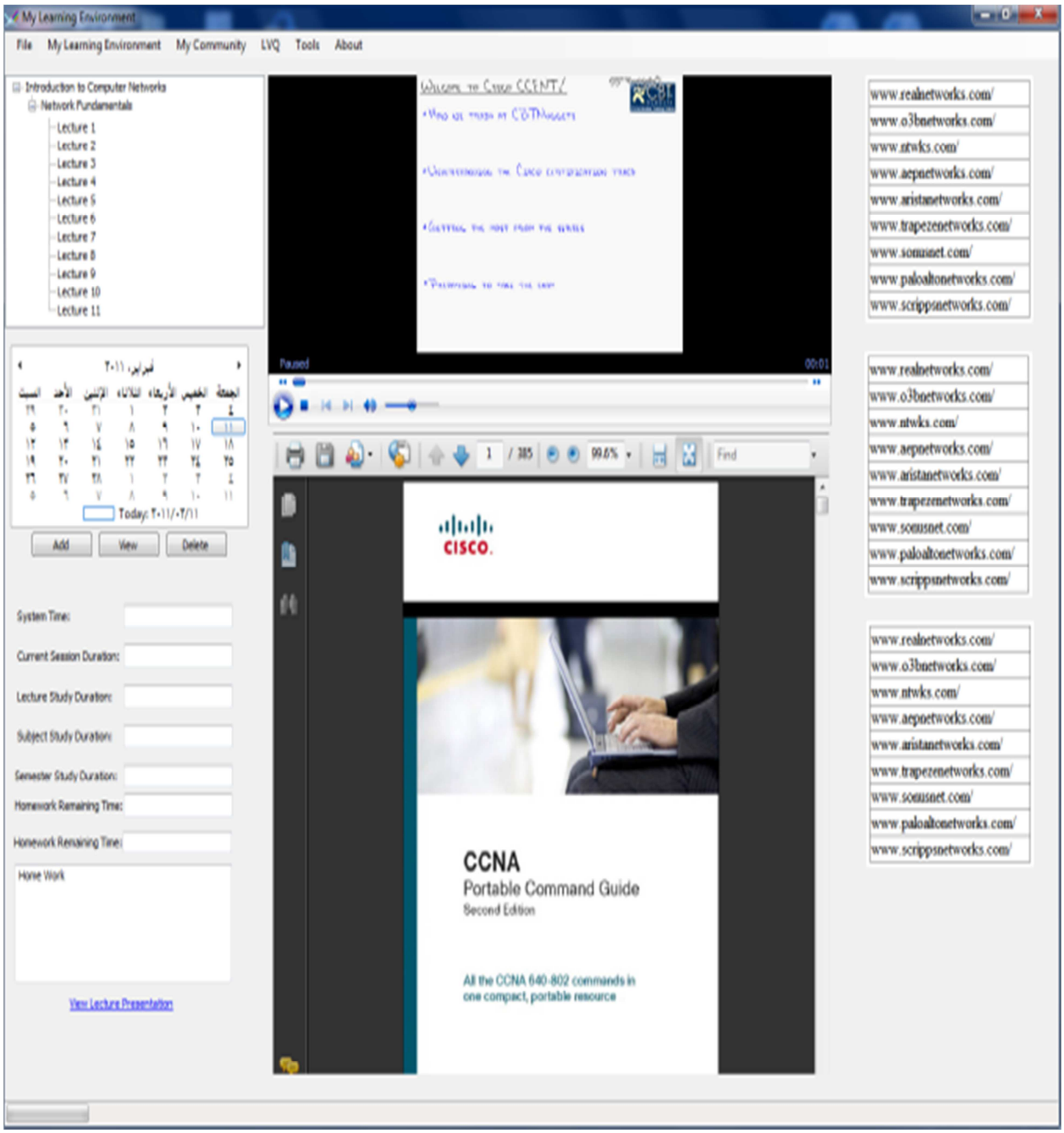


Figure 5.32: Student Learning Environment

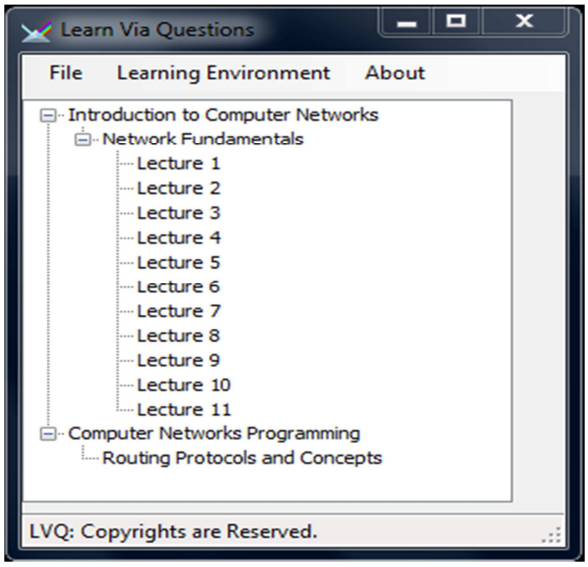


Figure 33: Learn Via Questions Main Screen

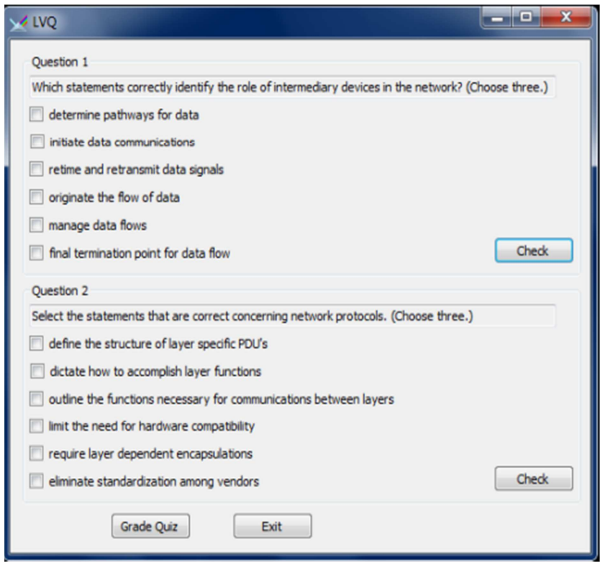


Figure 34: Learn Via Questions - Quiz Screen

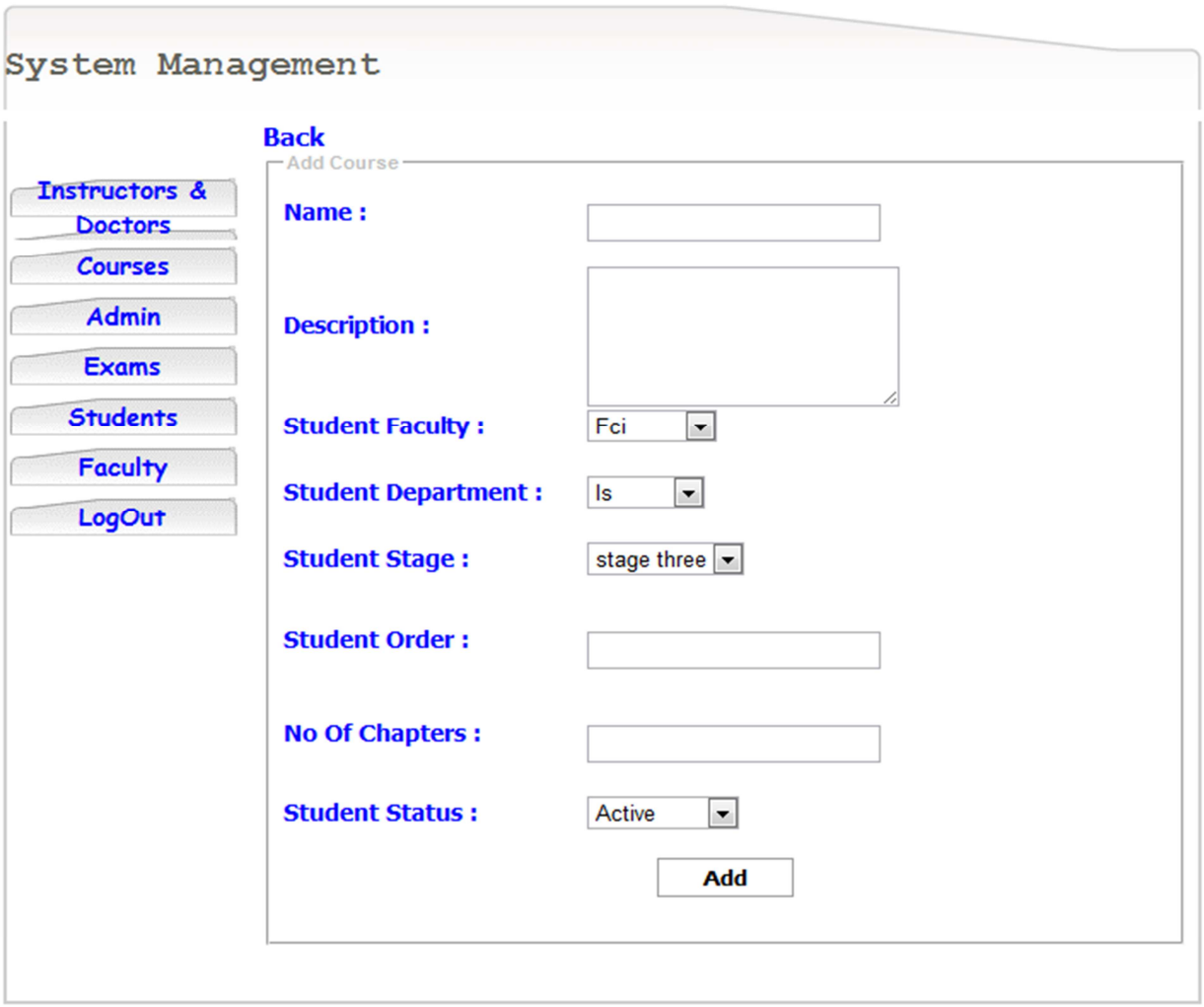


Figure 35: Instructor's Portal

5.11 Summary

This chapter presented different implementation details of the presented Adaptive e-Learning Models and Intelligent Services. Different database tables, class diagrams, and interfaces are presented. Presented details covers:

- Intelligent Student Tracking Service
- Students Manager Service
- Students Usage Data Manager
- LOs Manager Service
- Intelligent LOs Recommender: it includes:
  - Pending LOs for Recommendation Manager Module
  - Crawler Module
  - Document Processor Service: it consumes the following:
    - Tokenizer Module
    - Stopping Words Removal
    - Stemmer Module
    - Query Expansion
- Intelligent Meeting Manager for Suspended Students
- Intelligent Document Classifier

Chapter then presents some of the Student's e-Learning environment adaptive features mainly presented through the desktop application learning environment. The desktop application is not isolated from the web environment available to the student; however it is presented to overcome bandwidth and lack of internet availability challenges. Chapter then displays the instructor portal used mainly for managing the entire system.