Universiteit Leiden

Computer Science

A System for Collecting and Analyzing UML Models and Diagrams

Name: Chengcheng Li

Student-no: s1026968

Date: 13/08/2012

1st supervisor: Michel Chaudron

2nd supervisor: Bilal Karasneh

Contents

This paper is a gift for my family who have supported me all the way.

List of Figures

List of Tables

Abstract

UML is a most widely used language for building and describing models. There are 14 kinds of UML diagrams. Among them, class diagram is one of the most important in software designing. Within a class diagram, the basic elements and structure of a software system is described. For the developers, class diagrams are the guide of the process of development phase. The standard of designing UML class diagrams is rather loose. Various styles can be found from different developers. When researchers want to get research on class diagrams, they meet similar problem that there is no efficient way to collect large number of UML class diagrams to work in – analyzing, querying and so on. As most of models on the Internet are images, we focus on collecting them, providing usability of storing and querying models that are saved in images, and supporting reusability of these models.

In this thesis, a software for collecting a large number of UML class diagram images from the Internet is developed. It can download images of UML class diagrams efficiently and build a database that contains information about these images. As a part of the project to transform models in images to XMI, the paper provides a method of transforming XMI files into relational database. Then, a website is built for storing and online querying models.

Keywords: UML class diagram, web crawler, XMI, model search engine

Chapter 1. Introduction

1.1 Overview of UML

Unified Modeling Language (UML) is a standard modeling language that was created in 1997 by the Object Management Group[1]. Since then, it has been the industry standard for modeling software-intensive systems. In UML 2.2, there are 14 types of diagrams. They are divided into 3 categories, structure diagrams behavior diagrams and interaction diagrams. Here's the table of their role in the UML standard[1].

Category	Name	Role
Structure diagrams	Class diagram	Contains the attributes, methods and relationship between the classes of a system
	Component diagram	Splits a system into components and shows the dependencies among them
	Composite structure diagram	Shows the detailed structure of a class and the collaborations that this structure makes possible
	Deployment diagram	Describes the hardware deployment of a system
	Object diagram	Shows a complete or partial view of the structure of an example modeled system at a specific time
	Package diagram	Splits a system into different packages and shows the dependencies among them
	Profile diagram	Shows the classes and packages at the metamodel level together with the extensions of them.
Behavior diagrams	Activity diagram	Shows the detailed workflow of the components in a system.
	State machine diagram	Describes the states inside a system and the transitions among them
	Use case diagram	Describes the interactions between the system and its users. A use case is the target of a functionality provided by the system.
Interaction diagrams	Communication	Shows the interactions between

	diagram	different kinds of components within the system. It describes both the static structure and the dynamic behavior of the system.
	Interaction overview diagram	Describes an overview of a control flow with nodes that can contain interaction diagrams.
	Sequence diagram	Uses messages to show the communication between objects within the system as well as the lifecycles of them.
	Timing diagram	Focuses on timing constraints to explore the behavior of objects during a period of time.

Table 1. 14 kinds of UML diagrams

1.2 Overview of UML Class Diagrams

A UML class diagram represents the content of classes and the relationships between them. A class is represented by a rectangle with three parts lined vertically. On top of a class is the name of it. It is mandatory. In the middle part of the rectangle of a class, there are members and attributes. On the bottom, there are methods within the class. Here's an example of a class "Person" with four members and seven methods. The sign '-' in front of attributes or methods means that this attribute or operation is private, and the sign '+' means it is public.

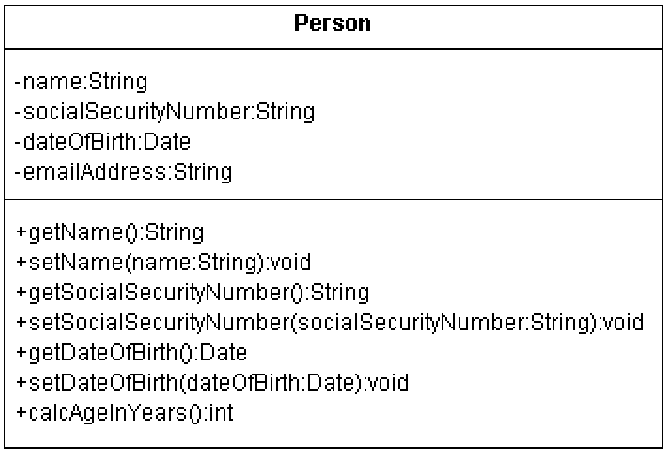

Person
-name:String
-socialSecurityNumber:String
-dateOfBirth:Date
-emailAddress:String
+getName():String
+setName(name:String):void
+getSocialSecurityNumber():String
+setSocialSecurityNumber(socialSecurityNumber:String):void
+getDateOfBirth():Date
+setDateOfBirth(dateOfBirth:Date):void
+calcAgeInYears():int

Figure 1. An example of a class diagram

There are several relationships between the classes. Here's a table of their definition and how they are represented in the UML class diagrams.

Name	Description	Representation
Generalization	It describes the relationship between	An arrow pointing to the

	particularity and generality.	parent class
Realization	It describes the relationship between an interface and a class implementing it.	An arrow with dash line pointing to the interface
Association	It's the relationship if one class has certain members or methods of another class.	A line with an arrow pointing to the class that is associated.
Aggregation	It's the relationship if one class owns another.	A line with an empty diamond pointing to the owner.
Composition	It's the relationship if one class is a part of another that cannot exist alone.	A line with a solid diamond pointing to the owner.
Dependency	If a class needs another one to implement, they have the relationship of dependency.	A dash line with an arrow pointing to the class that's relied on.

Table 2. Relationships of class diagrams

UML class diagrams are very important. They can show the static structure of a system. They have provided the basic elements for other diagrams. They are especially useful for the developers. They are mainly used in the purpose of analysis and design. During the analysis phase, one should not take the technology into account. Abstracting the system into different modules is an efficient way. The UML class diagram is a way to represent the abstraction. When a project goes into the phase of design, the UML class diagram used in analysis phase may be changed to suit for the development environment. They become the basis of the code for the project. Different people have different styles of codes. The most efficient way for them to communicate is by the UML class diagrams.

1.3 Overview of Our Project

Our project aims for building a system that can establish databases of images of UML class diagrams from the Internet, transfer the images to XMI models and make queries within the models. It includes the image collecting part, transforming part and querying part.
Image is different from text. It’s very difficult to abstract information from images. Current researches of model search engine focus on dealing with the models stored in text files. Our project supports the usability of collection, conversion, storage and query for models from images. As it’s expensive for developers to find software assets, which results to lack of knowledge about reusable assets, this is one of main reasons why developers are not willing to reuse software[2][3][4][5]. For this, finding reusable assets is a fundamental aspect of an effective reuse program[30], so we try to find models and support reusability of these models via converting them into suitable format that is easy to be modified.
In this paper, the image collecting part and the XMI querying part is included. For the image collecting part, we have developed a program *ImageCrawler* that behaves as a web crawler to collect images of UML class diagrams from the Internet. Then, a website is established for building a database of images we have collected. For the XMI querying part, we implement functions by mapping XMI models into relational databases and make queries within them.

Chapter 2. *ImageCrawler*

2.1 Overview of Web Crawler

We all know that it's convenient for us to "Google" something. Google, as the biggest search engine in the world, has information of billions of websites and keeps them up to date. How can that be done?

Search engine is the technology which started in 1995. As more and more information has been put online, it becomes more and more important. Search engine collects information from the Internet, classifies them and provides the search function of the information for the users. It has become one of the most important Internet services.

The web crawler is a program that can fetch information from the Internet automatically. It plays an important part in search engine because it can download data from the Internet for the search engine.[6][7]

The basic operation of the web crawler is to fetch information from the website. The process of fetching data is the same as we surf online. The web crawler mostly deals with URL. It gets the content of the files according to the URL and classifies it. Thus, it's very important for the web crawler to understand what a URL stands for.

When we want to read a website, the web browser, as the client end, sends a request to the server end, gets the source of the page, explains it and shows it to us. The address of a webpage is called uniform resource locater(URL) which is a subset of universal resource identifier(URI). URI is the identifier for every resource on the website: html documentation, images, videos, programs and so on. URL is the string to make the description of the information online. It contains three parts: the protocol, the IP address of the server where the resource is and the path of the resource at the server end.

When we put the web crawler into practice, we use it to traverse the entire Internet and get the related information, which is just like "crawling". During this procedure, it regards the Internet as a huge graph, and every page is a node inside with the links in a page as directed edges. Thus, the crawler will take either depth first search(DFS) or breadth first search(BFS). However, in DFS, the crawler might go "too deep" or get trapped. So BFS are mostly chosen.

In the process of BFS, a node is chosen as the starting node. A queue is needed as the data structure to save the nodes. The algorithm is:

(1) Starting node V is in the queue.

(2) If the queue is not empty, continue.

(3) Node V is signaled as visited.

(4) Get the nodes that adjacent to node V and push them into the queue. Then go to step (2).

Here is an example:

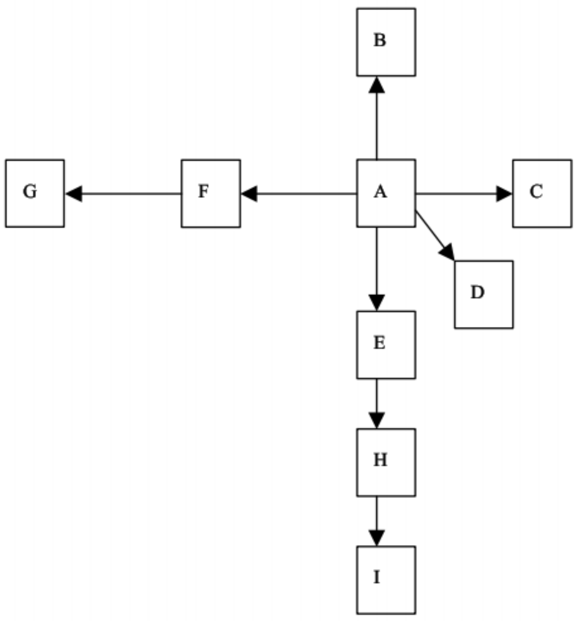


Figure 2. Process of BFS

In this figure, we use node A as the starting point, and the process will be:

Operation	Nodes in the queue
A in	A
A out	empty
BCDEF in	BCDEF
B out	CDEF
C out	DEF
D out	EF
E out	F
H in	FH
F out	H
G in	HG
I in	GI
G out	I
I out	empty, and end

Table 3. Process of operation in a queue

So the order that nodes are visited will be: A, B, C, D, E, F, H, G, I.
The web crawler starts from a URL as the starting node. If there is a link that leads to a file, like a PDF file for the user to download, it's a termination because the crawler cannot get links from it. The process is to start from the initial URL, put links inside it into the queue. The links that's visited will be signaled "visited". If a link is already visited, it will be ignored next time it's found. The process can be shown by this picture:

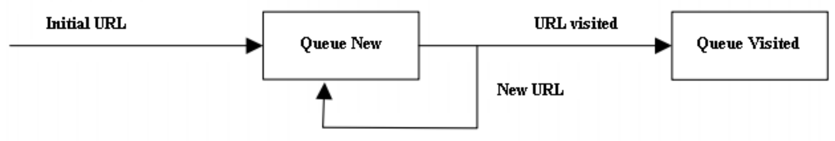


Figure 3. Process of URL queuing

(1) Put the initial URL into the Queue New.
(2) Get URL A from Queue New, and get all links it has.
(3) If the links exist in the Queue Visited, ignore. Otherwise, put them into Queue New.
(4) Put the URL A into the Queue Visited. Go to step (2).
BFS is the most widely used method by the web crawlers. The main reason is that when we are designing a webpage, we usually put links to the pages that have contents related to the current page. Using BFS can find them by first time.

A web crawler should work continuously. Good stability and reasonable usage of resource is required. Web crawler is a vulnerable part of a search engine. It deals with web servers that cannot be under control of the search engine system. A bug from one webpage may put the web crawler into severe situations like halt, crash or other unpredictable problems. Thus, the design of a web crawler should take different conditions of the network into account.

2.2 Related Work

Other than the large search engines like Google and Yahoo, it is difficult to get a large number of UML class diagrams in a short time. Even if people use the search engines, no downloading service is available to save them to the local disk.
There are several tools that are specially designed for downloading images from the Internet. However, they have drawbacks. Some of the tools cannot meet the requirement of downloading images with a large number and full sized. Most of them are commercial.

- Google has provided an API for downloading images from the result of image search.[8] However, the API has a limitation of getting a maximum of 64 results on 8 pages. There are tools based on this API, and they can only get at most 64 images. It's far from enough.
- Firefox has a plug-in named "Save images" that can save images from the current tab page the user has opened. The problem is that the result of Google image search shown on the webpage is just thumbnails. When we use the add-on in Firefox, all we have got are just thumbnails from this one page. The images downloaded are too small to use. What's more, the tool can only download images from pages that have opened in the browser. It's not possible to open all the pages of the list of images and download them one page by one page.
- Bulk Image Downloader is a software that can download full sized images from almost any thumb nailed web gallery. However, it takes $24.95 to buy it. Otherwise, functions will be incomplete.

Thus, we have developed our own software named *ImageCrawler* to collect images from the Internet.

2.3 Software Requirements

The requirements of our software have two points:
1. We want to get as many images as possible.
2. We want to finish the process efficiently.

To finish the task, we have to implement a web crawler for images all over the Internet. A web crawler with high performance should have two features:
1) It should be able to grab great capacity of data from the Internet. That's why we use it.
2) It should run on a distributed system. As the quantity of data is extremely large in the Internet, an integrated system cannot meet the requirement. The crawlers should work on a distributed system parallel to improve the efficiency.

However, as the task should be finished in a normal PC, the efficiency of a normal web crawler is not good. Different from a normal web crawler that gets every piece of information from the Internet, the crawler we want to implement is just to download images of UML class diagrams from the Internet to establish the database. Thus, we can use the result of an image searching website as our starting node.

2.4 Why Google?

Google, as one of the most widely used search engines in the world, can meet our requirements.
It can get a large number of images focused on a special topic. When we go to the image search page of Google, we can get the list of images from various websites. It's the assembly of images related to the key words we have input. The time for Google image search is extremely fast because the information of billions of images has already been saved in the server of Google. Thus, if our web crawler can use the result of Google image search as a starting node, the efficiency is great.
In October 2009, Google has released the function to search by images.[9] When an image is uploaded, the algorithm is applied on it to abstract the features from it. The features can be textures, colors, and shapes. Then, the features are sent to the backend of Google and compared with the images on the server. If there is an image that has similar features with the original picture, the algorithm takes it as a confident match. Then, Google will list the confident matches on the website for the user.
The advantage of this algorithm is that it simulates the process of human being watching an image. If the image for query has a unique appearance, the result will be very good. The result for unique landmarks like the Eiffel Tower is fantastic.
However, when we are using the function to search for UML class diagram, the result is not so efficient. The reason is that UML class diagram are mainly combined by shapes like rectangles, arrows and text. There are no unique symbols. When we upload a UML class diagram to the Google search by image, the features Google will extract are just normal ones. Thus, the result is not only UML class diagram but also graphs with curves and rectangles that relate to math research, stock market, and medical research and so on.
Thus, the function we want to use is the traditional way to search images by key words. At least for UML class diagrams, the accuracy of searching by image does not take advantage of searching

by key words.
In the meantime, Google provides details of personal result, which collects images related to your most websites that the user has visited or is interested in, which is helpful for collecting models the user wants.[10]
Other search engines also have similar functions of image search, but Google provides a better user interface and more powerful filter for the images. The most important reason we choose Google is that the URL of Google image search can be easily constructed with different parameters. We can use Google image search in our program without going to the web browser first.

2.5 Implementation of *ImageCrawler*

The program of *ImageCrawler* does not rely on the API of Google image search. It is the program that can download as many images as possible from the result of Google image search. The key words can be set by the user as well as the number of images that he wants to get. It simulates the process of downloading images from Google manually. It implements the theory of web crawler, gets the data from Google image search and downloads them automatically. After that, the information of the images will be saved in the database. At the same time, the program has the function to build a “blacklist” of URLs of images that are not UML class diagrams. There is another function that gives a rank of domains of images to see which websites contribute how many images.

2.5.1 Database Design

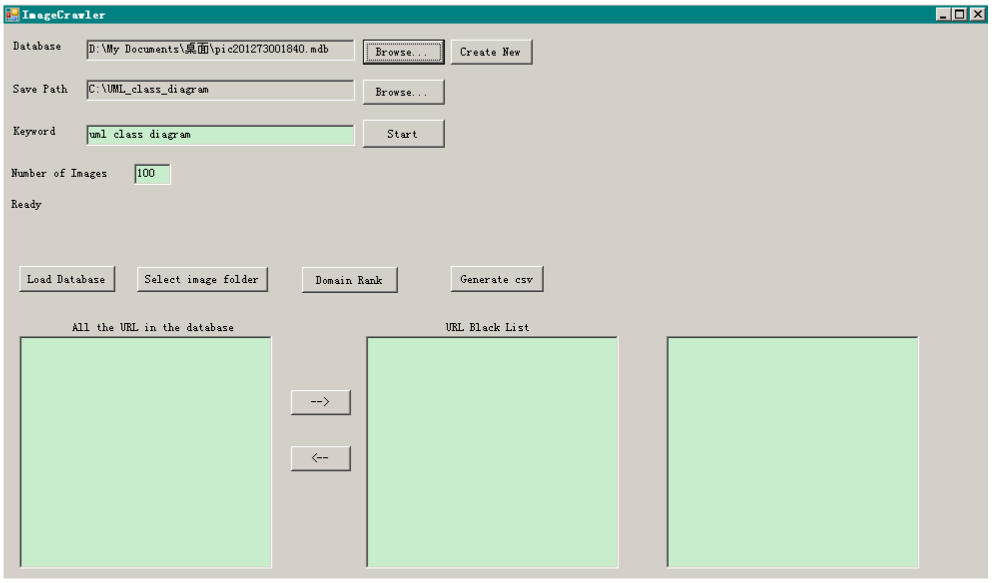
The database is designed for keeping the information of images we have downloaded. Most of the information of the images is saved in the table "pic". As URL is unique in the Internet, we use the URL as the primary key. The width and height, as the basic attributes of an image, is also saved into the database. What's more, there is the record of the pixel format of an image in the database. It's useful for the following steps of digital image processing. Some methods can only apply to specific kinds of images. For example, the *ExhaustiveTemplateMatching* class, which implements exhaustive template matching algorithm and can be used to detect shapes in the UML class images, can only be applied to grayscale 8 bpp and color 24 bpp images. Another key we have added to the database is "isUML". It is a Boolean value that shows whether the image is an UML class diagram. Although the key words we have input are "uml class diagram", no search engine can ensure that all the images that found are strictly "uml class diagram". So there should be a key that shows which images are UML class diagram that can be used in other operations.
As not all images downloaded are UML class diagrams, there is a list needed that can save the URLs of such images to save the trouble to download them again next time. Thus, a “blacklist” is added into the database, together with a “whitelist”. The table "picwhite" and "picblack" will store the urls of the images that are/are not UML class diagrams.
Not only URLs of the images but also the domains are interested by us. We want to find out which websites can provide UML class diagrams more than others. There are two tables named "blackcount" and "whitecount" that store the domains where the images of the black list or white

list come from and how many images have been downloaded from each domain. Here's the table of the design of this database.

Table: pic, Contains all the information of images		
Key	**Type**	**Primary Key**
ID	Auto-increment	
Url	Text	Yes
Width	Text	
Height	Text	
Pixelform	Text	
Fname	Text	
Comments	Text	
isUML	Boolean	
Table: picblack, Contains the black list		
Key	**Type**	**Primary Key**
ID	Auto-increment	
Url	Text	Yes
Table: picwhite, Contains the white list		
Key	**Type**	**Primary Key**
ID	Auto-increment	
Url	Text	Yes
Table: blackcount, Contains the domain statistics of the black list		
Key	**Type**	**Primary Key**
ID	Auto-increment	
Domain	Text	Yes
Count	Text	
Table: whitecount, Contains the domain statistics of the white list		
Key	**Type**	**Primary Key**
ID	Auto-increment	
Domain	Text	Yes
Count	Text	

Table 4. Design of the database

The interface of the program is as follows.



Database D:\My Documents\桌面\pic201273001840.mdb
Save Path C:\UML_class_diagram
Keyword uml class diagram
Number of Images 100
Ready

Figure 4. Interface of ImageCrawler

Here's the class diagram:

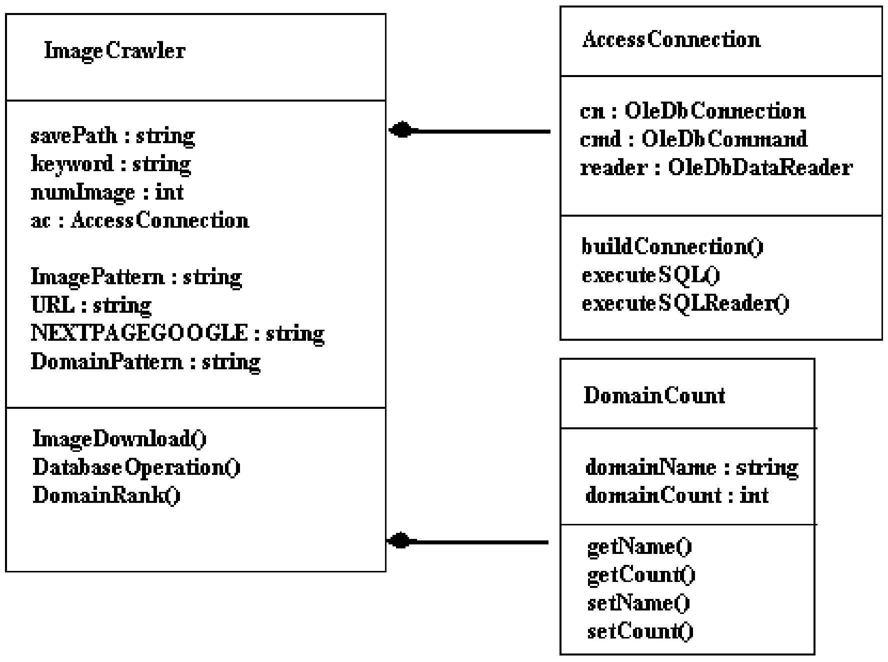


Figure 5. Class diagram of ImageCrawler

2.5.2 Image Downloading Part

2.5.2.1 Initialization

The program takes the key word the user has input and constructed a URL for searching images from Google. The URL for downloading images from Google is constructed by two parameters: key word and search category (image search). For example, the URL for downloading images that relates to "UML" is: http://www.google.com/search?q=uml&tbm=isch".

At the same time, the program takes the number of images that the user has input and saves it into a variable.

Before the process starts, the program requires the user to select a database to save the information of the images to be downloaded. There is a function *checkDB()* to check whether the database the user has selected meets the format of the database we have designed. In this function, the program gets the names of all the tables inside the database and save them into *DataTable* objects. For the keys inside the tables, there is a function *getCols()* that also uses the *DataTable* objects. The function will put the names of the columns in a table and compare them with the database we have designed. If all the tables and columns exist, the database is ready for the process.

If the user has selected a database, we suppose that there is already information about many images saved in the database. To avoid downloading images that are already in the database, a list of URL will be established. When the URL of an image is found in the next steps, the program will check whether the image has already been in the database. If so, the downloading part will be ignored.

2.5.2.2 URL Analysis

When we have constructed the URL, we should download the data of the page. In .Net, the *WebClient* class is used. *WebClient* class provides the method to send and receive data from a resource that is signaled by a URI. The *DownloadData* method is to download the page data from the URI provided. So we take the URL we have constructed as the parameter of the *DownloadData* method and get the source of the page. Within the page data, there are links to kinds of resources. This is like the starting node in the theory of web crawler.

So how do we get the links from the source code of the page? One of the most efficient ways is by regular expression. Regular expression is a string which can be used for parsing and manipulating text. In complex search-and-replace operations, and validating whether a piece of text is well-formed, regular expression takes great advantage than other methods. For the image crawler, we have two kinds of strings to match. One is the URL that represents a link to another page. It's like the "adjacency node". The other is the URL that directly links to the resource of images. It's the end of a process of the crawler's work.

There comes a problem. When we have got the data of the page, not all links are what we need. We need only two types of URL as discussed above. Thus, we have to give two regular expressions that correspond to the two types of URL to the crawler for matching in the page data.

The first is the URL which links to another page that has the result of image search. First, we should analyze the structure of Google's page of image search. In the web browser, the list of images is shown in standard version. It dynamically shows pictures in other pages as we pull the scroll down. However, the page data the program can get is of the basic version. If we want to go to another page, we have to click on the number that has the link.

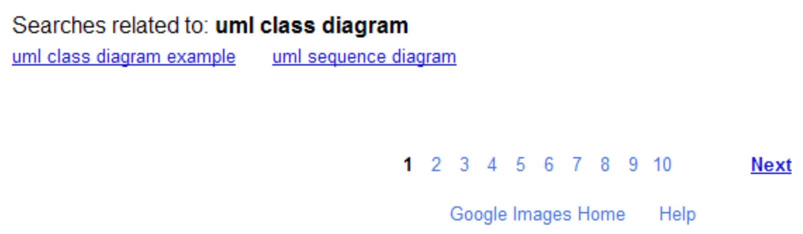

Searches related to: **uml class diagram**
uml class diagram example
uml sequence diagram
1 2 3 4 5 6 7 8 9 10
Next
Google Images Home
Help

Figure 6. Links of pages of Google image search

Thus, what we want to get are the links of numbers represented. They start with "search?q=" and the key words. There is the parameter "tbm=isch" which means the search is for images. Another important parameter is "start=" followed by a number. After the searching process, there are thousands of images. The number after the "start=" parameter is to show which is the first image that the URL links to. For example, the URL of page 2 contains "start=20". The number of images starts with 0, which means the second page shows the images from 21st to 40th. There are other parameters such as "hl=en" which means the language of this page is in English. They are not important for this process, so we ignore them. Thus, the regular expression of links that point to other pages of list of images is:

"/search\\u003Fq=[a-zA-Z0-9;&-=]*start=[0-9][a-zA-Z0-9;&-=]*".

When we have found a link, just add "http://www.google.com" in front of them and the link is fully constructed. Then, the link is stored in a list that has all the links found.

The second regular expression we have to construct is the URL for images. This is relatively an easier one. The URL for images should start with the "http(s)" and end with an extension of image file. So the regular expression is:

"((http(s)?://)?)+(((/?)+[\w-.]+(/))*)+[\w-./]+\.+(jpg|jpeg|png|ico|bmp|gif)".

When we have found a link to an image, we first check whether it is in the database, which means it has been downloaded before. If not, we save it to the list that contains all the URL of images found. Then there is the process to download it.

2.5.2.3 Image Download

After we have got the URL for an image, we can download it using *HttpWebRequest* to get the data stream of this image. Then, save it as an instance of the *System.Drawing.Image* object. The next step is to save the information of the image into our database. We use the *Oledb* namespace to implement it. The URL, filename, width, height and pixel format of the images will be saved into the database.

2.5.3 Domain Rank Part

As not all images downloaded are UML class diagrams, there should be the function to distinguish them. The program can show the list of all URLs in the database and show the corresponding image of selected URL. The user can put them into the “blacklist” or delete them from “blacklist” manually. The “whitelist” contains the URLs of images that are UML class diagrams.

Based on the “blacklist” and the “whitelist”, the program can abstract the domains where the images come from. Then, the program can give the statistical data of the domains and make a rank of them.

Our database is built as an access file. As the database will be used in the website, and *csv* file is more popular in the web applications, there is a way to generate csv files from our database. The user can click on the “Generate csv” button, and five *csv* files corresponding to the five tables within the database will be created.

2.6 Validation

With the *ImageCrawler* we have developed, a database with 2341 images has been established. The key words we have used are: "uml class diagram", "uml class diagram example" and "uml class model". The result of images differs with different key words. The whole process has cost an hour. The time cost depends not only on the algorithm, but also on the Internet condition.

We have manually established the “blacklist” using the function in the "Domain Rank Part" of *ImageCrawler*. Images that are not UML class diagrams, or UML class diagrams but too blur to distinguish or a screen shot that contains only a part of a UML class diagram are put into the “blacklist”. As a result, 947 of the 2341 images have been put into the “blacklist” with 1394 UML class diagrams.

Compared to other tools that collect images from the Internet, *ImageCrawler* has its advantage.

1. The tool that uses Google API to download images from Internet has a limitation of 64 images.[11] *ImageCrawler* does not have this limitation so that it’s easy for the users to collect a large number of images.
2. The plug-in of Firefox named “Save images” has two drawbacks. It can only download the images appear in the current tab opened in the Internet browser. The tool will just download images as they are represented in the page. When it’s used to download the images listed as thumbnails in the result page of Google image search, it cannot download the images with their original sizes which are really what the users need.
3. *ImageCrawler* is free for the users. It’s developed by .NET framework. Thus, it has a good expandability. In future development, it can be transformed to an online application instead of Windows application to provide more flexible usability.

The result of the statistics of domain name has been saved in the database. In the “whitelist”, there are 715 different domains involved in. Among them, http://upload.wikimedia.org ranks at the top because 30 UML class diagrams come from the domain. Following it, there are websites like http://www.ibm.com with 29 UML class diagrams downloaded, http://ars.els-cdn.com with 25 and http://docs.nullpobug.com with 21. 12 domains contain 10 to 17 UML class diagrams for each. 58 domains contain 4 to 9 UML class diagrams. 641 domains have less than 3 UML class diagrams downloaded. In the “blacklist”, 417 different domains are involved in. http://img.docstoccdn.com, with 53 images that are considered not UML class diagrams, ranks at the top of the “blacklist”. There are 13 domains that contain over 10 images that are not UML class diagrams. 133 domains contain 2 to 9 images that are not UML class diagrams. The rest 270 domains have only one image that is not UML class diagram for each.

2.6.1 Source Code with UML

Another work we want to fulfill is to find out whether there are source codes provided together with the UML class diagrams. We have searched on the domains on the “whitelist” that have provided more than 3 UML class diagrams. Among the 29 websites, 11 of them have source code provided. They can be classified into 4 categories.

1. UML education/standard. http://www.xml.com(6 images provided), http://www.uml.org.cn(15 images provided) and http://www.c-jump.com(6 images provided) belong to this category. The source codes on these websites are provided for the purpose of education to explain how UML class diagrams are used.
2. Software reference. http://mdp-toolkit.sourceforge.net (10 images provided) and http://gdal.org (6 images provided) belong to this category. Both the websites are reference to certain libraries or software toolkit. The source code provided are examples to explain how certain functions are used.
3. Software design. http://www.felix-colibri.com (14 images provided), http://staff.aist.go.jp (4 images provided), http://www.codeproject.com (17 images provided), http://www.c-sharpcorner.com (3 images provided) and http://iw.mapandroute.de (6 images provided) belong to this category. The source codes provided by these websites are the implementation for the software.
4. Blog of IT articles. http://www.blogjava.net (4 images provided) is the only one in this category. It's the blog of articles related to the IT industry. This category may contain one of the three classes above. The source code provided may be the implementation of certain software or example codes attached to some educational content of UML standard.

2.7 The Limitation of *ImageCrawler*

The advantage of *ImageCrawler* is based on Google image search, so is the weakness. The reason is that the pages of the result of an image search are at most 50. The average number of images within one page is 20. Thus, the maximum number of images the program can get with one piece of key word is 1000. With the URL of some images are out of date, the actual result may be less. So to solve this, we use several key words for search to build our database.

Another weak point is accuracy. As the searching process is motivated by key words, the images found are those with descriptions that include the key words. Thus, not all images found are really UML class diagrams. Some of them maybe the screenshot of a presentation named "UML class diagram" or a photo of a book that relates to "UML class diagram".

2.8 Focused Web Crawler

A way to improve the efficiency of a web crawler is the focused web crawler. Different from normal web crawlers, a focused web crawler does not rely on a huge coverage of websites. It mainly focuses on the websites that belong to a certain topic. Thus, focused web crawler is better performed for fetching data for the users with a specific topic[12][13][14][15][16].

There are three aspects that should be concerned in designing a focused web crawler.
1. The definition of the topic which the crawler is going for.
2. The analysis and filter of the content of data fetched by the crawler.
3. The method to search through the Internet.
In this case, the topic is the UML class diagram.
As discussed above, normal web crawler often uses breadth first search to explore the Internet. The implementation of BFS is relatively simple. In general, websites of the same topic tend to gather together. Thus, BFS can be used in a focused web crawler. Given the start node, the content of nodes adjacent to it is considered belonging to the same area. The drawback of BFS in implementing focused web crawler is that as the coverage becomes bigger, more irrelevant pages will be fetched and downloaded. What's more, when the number of irrelevant pages is much larger than that of relevant ones, the web crawler may be trapped in a vicious cycle.
There is another way to avoid this problem. When the links within a page are detected, the web crawler analyses them before downloading the data. The analysis is to find which links are most relevant to the topic and ignore the others. The efficiency is better than BFS. However, the prediction cannot be 100% accurate. Some relevant pages may be thrown away.
There are some methods to analyze the quality of a webpage. In general, within a website, the links to the most important pages will be provided on its homepage or other places that are easy for the user to find. If a page has been linked to by many websites, it can be considered the one with high importance. PageRank is a technique that is based on this idea. On the other hand, if the number of links to certain page is too large, it may not be a valuable one because there is possibility that it's just a "junk page" that appears many times just to improve the PageRank value.
Focused web crawler can be embedded to our *ImageCrawler* program. However, the time of webpage analyzing may slow down the efficiency.

2.9 Perceptual Hash Algorithm

As for the accuracy problem that not all images downloaded are really UML class diagrams, a way to solve it is to use the "Perceptual Hash Algorithm"[17][18][19][20]. The algorithm can generate the "fingerprint" for images and compare them. The more similar the "fingerprints" of two images are, the more similar the two images are.
The algorithm consists of several steps.
1) To zoom an image into a small size. For example, zooming an image to a small square that the width and height are both 8 pixels. So the small image is combined by 64 pixels. Then, the details of the image are eliminated. Only the structure and level of brightness are preserved. Thus, the difference of images with different sizes is eliminated.
2) Make the image a greyscale one.
3) Calculate the average of gray value of all the 64 pixels.
4) Compare the gray value of each pixel with the average one. If larger than or equal to it, the corresponding pixel is marked 1. Otherwise, mark it 0.
5) Now we have a table with 64 bit of 1 or 0. This is the table that represents the image. When we want to know how similar two images are, just compare the tables of them to see how many bits are the same.

2.9.1 Implementation of Filter4ImageCrawler

Filter4ImageCrawler is the program that uses perceptual hash algorithm as a filter for the images downloaded by *ImageCrawler*. It loads the database that's built by the *ImageCrawler* program, locates the folder where the images downloaded are, and takes an image as the example for the filtering process. As the result, the database loaded will be changed. There is a key in the table "pic" in the database named "isUML". It is a Boolean value that shows whether the image is a UML class diagram. The value of this column will be changed according to the result of the program. If an image is similar enough to the example image, its corresponding "isUML" attribute will be true. The threshold of similarity is set to 25. It means that after the process of perceptual hash algorithm, if there are less than 25 pixels different between the testing image and the example image, we can regard them to be similar.
Here's the class diagram of the program.

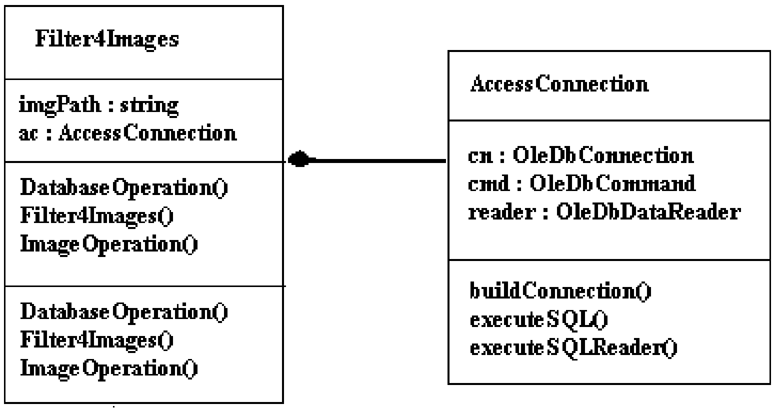
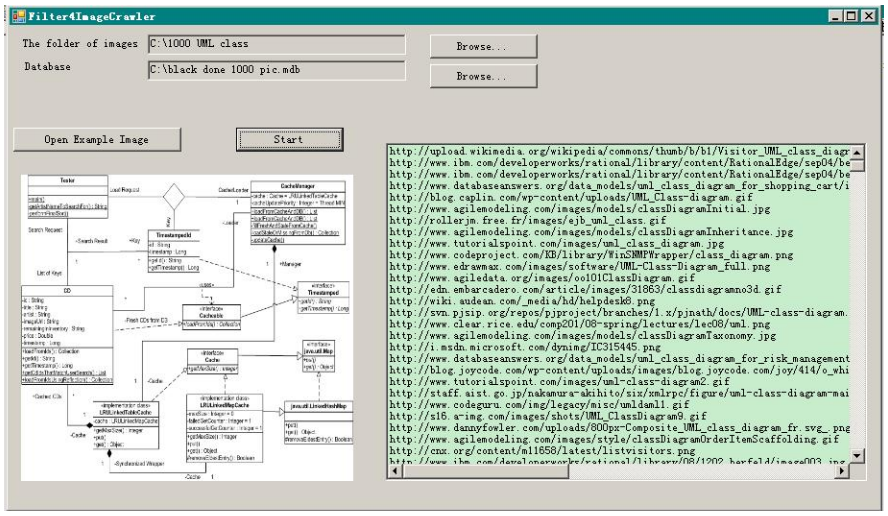


Figure 7. Class diagram for Filter4ImageCrawler

Here's the screen shot of this program.



The folder of images: C:\1000 UML class
Database: C:\black done 1000 pic.mdb

```
http://upload.wikimedia.org/wikipedia/commons/thumb/b/b1/Visitor_UML_class_diagr
http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/be
http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/be
http://www.databaseanswers.org/data_models/uml_class_diagram_for_shopping_cart/i
http://blog.caplin.com/wp-content/uploads/UML_Class-diagram.gif
http://www.agilemodeling.com/images/models/classDiagramInitial.jpg
http://rollerjm.free.fr/images/ejb_uml_class.gif
http://www.agilemodeling.com/images/models/classDiagramInheritance.jpg
http://www.tutorialspoint.com/images/uml_class_diagram.jpg
http://www.codeproject.com/KB/library/WinSNMPWrapper/class_diagram.png
http://www.edrawmax.com/images/software/UML-Class-Diagram_full.png
http://www.agiledata.org/images/oo101ClassDiagram.gif
http://edn.embarcadero.com/article/images/31863/classdiagramno3d.gif
http://wiki.audean.com/_media/hd/helpdesk8.png
http://svn.pjsip.org/repos/pjproject/branches/1.x/pjnath/docs/UML-class-diagram.
http://www.clear.rice.edu/comp201/08-spring/lectures/lec08/uml.png
http://www.agilemodeling.com/images/models/classDiagramTaxonomy.jpg
http://i.msdn.microsoft.com/dynimg/IC315445.png
http://www.databaseanswers.org/data_models/uml_class_diagram_for_risk_management
http://blog.joycode.com/wp-content/uploads/images/blog.joycode.com/joy/414/o_whi
http://www.tutorialspoint.com/images/uml-class-diagram2.gif
http://staff.aist.go.jp/nakamura-akihito/six/xmlrpc/figure/uml-class-diagram-mai
http://www.codeguru.com/img/legacy/misc/umldaml1.gif
http://s16.a-img.com/images/shots/UML_ClassDiagram9.gif
http://www.dannyfowler.com/uploads/800px-Composite_UML_class_diagram_fr.svg_.png
http://www.agilemodeling.com/images/style/classDiagramOrderItemScaffolding.gif
http://cnx.org/content/m11658/latest/listvisitors.png
http://www.ibm.com/developerworks/rational/library/08/1202_berfeld/image003.jpg
```

Figure 8. Interface for Filter4ImageCrawler

2.9.2 Validation of Filter4ImageCrawler

To validate the correctness of recognizing UML class diagrams of the program, several experiments have been implemented. Before the experiment, a database of 1000 UML class diagrams has been established. The attribute of "isUML" of the database is manually filled. So the database is a standard version of the list to show whether each image downloaded is actually a UML class diagram. In the experiment, the program will take an image of UML class diagram as the example image, abstract its structure using the perceptual hash algorithm and compare the structure with the structures of the images we have collected. Then, a Boolean value of "isUML" is given for every image in the database. After the program is finished, we compare the database it has changed with the standard list to validate the efficiency of the program. As an image is needed as an example for the process of the filter, we take 6 images to perform 6 experiments. They are divided into two groups: “colored” and “black and white”. In each group, three subgroups has been made by the content of the images: simple, normal and complicated.

Experiment 1. Black and white simple UML class diagram.

Example image:

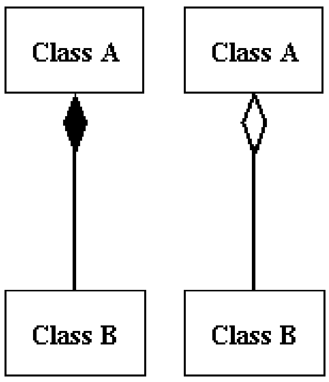


Figure 9. Example of black and white simple UML class diagram

Result: 47.06% of the result is the same as in the standard list.
Experiment 2. Black and white normal UML class diagram.

Example image:

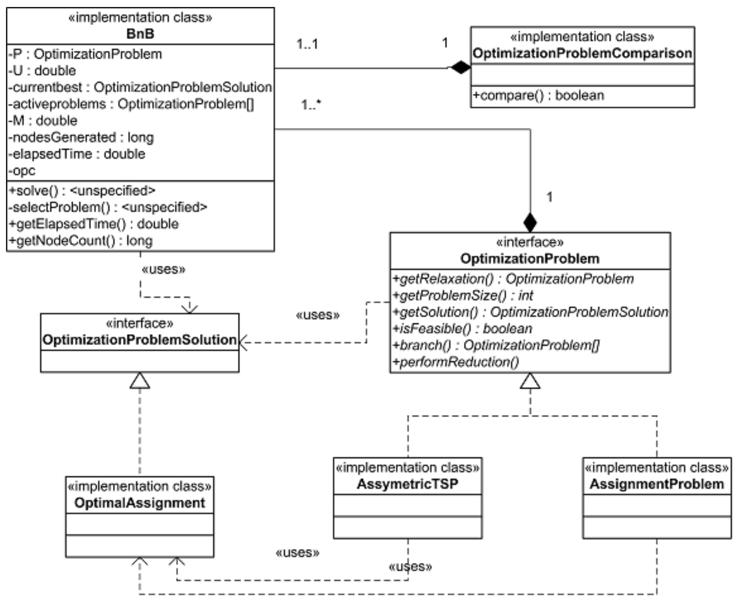


Figure 10. Example of black and white normal UML class diagram

Result: 45.69% of the result is the same as in the standard list

Experiment 3. Black and white complicated UML class diagram.

Example image:

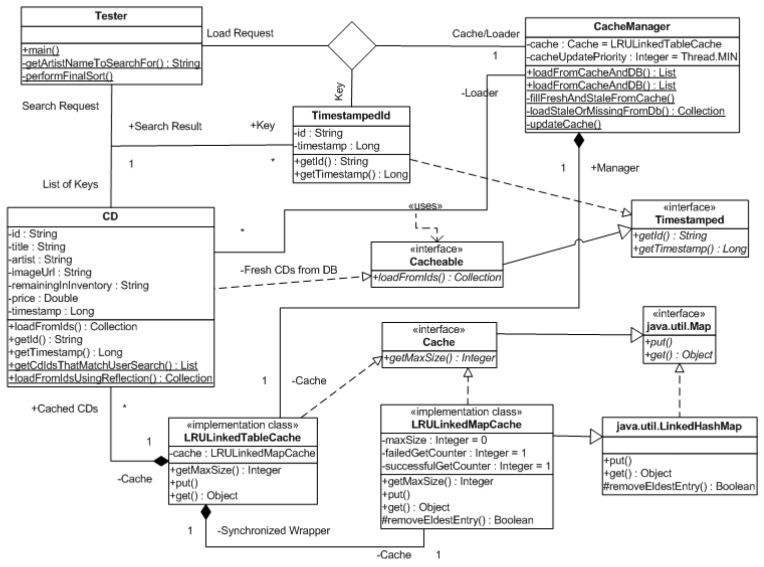


Figure 11. Example of black and white complicated UML class diagram

Result: 46.87% of the result is the same as in the standard list

Experiment 4. Colored simple UML class diagram.

Example image:

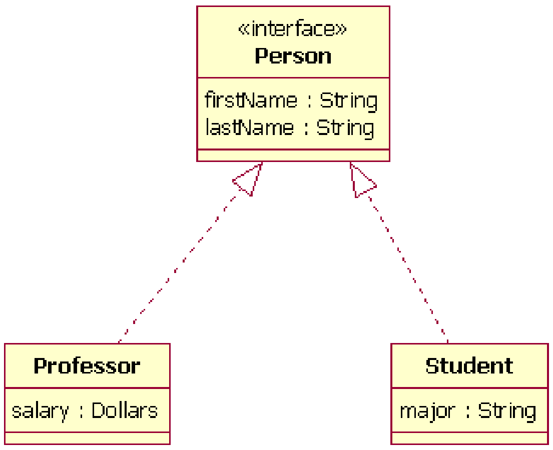


Figure 12. Example of colored simple UML class diagram

Result: 46.09% of the result is the same as in the standard list

Experiment 5. Colored normal UML class diagram.

Example image:

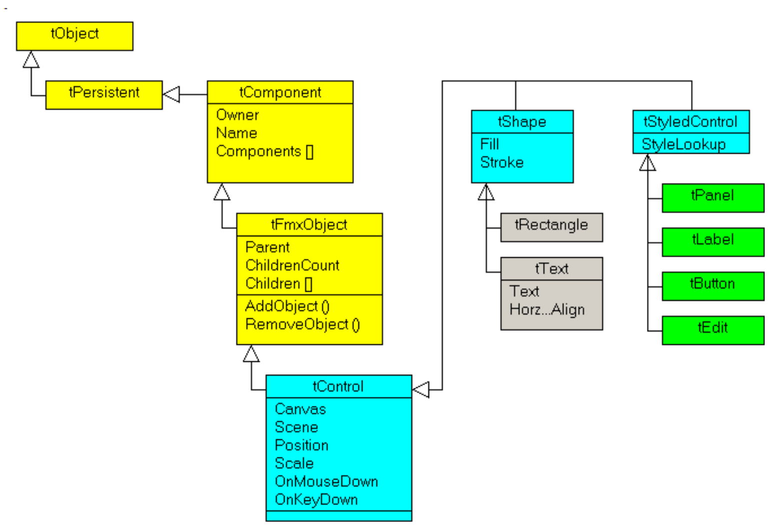


Figure 13. Example of colored normal UML class diagram

Result: 47.55% of the result is the same as in the standard list

Experiment 5. Colored complicated UML class diagram.

Example image:

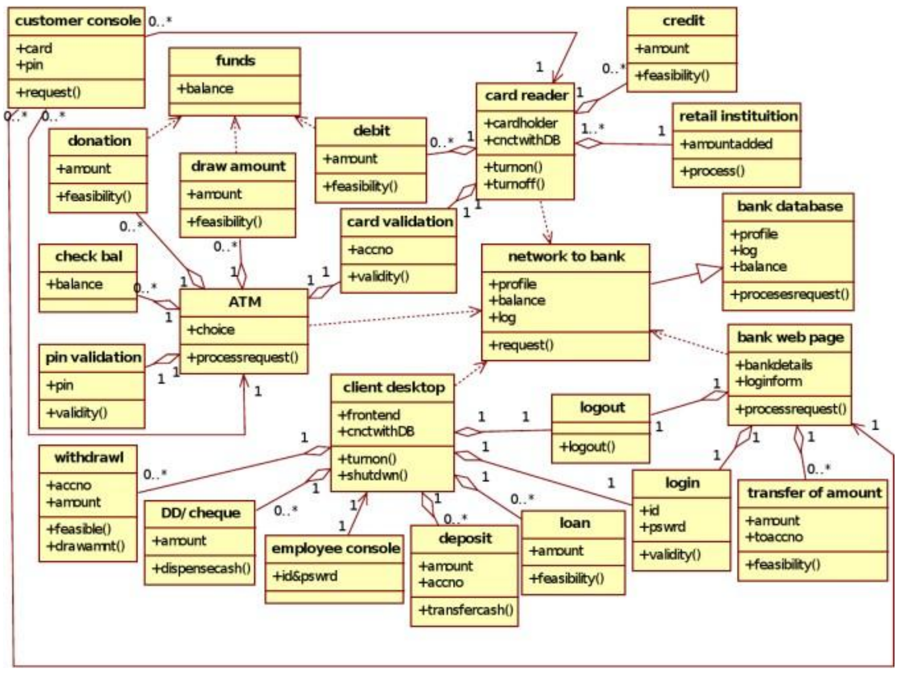


Figure 14. Example of colored complicated UML class diagram

Result: 47.16% of the result is the same as in the standard list

2.9.3 Conclusion of Filter4ImageCrawler

From the result, we can see that the correctness the program can give is nearly 50%, no matter the example image is black and white or colored, or the example image is simple or complicated. The results the program gives remain in a stable level, but not high enough to be trusted.

The reason comes from the algorithm itself. The perceptual hash algorithm compares similarity between two images by their "fingerprints". If two images have something in common, the more unique it is, for example, the Eiffel Tower, or the portrait of a movie star, the more similar the fingerprints of the images are. As for UML class diagrams, the components are mostly shapes like rectangles, triangles, and diamonds together with some text. They are not very unique features in an image. Thus, the fingerprint of a UML class diagram tends to be an ordinary one. As a result, the correctness of this program is not good enough because it mistakes some images as UML class diagrams that are not according to their fingerprints.

A suggestion for the future development of this program is to use shape recognition technology. The fingerprints of UML class diagrams are the elements. If the program can detect the shapes like rectangles with text inside, arrows pointing to rectangles and so on, the accuracy will be improved to a higher level. After all, this program is takes an example image instead of keywords to filter images. It’s difficult to reach a high accuracy.

Chapter 3. Website for UML Database Collection

3.1 Improvement to Distributed Web Crawler

There is a problem in *ImageCrawler* that the number of images has been limited by Google image search, the solution is to improve the hardware. Because we have implemented this *ImageCrawler* on a PC, to start from zero is not efficient. That's why we use the result of Google image search as the starting node.

The *ImageCrawler* program is implemented by a PC which does not have the ability to deal with great capacity of information from the Internet. A web crawler of high performance can be reached by distributed system.[21][22] Most search engines have different IDCs (Internet Data Center). Each IDC is in charge of a range of the IPs. Within each IDC, there are many instances of web crawler working parallel.

There are two methods used by the distributed web crawler.

1. All web crawlers of the distributed system rely on the same connection to the Internet. They co-operate to fetch data and combine the result within a local area network. The advantage of this method is that it's easy to expand the hardware resource. The bottleneck will appear in the bandwidth of the connection to the Internet that are shared by all instances of web crawlers.
2. The web crawlers of a distributed system are far away from each other geographically. The advantage of this method is that there is no problem about the insufficiency of bandwidth. However, the conditions of the network differ from place to place. Thus, how to integrate the result of different web crawlers is the most important problem.

3.2 Website for Database Collection

As the hardware for building a distributed system is not possible at the moment, a website can be established to collecting databases from users who have downloaded the *ImageCrawler* program.

The website provides the download link of the program. When a user has built a database, he can upload it in the website. After a database has been uploaded, the website will read its content and write them into the database saved in the server that contains all the information from the databases uploaded.

It is an alternative way to the distributed system of web crawler. If more and more people use the program to collect images and upload the URLs of the images, together with the “whitelist”/”blacklist” showing which URLs contain images that are (not) UML class diagrams, and the statistics of the domain from the “whitelist”/”blacklist”, a database will be available on the server that contains the index of a large number of images that are UML class diagrams. It's like the index of images built by a search engine, only that people using the software take the place of web crawlers.

3.2.1 Introduction of Drupal

Drupal is a free open-source content management system(CMS) that's developed by PHP[23]. It's best used for building a backend system. About 2% websites in the world are based on Drupal, including http://www.whitehouse.gov, the website for the White House. Here's a list that shows the famous websites that use Drupal as the framework:
http://www.seo-expert-blog.com/list/the-most-incomplete-list-of-drupal-sites .
Drupal is combined by modules. It is convenient to add or remove functions on it. In fact, rather than a CMS, it is more regarded as a framework for a content management website. Web developers around the world have contributed a lot in making it more and more powerful. At the same time, it has a useful access control system. Different permissions can be provided for different users of the websites built on Drupal. Last but not least, it is appreciated by graphic designers because the appearance of Drupal websites is flexible to adapt. Various "theme" templates are available for different requirements.

3.2.2 Website Design

The target of our website is to set up a database that contains the index of images of UML class diagrams. The functions of our website are:

1. User control system. People can register and log in through this user control system. The only requirement is a unique username and the email address. A confirmation letter will be sent to the email address after register to provide the user with a one-time log in link. Through this link, the user can set his password, portrait and other information. The user control system can let the users build their own databases and protect their privacy. The user can also choose whether to share his database with others.
2. The download link of the *ImageCrawler* program for the users to build their own database of images of UML class diagrams.
3. The function for the users to upload their databases and merge to our core database. Our core database contains the entire index that the users have collected. The required format of the upload file is *csv*, which can be generated by the *ImageCrawler* program. If it's the first time a user uploads the database, his database containing the five tables will be created on the server. The five tables will be named after the user's id. Thus, the users' databases won't have the conflict of the same names. What's more, using user's id instead of username(although unique) to name his database can give the most privacy for the user.
4. The function to show the users the core database and databases of other users. A user can choose whether his personal database is allowed to be shown to others. When a database contains the URL of images, the user can choose whether to show the thumbnails in the webpage. If chosen, the corresponding thumbnail will be shown dynamically when the mouse moves over the URL of an image. However, the thumbnail-showing function will take some resource of the user's system which may slow down his computer.

3.2.3 Future Work for the Website

Till now, the function of the website is mainly for database operation. Drupal is originally a content management system. So our website can be extended to a community for the users to make comments for the databases and exchange ideas with each other.
There are two drawbacks of the website.

1. When the user uploads his database, the csv file will be uploaded. Although there's a function to check whether the file type is *csv*, there's no way to check the content of the file. If it's not in the same format as the *csv* files created by the *ImageCrawler*, undefined content will be merged into the databases on the server.
2. Although there's an option for the user to choose whether he wants to share his database with other users, the content of his database will be included in our core database. Thus, when the core database is shown, a user's privacy may have been violated. One way to solve this problem is not to show our core database to the users. However, we suppose that users of our website are mostly willing to share what he has collected from our program. Thus, showing the users our core database is a better choice.

3.3 Change of *ImageCrawler*

To cooperate with the website, the *ImageCrawler* has been changed. It can search by URL instead of by key words. The program behaves as a normal web crawler.

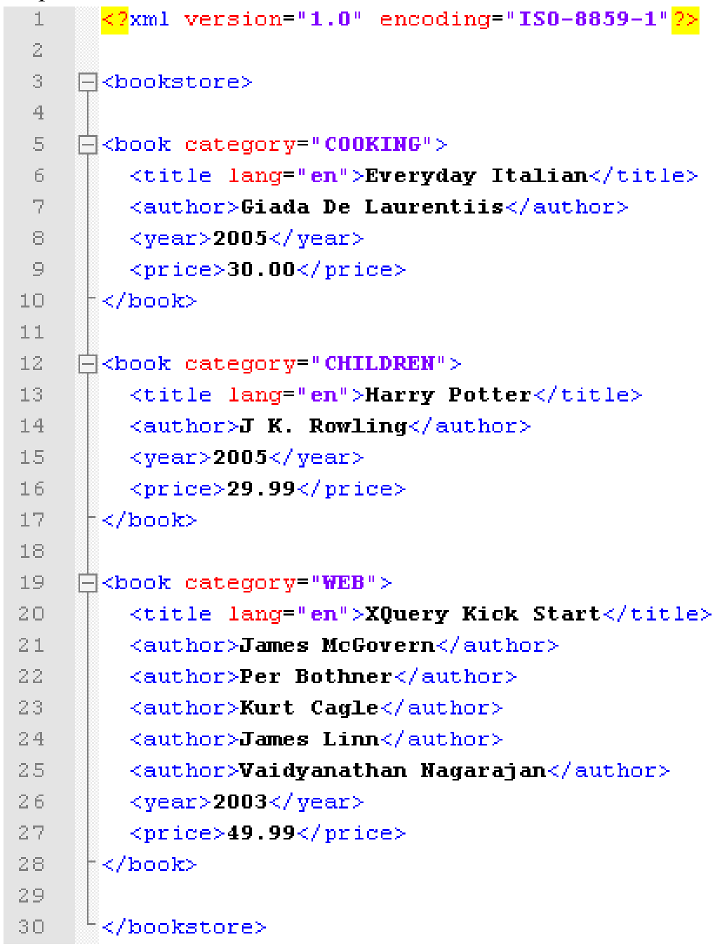
1. It uses the URL provided by the user as a starting node.
2. It gets the page data of the URL and analyzes the page content.
3. It uses the breadth first search to collect all the URLs included in the content of the page and saves them into a list. The list will only receive URLs that are not already included. Otherwise, the program may deal with a URL that's visited before and trapped in an infinite cycle.
4. It detects all the links to the images within the page content and downloads them.

The other functions remain the same.
The program no longer relies on the result of the Google image search. Users can focus on one website and collect the images from it. Compared to the result of Google image search which contains images from different websites, the result of searching by URL can be more focused on one topic because the images downloaded are mainly from one website.
Combining the *ImageCrawler* and the website, an alternative way for distributed system is finished. It's a better way because the website is not only for uploading and showing databases, but also a way for people to communicate and has a good extensibility for more functions.

Chapter 4. Website for XMI Storage and Query

4.1 Overview of XML

XML is short for "Extensible Markup Language"[24]. It is a language that uses tags to mark the electronic files to make them structural. It can be used to mark the data and define the data types. The tags can be defined by the users which provide it with a strong extensibility. XML is a subset of SGML(Standard Generalized Markup Language) and the recommended standard by the W3C(World Wide Web Consortium). It is very popular and useful in web transmission.
The grammar of XML is similar with HTML. However, XML is not an alternative of HTML. XML is used to store data while HTML is used to format and print data to the webpage.
XML consists of tags and the tags must appear in pairs which is different with HTML. XML is case sensitive, which requires a stricter attitude in using it. An XML file is like a tree. It starts from a root node and expand to the leaves.
Here's an example of an XML file.



```xml
<?xml version="1.0" encoding="ISO-8859-1"?>

<bookstore>

<book category="COOKING">
  <title lang="en">Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>

<book category="CHILDREN">
  <title lang="en">Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>

<book category="WEB">
  <title lang="en">XQuery Kick Start</title>
  <author>James McGovern</author>
  <author>Per Bothner</author>
  <author>Kurt Cagle</author>
  <author>James Linn</author>
  <author>Vaidyanathan Nagarajan</author>
  <year>2003</year>
  <price>49.99</price>
</book>

</bookstore>
```

Figure 15. An example of XML file

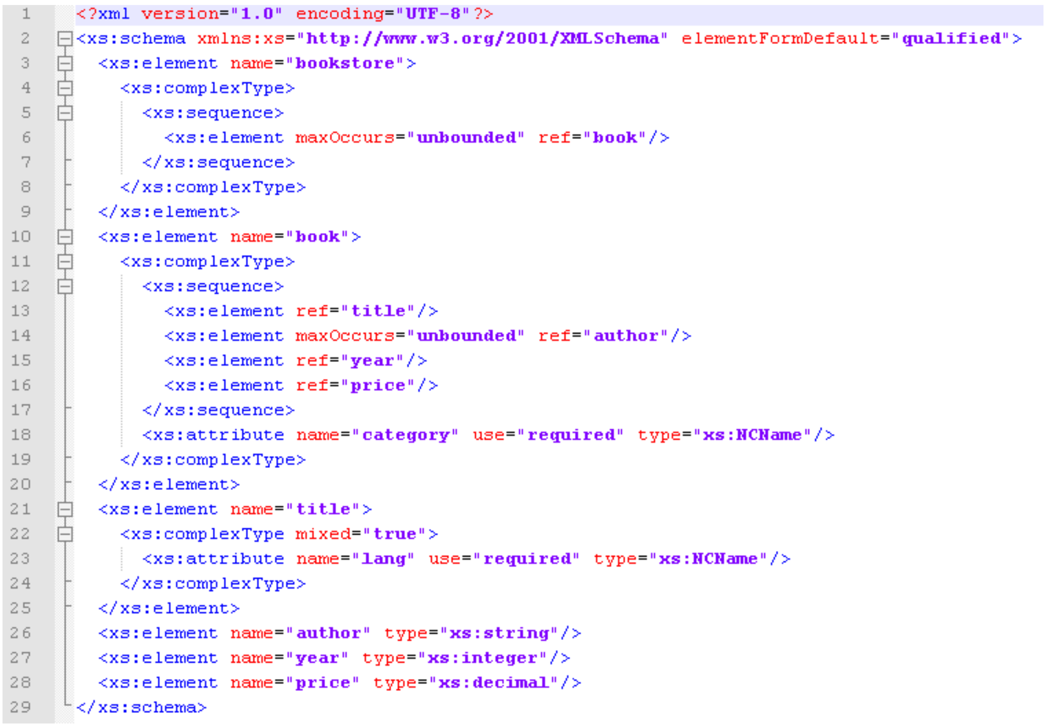
In this file, the first line is the declaration of this file. It defines the version of XML is 1.0 and the encoding is “ISO-8859-1”.
From line 2 to line 30, the entities in the tags are called elements. The <bookstore> and </bookstore> are the two parts of the root element. The root element should appear in every XML file. It is the parent node of any other elements.
An element can have attributes. For example, every <book> element has an attribute called “category”. The value of an element appears between the start and end tags of the element. For example, “Everyday Italian” is the value of the first <title> element.

4.1.1 Overview of XML Schema

XML Schema describes the structure of an XML file. It is also a standard of W3C. It contains several aspects[25].

- An XML Schema defines the elements that appear in the XML file.
- An XML Schema defines the attributes that appear in the XML file.
- An XML Schema defines the sequence of the appearance of elements.
- An XML Schema defines the number of elements.
- An XML Schema defines whether an element is empty.
- An XML Schema defines the data type of elements and attributes.
- An XML Schema defines the default value of elements and attributes.

XML Schema uses the same grammar with XML. Thus, it is easy to understand and write.
Here’s the XML Schema corresponding to the XML example above.



```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="bookstore">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" ref="book"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="title"/>
        <xs:element maxOccurs="unbounded" ref="author"/>
        <xs:element ref="year"/>
        <xs:element ref="price"/>
      </xs:sequence>
      <xs:attribute name="category" use="required" type="xs:NCName"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="title">
    <xs:complexType mixed="true">
      <xs:attribute name="lang" use="required" type="xs:NCName"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="author" type="xs:string"/>
  <xs:element name="year" type="xs:integer"/>
  <xs:element name="price" type="xs:decimal"/>
</xs:schema>
```

Figure 16. An example of XML Schema

The “xs:” that appears in front of every element is the namespace. The <schema> tag is the root element of an XML Schema file. It must appear.

The <element> elements describe the elements that appear in the XML file. In this case, there are "bookstore", "book", "title", "author", "year" and "price". If the value of an element is a simple type, it will be described as an attribute of the element. For example, the "author" element contains the type "string". Thus, in line 26, the string type is described as an attribute of the "author" element. If the value of an element is a complex type, it will be described explicitly. For example, the element "book" in the XML file has four elements embedded, so in line 11 and 19, it is described as a complex type.

4.2 Overview of XMI

XMI is short for "XML-based Metadata Interchange". It provides a method for the developers to exchange the metadata by XML.
XMI integrates three industry standards. They are XML, UML and MOF(Meta Object Facility), an OMG(Object Management Group) language for specifying meta models.[26]

4.2.1 XMI vs. Relational Database

Our project uses XMI as the output of the OCR process for images of UML class diagrams. We have built a website for XMI storage and query.
First, there's a comparison of the difference between an XMI file and the relational database.[27]

	XMI	RDB
Data storage	File system	Database system
Data structure	Only related to logical structure	Only related to logical structure
Pattern description	DTD, Schema	Relational patterns
Logical interface	SAX, DOM	ODBC, JDBC
Query language	XQuery	SQL
Safety level	Low	High
Concurrent query	No	Yes
Usage	Standard for Internet data exchange	Method for storing and querying for data
Operating language	No standard language	SQL
Internet usage	Directly	Need application
Data description	Structural or semi-structural	Structural

Table 5. Comparison of XML and relational database

From the table, we can see that XMI is really a good method for exchanging data through the Internet. However, it's not good in operating with data. On the other hand, relational database can efficiently query within the data, but it needs a flexible way to publish the data into the Internet. Thus, to combine XMI with relational database is a good solution.

4.3 Related Work

4.3.1 XMI Storage

Phil Bohannon's paper[28] has provided a way to map XMI files into relational database. It comes up with a cost-based method for XMI storage in relational database.

In this method, the first step is to generate a physical XML Schema. A physical XML Schema is an equivalent way of XML Schema. It eliminates the multi-valued elements by generating separate new types for each of them. Then, for the attributes of each type, the size, minimum and maximum value and the number of distinct values is inserted in the physical XML Schema. Then, it is ready to map from the Schema to the relations.

The physical XML Schema is a great method of mapping from XML Schema to relational database. However, it ignores the constraints of the elements like "minoccurs", "maxoccurs" or "unique"[29]. What's more, it deals with the XML Schema so that it is used to create the tables instead of build a whole database. But it provides us a good idea about how to design a database for an XMI file.

4.3.2 XMI Query System

In Daniel Lucredio, Renata P. de M. Fortes and Jon Whittle's paper[30], a metamodel-based searching engine named "Moogle" is created. The system is an efficient model search engine.

Here's the architecture of the *Moogle* search engine.

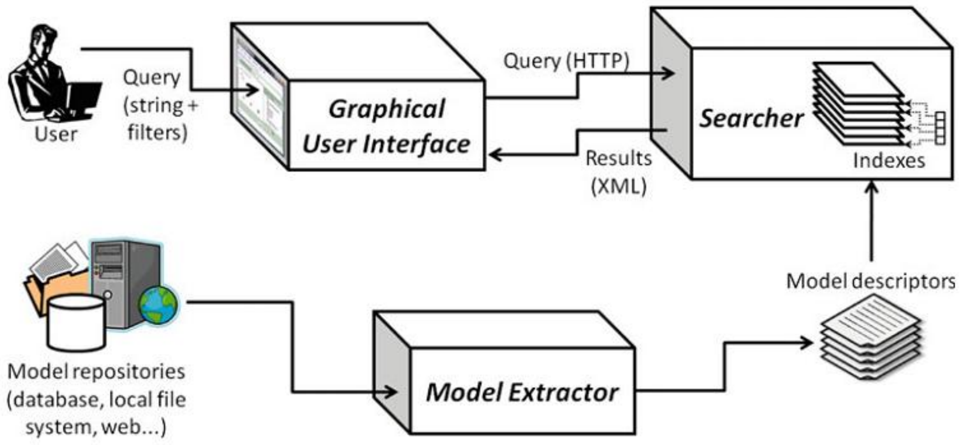


Figure 17. The architecture of MOOGLE

The system consists of three parts: the model extractor, the searcher and the user interface.

The model extractor can find and read model files from different repositories. It reads the content of the model files, extract necessary content and generate model descriptors which will be used in the searcher function.

The searcher is based on an open source search engine: Apache SOLR. With SOLR, an index for the model descriptors is established that contains all the necessary information of them.

The graphical user interface of *Moogle* provides the users with three functions: simple search, advanced search and metamodel browsing.

Moogle is a good idea for us to build a website in searching for models. It supports query in different kinds of models. However, it has some drawbacks.

The core function -- model extractor uses the configuration file to get necessary content from them. A valid unique namespace declaration is required. Thus, if the namespace declaration of a model file is damaged, it will stop generating no matter how well the other part is preserved.

Moogle is a search engine for model files that are presented with text. What we want to achieve is a project for models that are presented in UML class diagrams. As there are already different search engines that deal with text, it's more difficult to turn images into text models and do the query.

Another drawback is the searching method. *Moogle* uses indexes for model descriptors to implement the query process. It is a search within the files. Thus, the efficiency will not be as good as querying in a relational database.

4.4 XMI Database

Our standard is based on the XMI files exported from starUML. There are four categories that we take into account: class, attribute, operation and association. Our database is built on that foundation.

4.4.1 Database Design

Our database contains four tables: *tblClass*, *tblAttribute*, *tblOperation* and *tblAssociation* corresponding to classes, attributes, operations and associations. Here are the keys in the tables.

tblClass		
Key	**Type**	**Comment**
name	varchar(200)	Name of the class
visibility	varchar(20)	Public, private
isAbstract	varchar(20)	Whether it's an abstract class
xmi_id	varchar(200)	The id for this class
datetime	varchar(20)	The date and time this item is generated
url	varchar(200)	The URL of the source image of this class

Table 6. The table to store classes

tblAttribute		
Key	**Type**	**Comment**
name	varchar(200)	Name of the attribute
visibility	varchar(20)	Public, private
typeofAttr	varchar(200)	The type of this

		attribute
owner	varchar(200)	The id of the class this attribute belongs to
xmi_id	varchar(200)	The id for this attribute
datetime	varchar(20)	The date and time this item is generated
url	varchar(200)	The URL of the source image of this attribute

Table 7. The table to store attributes

tblOperation		
Key	**Type**	**Comment**
name	varchar(200)	Name of the operation
visibility	varchar(20)	Public, private
isAbstract	varchar(20)	Whether it's an abstract operation
owner	varchar(200)	The id of the class this operation belongs to
xmi_id	varchar(200)	The id for this operation
datetime	varchar(20)	The date and time this item is generated
url	varchar(200)	The URL of the source image of this operation

Table 8. The table to store operations

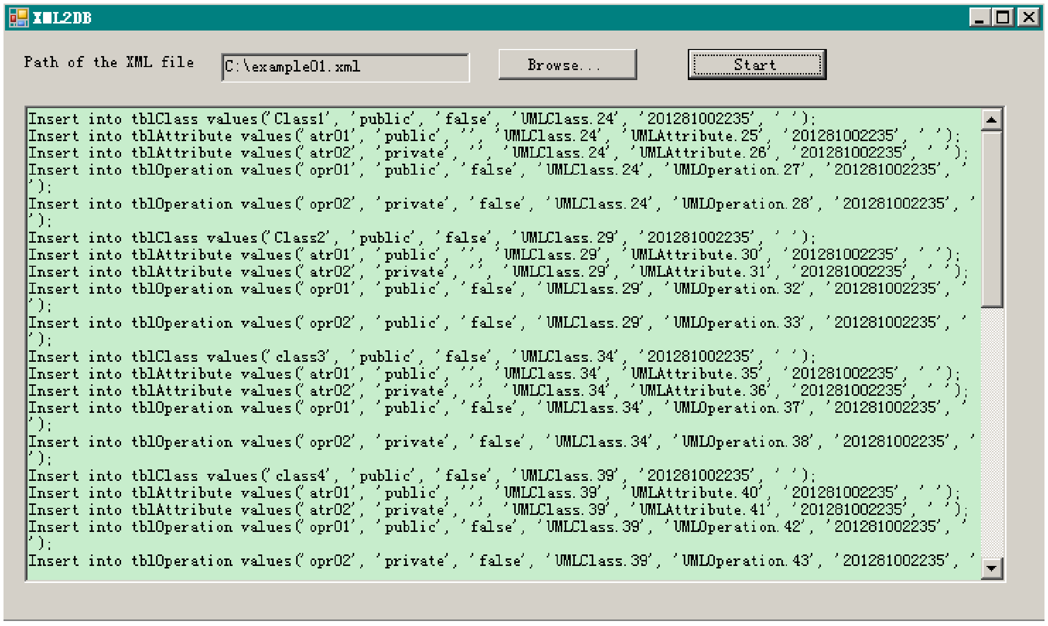
tblAssociation		
Key	**Type**	**Comment**
name	varchar(200)	Name of the association
visibility	varchar(20)	Public, private
ordering	varchar(200)	The direction of this association
aggregation	varchar(200)	The type of the aggregation
association	varchar(200)	The id of the association this item belongs to
typeofAsso	varchar(200)	The id of the class this association points to
xmi_id	varchar(200)	The id for this operation

datetime	varchar(20)	The date and time this item is generated
url	varchar(200)	The URL of the source image of this operation

Table 9. The table to store associations

4.4.2 SQL Generator

We have developed a program XML2DB to convert from an XML file to SQL commands to insert the content to our database. The main feature we have used is the *System.Xml* namespace of *.Net*. In this program, we load the XML file generated by the OCR process, get the attributes of classes, attributes, operations and associations in this file, generate the "Insert into" command and write them into a SQL file. The SQL commands generated will be shown in the textbox.
Here's a screen shot of this program.



```
Insert into tblClass values('Class1', 'public', 'false', 'UMLClass.24', '201281002235', ' ');
Insert into tblAttribute values('atr01', 'public', '', 'UMLClass.24', 'UMLAttribute.25', '201281002235', ' ');
Insert into tblAttribute values('atr02', 'private', '', 'UMLClass.24', 'UMLAttribute.26', '201281002235', ' ');
Insert into tblOperation values('opr01', 'public', 'false', 'UMLClass.24', 'UMLOperation.27', '201281002235', '
');
Insert into tblOperation values('opr02', 'private', 'false', 'UMLClass.24', 'UMLOperation.28', '201281002235', '
');
Insert into tblClass values('Class2', 'public', 'false', 'UMLClass.29', '201281002235', ' ');
Insert into tblAttribute values('atr01', 'public', '', 'UMLClass.29', 'UMLAttribute.30', '201281002235', ' ');
Insert into tblAttribute values('atr02', 'private', '', 'UMLClass.29', 'UMLAttribute.31', '201281002235', ' ');
Insert into tblOperation values('opr01', 'public', 'false', 'UMLClass.29', 'UMLOperation.32', '201281002235', '
');
Insert into tblOperation values('opr02', 'public', 'false', 'UMLClass.29', 'UMLOperation.33', '201281002235', '
');
Insert into tblClass values('class3', 'public', 'false', 'UMLClass.34', '201281002235', ' ');
Insert into tblAttribute values('atr01', 'public', '', 'UMLClass.34', 'UMLAttribute.35', '201281002235', ' ');
Insert into tblAttribute values('atr02', 'private', '', 'UMLClass.34', 'UMLAttribute.36', '201281002235', ' ');
Insert into tblOperation values('opr01', 'public', 'false', 'UMLClass.34', 'UMLOperation.37', '201281002235', '
');
Insert into tblOperation values('opr02', 'private', 'false', 'UMLClass.34', 'UMLOperation.38', '201281002235', '
');
Insert into tblClass values('class4', 'public', 'false', 'UMLClass.39', '201281002235', ' ');
Insert into tblAttribute values('atr01', 'public', '', 'UMLClass.39', 'UMLAttribute.40', '201281002235', ' ');
Insert into tblAttribute values('atr02', 'private', '', 'UMLClass.39', 'UMLAttribute.41', '201281002235', ' ');
Insert into tblOperation values('opr01', 'public', 'false', 'UMLClass.39', 'UMLOperation.42', '201281002235', '
');
Insert into tblOperation values('opr02', 'private', 'false', 'UMLClass.39', 'UMLOperation.43', '201281002235', '
```

Figure 18. The interface of XML2DB

4.5 Website Design for XMI Storage and Query

There are two functions of the website: XMI storage and query.

4.5.1 XMI Storage

Taken safety into account, we only let the administrator of the website to upload the SQL file generated from the XML file. After the SQL file is uploaded, the website will abstract the commands inside and execute them. Thus, the content of the XML file is written in our database.

4.5.2 XMI Query

We provide a query function for the user of the XMI database. There are two ways to make a query of the XMI database.

- Search by URL.
  After a search of a database of the information of images, the result page will show a list of URLs of the images. The user can select by choosing the checkboxes in front of the items to search for all the information(classes, attributes, operations and associations) contained in the selected images. The result page will show the images and all their information. The information of each image will be shown in four tables: class, attribute, operation and association.
- Search by keywords.
  The user can input keywords, and the models containing the keywords will be selected. We provide all the information of the images containing the models to the user.
  If the user inputs more than one keyword, we take the keywords separated by space as the relation "AND". So the result should be items containing all the keywords. The user can use "OR" to connect keywords. Then, the result will show models that contain either of the keyword connected by the "OR" condition.
  By default, we provide a fuzzy enquiry for the user. The result will include models that the keywords partly match. For example, if the user inputs "edu", models with "education" will be matched. The user can choose the precise search mode by selecting the option on the page. Then, models with "education" will only be matched by the keyword "education".

4.6 Validation

The advantages of our website are:

- Simpler transformation.
  As we have our own OCR process to generate XML files, the transformation from XMI to relational database is simpler and faster.
- Good efficiency.
  Relational database is used to store the elements in XML files. We split the models into four parts: class, attribute, operation and association. We sacrifice the space to get the advantage of querying efficiency. Thus, the query process is faster than searching through model files on the server.
- Model images provided.
  As our project aims at querying models that are saved in images, users of our website will get the access of images we have collected from the Internet. Presenting models in a graphical view is friendlier than by text, even the text has been formatted.
- Access for users to communicate.
  Our website is based on a content management system, so there's access for the users to make comments and communicate their ideas with each other.

For future development, a smarter search engine can be developed. Our project can also expand to search with text files. As no official standard of XMI for models is established, many configuration files will be needed to transform from different model files to our standard in order to insert into our database. The website can be expanded to a community for researchers in this field to communicate and exchange ideas.

Chapter 5. Conclusion

In this project, we have built a system that establishes databases of images of UML class diagrams from the Internet, turns the images to XMI models via image processing technology, stores the models into relational database and queries within the models.
A website is established to share our results. It can also let the users make comments and communicate with each other.

For future research, the programs we have developed is planned to be written in the form of web applications and integrate with our website to make a complete online system for collecting, recognizing and querying models that are presented in images. More efficient web crawler algorithms can be developed to improve the percentage of UML class diagrams abstracted by the *ImageCrawler* program.

Appendix A. User Manual for *ImageCrawler*

This is the user-interface when the program starts. The top part is the image downloading part, and the bottom part is the domain rank part.

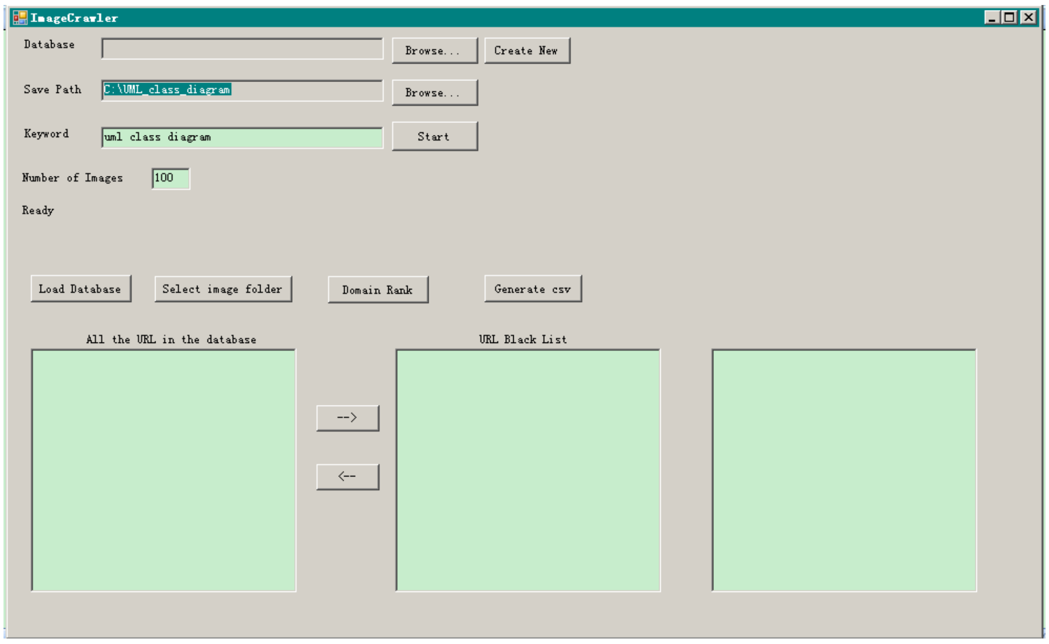


Figure 19. The initial interface of ImageCrawler

A.1 Image Downloading Part

The image downloading part is to download images from the result of Google image search based on the keywords the user has input. Then, the information of those images will be saved into an access database.

Step 1. Select the database that will restore the information of images to be downloaded.

There are two buttons to the right of the "Database" textbox providing two ways of the database selection.

The "Create New" button can create an empty access database file that contains all the five tables of our designed database – pic, *picblack*, *picwhite*, *blackcount*, *whitecount*. The new access file will be created in the same folder with the executable program file. It will be named "pic" with the current date and time attached to it. For example, if the database is created on 18:30:25, December 21, 2012, the name will be "pic20121221183025.mdb". In this way, the conflict of duplicated file name will be saved.

Another way is to click the "Browse…" button to select an access file that already contains the five tables. After the user has selected a database file, the program will check whether the database has the tables and keys required by the program.

When a database file is selected or created, the full path will be shown in the "Database" textbox.

Step 2. Select the folder the images will be saved.

The path is shown in the “Save Path” textbox.

There is a default value: "C:\UML_class_diagram". The user can change it by clicking the “Browse…” button next to the textbox.

Step 3. Input the key words to search.

The default keyword is "uml class diagram". The user can input his choice.

Step 4. Set the number of images to be downloaded.

The default number is 100. The user can input his choice.

Step 5. When step 1 to step 4 is finished, the user can start the downloading process by clicking the "Start" button. The program will first check if the database, save path, keywords and number of images have been set. If not, there will be a hint to tell the user to finish that first. The downloading process may take a while to finish. It depends on the number of images the user wants and the condition of the user’s Internet connection. When finished, a message box will pop up to acknowledge the user.

A.2 Domain Rank Part

The domain rank part implements an analysis function. The user can build a “blacklist” that contains the images not related to the keywords. Then, he can get the statistical data of the domains. What’s more, the program can give an output of the database in the form of csv files.

Function 1. Load the database.

Click the "Load Database" button to choose a database. After a database is chosen, the program will check whether the tables and keys of the database meet the requirement. Then, the URLs of the images will be loaded. The URLs in the table "pic" which contains all the URLs of the images will be shown in the list box on the left. The URLs in the table "picblack" which contains all the URLs of images in the “blacklist” will be shown in the list box in the middle.

Function 2. View the images in the database.

When a URL of either the left or the middle list box has been selected, the program will show the corresponding image in the picture box on the top right. But before that, the user must click on the “Select image folder” button to select where the images are saved.

Function 3. Manipulating the “blacklist”.

The user can manually add or delete items from the “blacklist”. There are two buttons between the list of all URLs and the list of URLs in the “blacklist”. By clicking the button with the arrow "-->", the selected URL in the list of all URLs will be added to “blacklist”. The function of the button with the arrow "<--" can delete a URL from the “blacklist”.

Function 4. Ranking the domains.

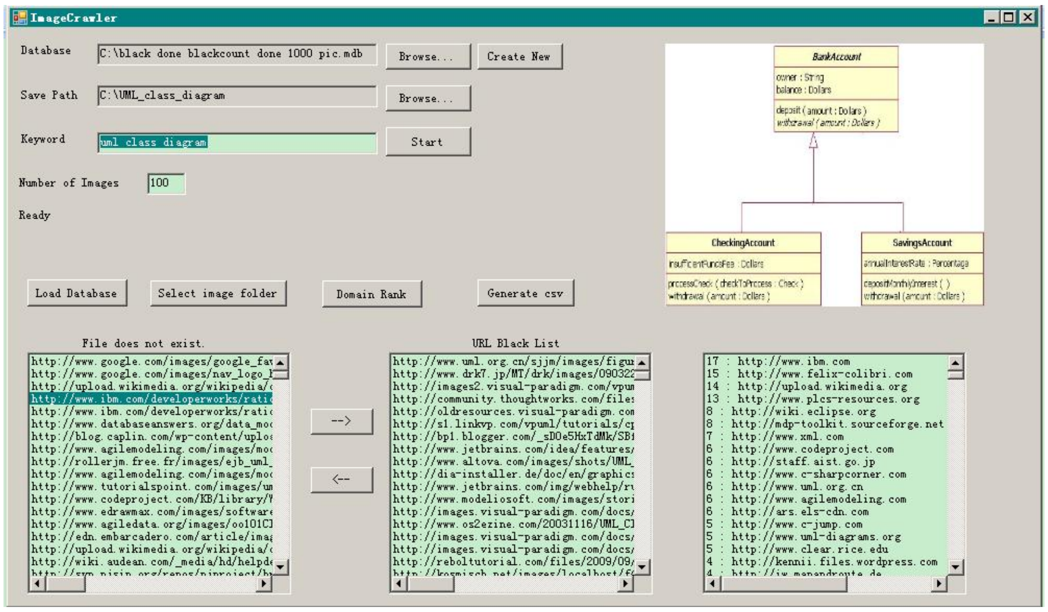
After the “blacklist” is established, the user may want to know which domains contribute many items to it.

After clicking the button "Domain Rank", the program will take an analysis of all the URLs that are in or not in the “blacklist”. Then, the domains of the images will be abstracted. The program will make a list of how many images have been downloaded from each domain. The result will be shown in the list box on the right side in descending order by the number of images contributed by the domains.

Function 5. Output to *csv* files.

As our online database requires *csv* files as the input, the program can generate *csv* files from the database. The user can click on the "Generate csv" button to generate five csv files corresponding to the five tables in the database. The five csv files will be created in the same folder with the executable program file. They will be named in the format of "csv_for_" and the name of the access database file and "_" with the table name. For example, if a database is named "pictures.mdb", the five csv files will be: "csv_for_pictures_pic.csv", "csv_for_pictures_picblack.csv", "csv_for_pictures_picwhite.csv", "csv_for_pictures_blackcount.csv" and "csv_for_pictures_whitecount.csv".

Here's a screen shot of the program in running.


ImageCrawler
Browse...
Create New
Browse...
Start
Load Database
Select image folder
Domain Rank
Generate csv
-->
<--
BankAccount
owner : String
balance : Dollars
deposit (amount : Dollars)
withdrawal (amount : Dollars)
CheckingAccount
insufficientFundsFee : Dollars
processCheck (checkToProcess : Check)
withdrawal (amount : Dollars)
SavingsAccount
annualInterestRate : Percentage
depositMonthlyInterest ( )
withdrawal (amount : Dollars)

```
Database          C:\black done blackcount done 1000 pic.mdb
Save Path         C:\UML_class_diagram
Keyword           uml class diagram
Number of Images  100
Ready
```

File does not exist.

```
http://www.google.com/images/google_fav
http://www.google.com/images/nav_logo_l
http://upload.wikimedia.org/wikipedia/c
http://www.ibm.com/developerworks/ratic
http://www.ibm.com/developerworks/ratic
http://www.databaseanswers.org/data_moc
http://blog.caplin.com/wp-content/uploa
http://www.agilemodeling.com/images/moc
http://rollerjm.free.fr/images/ejb_uml_
http://www.agilemodeling.com/images/moc
http://www.tutorialspoint.com/images/um
http://www.codeproject.com/KB/library/Y
http://www.edrawmax.com/images/software
http://www.agiledata.org/images/oo101Cl
http://edn.embarcadero.com/article/imag
http://upload.wikimedia.org/wikipedia/c
http://wiki.audean.com/_media/hd/helpde
```

URL Black List

```
http://www.uml.org.cn/sjjm/images/figur
http://www.drk7.jp/MT/drk/images/090322
http://images2.visual-paradigm.com/vpum
http://community.thoughtworks.com/files
http://oldresources.visual-paradigm.com
http://s1.linkvp.com/vpuml/tutorials/cp
http://bp1.blogger.com/_sD0e5HxTdMk/SBi
http://www.jetbrains.com/idea/features/
http://www.altova.com/images/shots/UML_
http://dia-installer.de/doc/en/graphics
http://www.jetbrains.com/img/webhelp/ru
http://www.modeliosoft.com/images/stori
http://images.visual-paradigm.com/docs/
http://www.os2ezine.com/20031116/UML_Cl
http://images.visual-paradigm.com/docs/
http://images.visual-paradigm.com/docs/
http://reboltutorial.com/files/2009/09/
```

```
17 : http://www.ibm.com
15 : http://www.felix-colibri.com
14 : http://upload.wikimedia.org
13 : http://www.plcs-resources.org
8 : http://wiki.eclipse.org
8 : http://mdp-toolkit.sourceforge.net
7 : http://www.xml.com
6 : http://www.codeproject.com
6 : http://staff.aist.go.jp
6 : http://www.c-sharpcorner.com
6 : http://www.uml.org.cn
6 : http://www.agilemodeling.com
6 : http://ars.els-cdn.com
5 : http://www.c-jump.com
5 : http://www.uml-diagrams.org
5 : http://www.clear.rice.edu
4 : http://kennii.files.wordpress.com
```

Figure 20. The interface of ImageCrawler in process

Appendix B. User Manual for *Filter4ImageCrawler*

Here's the screenshot of the Filter4ImageCrawler.

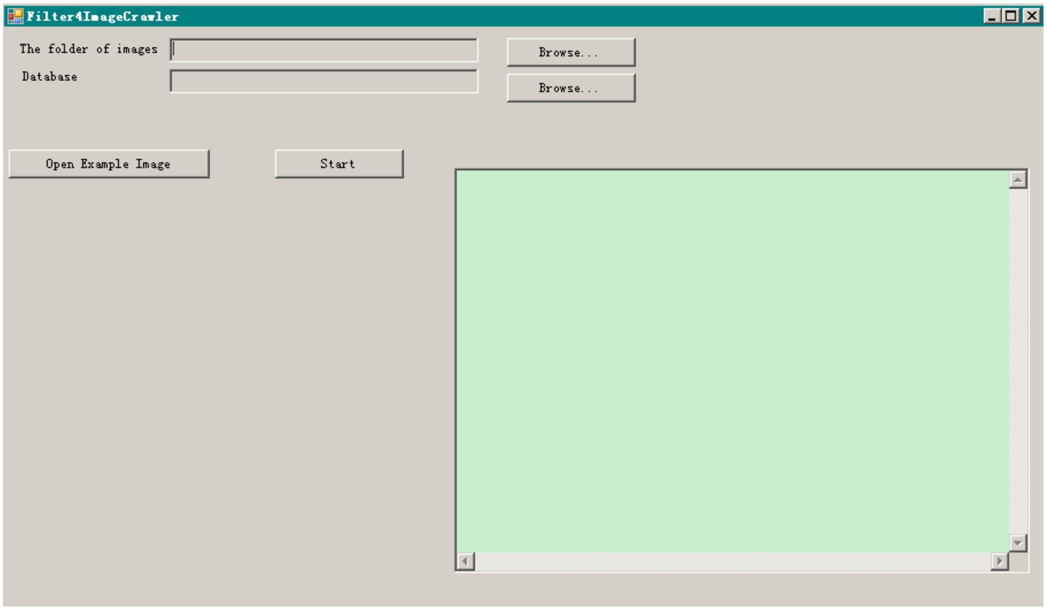


Figure 21. The initial interface of Filter4ImageCrawler

Step 1. Choose the folder of the images.

Click on the "Browse..." button next to the textbox on the top to set the path of the folder of images.

Step 2. Choose the database.

Click on the "Browse..." button next to the textbox of "Database" to set the path of the database. The program will check whether it meets the format of the database *ImageCrawler* has created. If not, there will be a message to warn the user to select again. The full path of the chosen database file will be shown in the "Database" textbox.

Step 3. Choose the example image.

The example image is the image that is taken as an example in the process of perceptual hash algorithm. All the images will be compared to the example for similarity.

Click on the "Open Example Image" button to choose an image as the example. The image will be shown in the picture box below the button.

Step 4. Start the process.

Click on the "Start" button, and the process will be started.

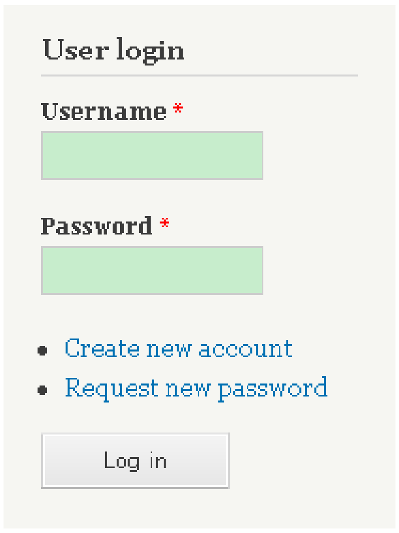
Step 5. View the result.

When the process is done, there will be a list of URLs of images shown in the list box on the right side. They are considered similar to the example image. The user can view the images by selecting a URL in the list box.

Appendix C. User Manual for Database Collection Website

C.1 User Control System

We use the user control system that's provided by Drupal. If a user has not logged in or registered, he doesn't have the permission to use the functions. A message will be shown on the webpage to let the user log in or register. Here's the screenshot of the log in part.



User login

Username *

Password *

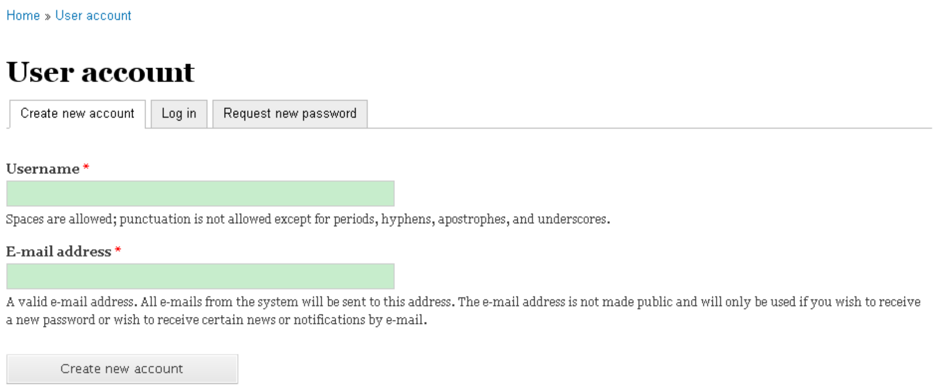
- Create new account
- Request new password

Log in

Figure 22. User login page

The "Create new account" link can let an anonymous use create his account. A unique username and email address is required for the user registration. When those information have been provided, a confirmation letter will be sent to the email address. Within this email, a link for one-time log in is provided. Considering the aspect of safety, the link will expire after one day. The user should log in via that link and set the password and other information. Then, the account will be activated to use the functions without restrictions.

Here's the screenshot of the "Create new account" page.



Home » User account

User account

Create new account | Log in | Request new password

Username *

Spaces are allowed; punctuation is not allowed except for periods, hyphens, apostrophes, and underscores.

E-mail address *

A valid e-mail address. All e-mails from the system will be sent to this address. The e-mail address is not made public and will only be used if you wish to receive a new password or wish to receive certain news or notifications by e-mail.

Create new account

Figure 23. User registration page

If a user has forgot his password, just use the "Request new password" function to get a new one.

When the username or email address has been input in this page, a one-time log in link will be provided to the user via his email address. Then the user can reset the password the same as when creating a new account.

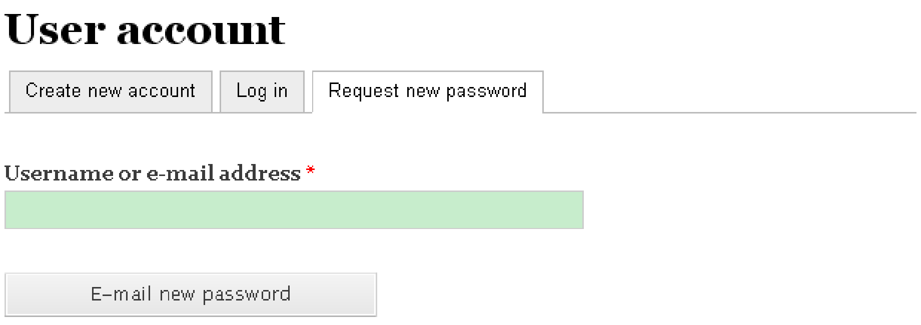
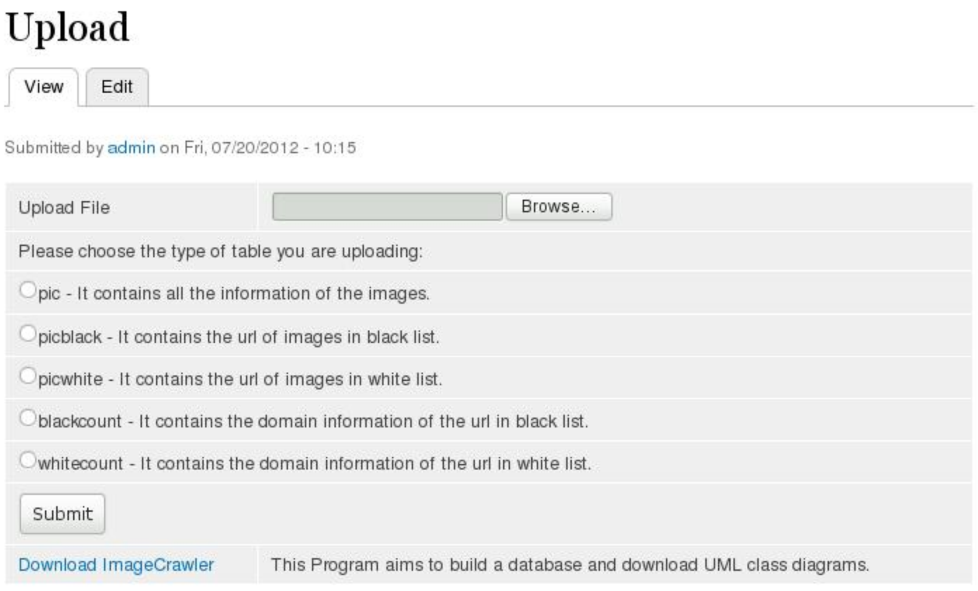


Figure 24. Request new password page

C.2 Upload Page

Here's the screenshot of the upload page.



Submitted by admin on Fri, 07/20/2012 - 10:15

Upload File

Please choose the type of table you are uploading:

pic - It contains all the information of the images.

picblack - It contains the url of images in black list.

picwhite - It contains the url of images in white list.

blackcount - It contains the domain information of the url in black list.

whitecount - It contains the domain information of the url in white list.

This Program aims to build a database and download UML class diagrams.

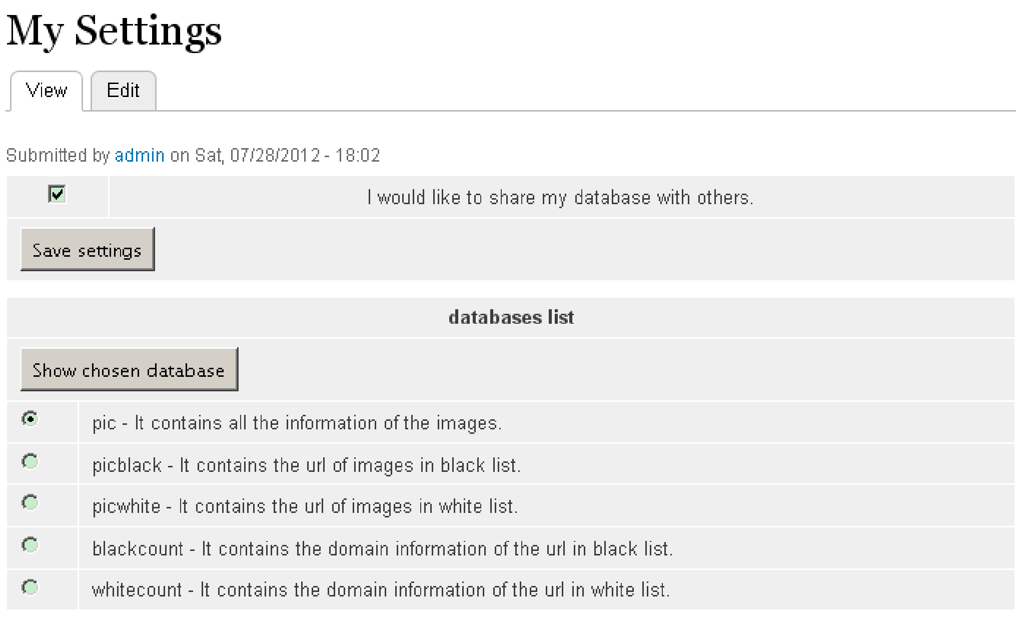
Figure 25. Upload page

In the bottom part of this page, the user can download the *ImageCrawler* program to build his database. Then, he can upload the *csv* files generated by the *ImageCrawler* program by clicking on the "Browse…" button. As our program will generate five *csv* file for a database that correspond to the five tables, the user should choose which table the *csv* file to be uploaded contains. When all the required information is set, just click the "Submit" button to upload the file.

The upload function will first check the type of the file uploaded. If a non-csv file has been uploaded, there will be a message to warn the user. If uploaded successfully, the information of the uploaded file will be shown to the user, like the name and size of the file. Then, the content of the *csv* files will be merged to his existed database on the server as well as our core database. If it's the first time the user uploads his database, the five tables will be built automatically on the server.

C.3 My Settings Page

Here's the screen shot of the page for the user settings.



Submitted by admin on Sat, 07/28/2012 - 18:02

✓	I would like to share my database with others.

databases list	
●	pic - It contains all the information of the images.
○	picblack - It contains the url of images in black list.
○	picwhite - It contains the url of images in white list.
○	blackcount - It contains the domain information of the url in black list.
○	whitecount - It contains the domain information of the url in white list.

Figure 26. My settings page

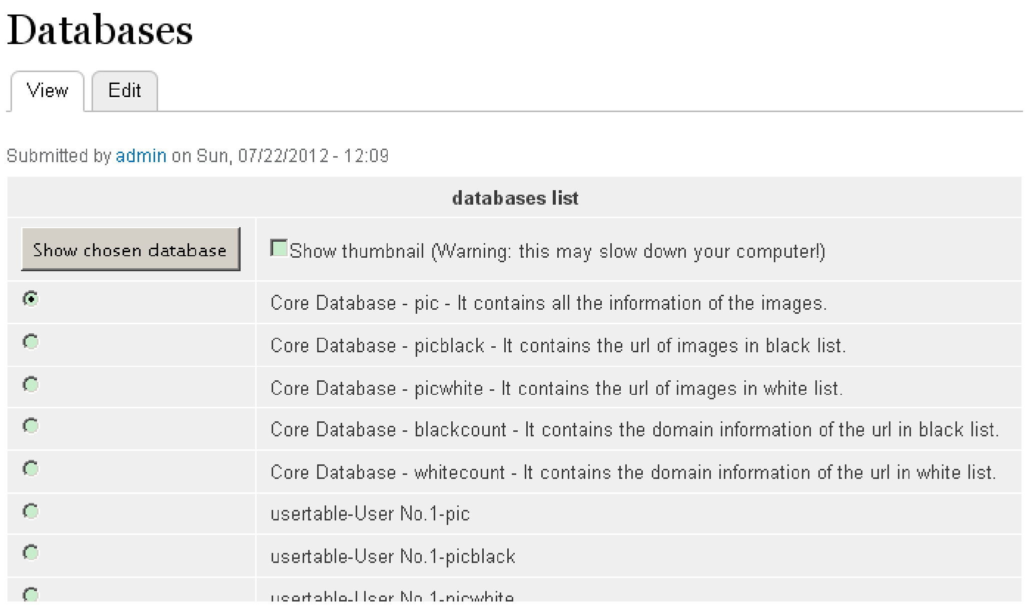
The page has been divided into two parts.

The top part is the general settings of the user. Now here's only one option: whether to share the user's database with the other users. The user can click on the "Save Settings" button to save his choice.

The bottom part is for the user to view his database. Choose one table and click on the button, the chosen table will be shown in a new page.

C.4 Databases Page

Here's the page of the list of the databases.



Databases

View | Edit

Submitted by admin on Sun, 07/22/2012 - 12:09

databases list	
Show chosen database	☐ Show thumbnail (Warning: this may slow down your computer!)
●	Core Database - pic - It contains all the information of the images.
○	Core Database - picblack - It contains the url of images in black list.
○	Core Database - picwhite - It contains the url of images in white list.
○	Core Database - blackcount - It contains the domain information of the url in black list.
○	Core Database - whitecount - It contains the domain information of the url in white list.
○	usertable-User No.1-pic
○	usertable-User No.1-picblack
○	usertable-User No.1-picwhite

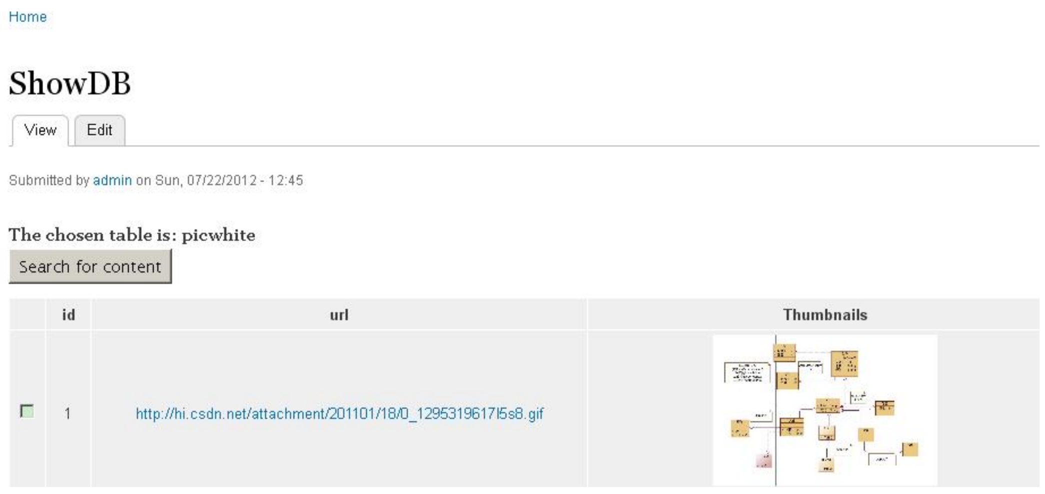
Figure 27. Databases page

In this page, the core database and the databases that are shared by the users are listed. Choose one of the tables in the list and click on the "Show chosen database" button, a new page will appear and show the user the corresponding content.

There is a setting of "Show thumbnail". When it's chosen and the database to be shown contains the URLs of images, the thumbnail of an image will be shown when the mouse moves over the corresponding URL. However, this function may slow down the user's computer.

C.5 Database-Showing Page

Here's the screenshot of the database-showing page.



Home

ShowDB

View | Edit

Submitted by admin on Sun, 07/22/2012 - 12:45

The chosen table is: picwhite

Search for content

	id	url	Thumbnails
☐	1	http://hi.csdn.net/attachment/201101/18/0_1295319617l5s8.gif	

Figure 28. Database-showing page

This page does not provide a direct link for the users. It can only be accessed via the "Databases" page or the "My Settings" page by selecting a database to show. It shows the content of the chosen database. If the database contains the URLs of images, the thumbnail will be shown to the user. In the meantime, the user can choose by clicking on the checkboxes in front of the items and clicking the “Search for content” button, the information(classes, attributes, operations and associations) of the selected images will be shown to the user.

Appendix D. User Manual for XMI2DB

This program aims at transforming XML files with models stored in to SQL files that contain "Insert" commands for our XMI database.
Here's the screen shot of this program.

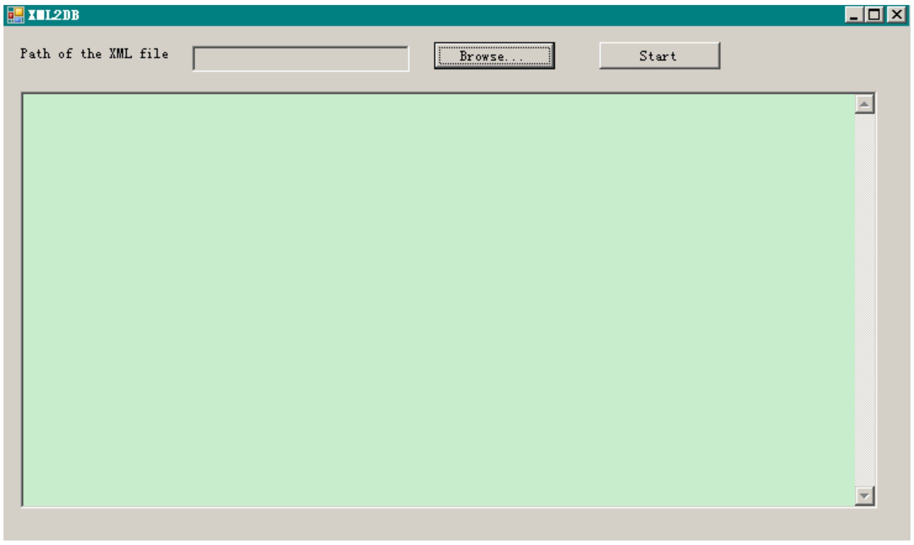


Figure 29. Initial interface of XML2DB

Step 1. Open the XML file.

The user should click on the "Browse…" button and select the XML file. The path of the file will be shown in the textbox near the button.

Step 2. Start the transformation.

By clicking the "Start" button, the program will abstract the classes, attributes, operations and associations from the XML files. In the end of the process, a message will pop up to inform the user, with the path of the SQL file created. The SQL file is named after the current data and time to avoid the problem of duplicated file name.

Step 3. Result of the process.

The SQL commands generated during the process will be shown in the textbox at the bottom part. It's a friendly interface for the user to check the elements detected and transformed in the XML file.

Appendix E. User Manual for XMI Query Website

E.1 Upload Page for SQL Files

Here's the screenshot of the upload page for SQL files generated by the XML2DB program.

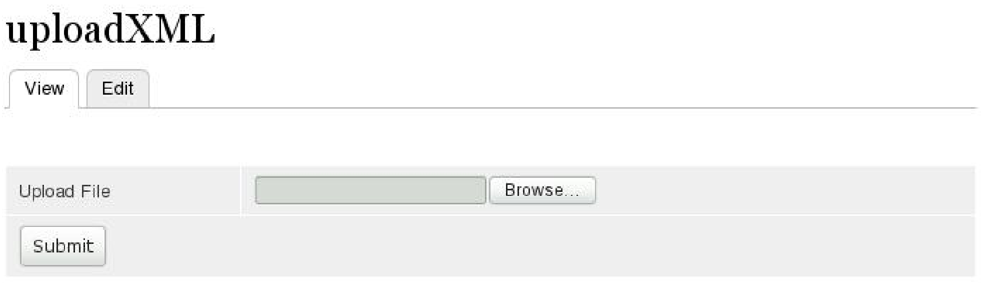
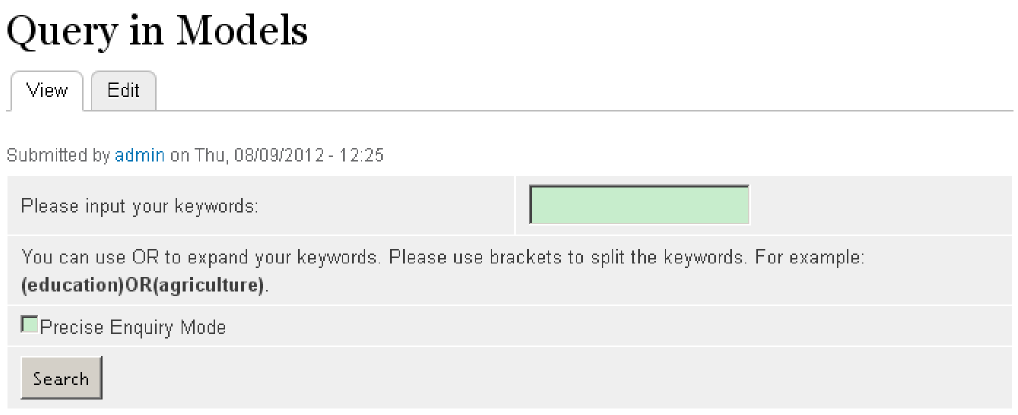


Figure 30. Upload page for SQL files

It's a very easy function. The user just click on the "Browse…" button, choose the SQL file generated and click the "Submit" button. The SQL commands in the uploaded file will be executed.
For safety concern, only the administrators have the access to this page.

E.2 Query in Models Page

This page provides the search function of the models in our XMI database.



Submitted by admin on Thu, 08/09/2012 - 12:25

Please input your keywords:

You can use OR to expand your keywords. Please use brackets to split the keywords. For example: **(education)OR(agriculture)**.

Precise Enquiry Mode

Figure 31. Query in models page

To use this function, the user just input the keywords in the textbox, and click "Search" button. The result will be shown in the same page with the URL of the images corresponding to the models found. The user can view the images instead of text to have a better impression of the models.
The keywords can be separated by spaces or connected by the "OR" keyword. Keywords

separated by spaces will be regarded as the relationship of “AND”. All the keywords should be matched in the result. Keywords connected by the “OR” function will be matched at least one of them in the result.
If the checkbox “Precise Enquiry Mode” is selected, the keywords will be precisely matched in the result. Otherwise, the result will include items that partly contain the keywords required.

E.3 Query of URL Page

When a user chooses to search several images in the list of URL, this page shows all the information stored in the XMI database of the images. Here’s the screenshot of this page.

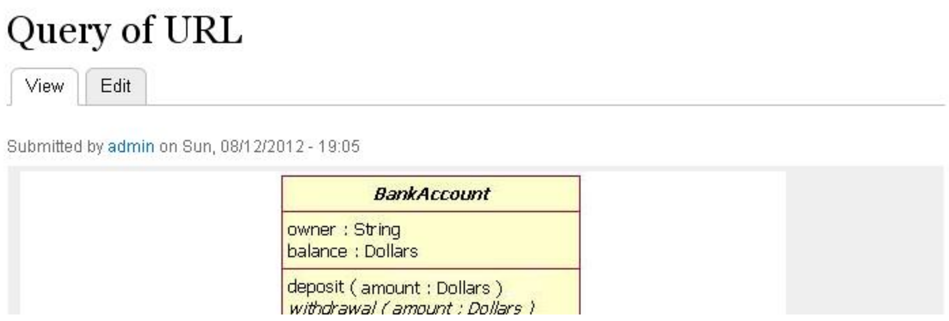

Query of URL
View
Edit
Submitted by admin on Sun, 08/12/2012 - 19:05
BankAccount
owner : String
balance : Dollars
deposit (amount : Dollars)
withdrawal (amount : Dollars)

Figure 32. Query of URL page

First, the image will be shown to the user. Then, under each image, four tables will be listed to show the classes, attributes, operations and associations included in this image.

Appendix F. Statistics of UML Class Diagrams Collection

F.1 Table “pic”

It is the table that contains the full information of the images downloaded. There are 2341 items in this table, with the width, height, pixel format and other attributes included.

F.2 Table “picwhite”

It is the table that contains the URL of the images that are considered UML class diagrams. 1394 items are included in this table.

F.3 Table “picblack”

It is the table that contains the URL of the images that are considered not UML class diagrams. 947 items are included in this table.

F.4 Table “whitecount”

It is the table that contains the statistics of domains where the images in the table “picwhite” come from. There are 715 different domains included in this table.

F.5 Table “blackcount”

It is the table that contains the statistics of domains where the images in the table “picblack” come from. There are 417 different domains included in this table.

Acknowledgements

Hereby I would like to thank my supervisors, Dr. Michel Chaudron and Mr. Bilal Karasneh, for their guidance and support of my thesis. Without their help, it would have been rather hard for me to go through the final step of my master program.
A special thank to Leiden University and the Netherland where I have found a platform to finish my master program, during which I have made an improvement not only in knowledge but also in the way of thinking.
I also want to express my deepest gratitude for my family. It's hard for one to study and live alone in a foreign country with a different culture. Their support and encouragement far from China has helped me to go through the toughest period of my thesis.

References

[1]http:// www.uml.org , “Object Management Group - UML”
[2] Frakes, W.B. , Pole, T.P. , “An empirical study of representation methods for reusable software components”, Software Engineering, IEEE Transactions, 1994
[3] Scott Henninger, “ An evolutionary approach to constructing effective software reuse repositories”, ACM Transactions on Software Engineering and Methodology (TOSEM), Volume 6 Issue 2, April 1997, Pages 111 - 140
[4] Mascena, J.C.C.P., de Lemos Meira, S.R., de Almeida, E.S.,Cardoso Garcia, V., “Towards an effective integrated reuse environment”, 5Th ACM International Conference on Generative Programming and Component Engineering (GPCE), short paper, Portland, Oregon, USA, 2006
[5] Vanderlei, T.A., Garcia, V.C., de Almeida, E.S., de Lemos Meira, S.R. "Folksonomy in a software component search engine coop-erative classi cation through shared metadata", Brazilian Symposiumon Software Engineering, Tool Session, Florianopolis, Brazil, 2006
[6] Brian Pinkerton, “Webcrawler : Finding what people want”, PhD thesis, University of Washington, 2000
[3] Xu Yuanchao, Liu Jianghua, Liu Lizhen, Guan Yong, “Design and Implementation of Spider on Web-based Full-text Search Engine”, Control and Automation Publication, Vol. 23, 2007
[8] https://developers.google.com/image-search/v1/devguide, “Image Search Developer's Guide of Google”
[9] http://support.google.com/images/bin/answer.py?hl=en&answer=1325808, Features of Google’s search by image
[10] http://support.google.com/websearch/bin/answer.py?hl=en&answer=2409866, Details of personal results by Google
[11] http://googleimagedownloader.com/, Google Image Downloader, the tool to download images with Google API
[12] Zhou Lizhu, Lin Ling, “Survey on the research of focused crawling technique”, Computer Applications, Vol. 25, No. 9, 2005
[13] Marc Ehrig, Alexander Maedche, “Ontology-focused crawling of Web documents”, Proceedings of the 2003 ACM symposium on Applied computing, 2003
[14] Zhou Demao, Li Zhoujun, “Survey of High-performance Web Crawler”, Computer Science, Vol. 36, No. 8, 2009
[15] Liu Jinhong, Lu Yuliang, “Survey on topic-focused Web Crawler”, Application Research of Computers, Vol. 24, No. 10, 2007
[16] Hsin-Tsang Lee, Derek Leonard, Xiaoming Wang, Dmitri Loguinov, “IRLbot: scaling to 6 billion pages and beyond”, Proceedings of the 17th international conference on World Wide Web, 2008
[17] Niu Xiamu, Jiao Yuhua, “An Overview of Perceptual Hashing”, ACTA ELECTRONICA SINCA, Vol. 36, No. 7, 2008
[18] Schaathun, H.G., “On Watermarking/Fingerprinting for Copyright Protection”, Innovative Computing, Information and Control, 2006.
[19] Monga V., Evans B.L., “Perceptual Image Hashing Via Feature Points: Performance Evaluation and Tradeoffs”, Image Processing, IEEE, 2006
[20] Zhao Yuxin, “A research on hashing algorithm for multi-media application”, PhD thesis, Nanjing University of Technology, 2009
[21] Bai He, Tang Dibin, Wang Jinlin, “Research and Implementation of Distributed and Multi-topic Web Crawler System”, Computer Engineering, Vol. 35, No. 19, 2009
[22] Wu Xiaohui, “An improvement of filtering duplicated URL of distributed web crawler”, Journal of Pingdingshan University, Vol. 24, No. 5, 2009
[23] http://drupal.nl/over-drupal, “Over Drupal”
[24] http://www.w3.org/TR/xml11/, “The specification of XML 1.1”

[25] http://www.w3.org/TR/xmlschema11-1/, "W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures"
[26] http://www.omg.org/spec/XMI/2.4.1/, "Official documents for XMI specification"
[27] Xiao Jihai, "A research of the transformation from XML Schema to entity relationship database", Master thesis, Taiyuan University of Technology, 2006
[28] Philip Bohannon, Juliana Freire, Prasan Roy, Jerome Simeon, "From XML Schema to Relations: A Cost-Based Approach to XML Storage", Data Engineering, 2002
[29] Ye Kaizhen, "The Study of XML Storage Technique in Relational Database", Master Thesis, SUN YAT-SEN UNIVERSITY, 2007
[30]Daniel Lucredio, Renata P. de M. Fortes, Jon Whittle, "MOOGLE: a metamodel-based model search engine", Softw Syst Model, 2012