МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ВОЛГОГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ОТЧЕТ

по производственной практике 2014г.

Студента

Фамилия\_\_Рогачев\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Имя\_\_Владислав\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчество\_\_Андреевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Факультет\_\_ФЭВТ\_\_\_\_\_\_\_ курс 3 группа \_ИВТ\_ – \_362\_

Тема дипломной работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

РУКОВОДИТЕЛЬ ПРАКТИКИ

Кафедра САПР и ПК Должность \_Профессор\_\_\_\_\_\_\_

Фамилия \_Кравец\_\_\_\_\_\_\_\_\_ Имя \_Алла\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчество \_Григорьевна\_\_\_\_

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

Волгоград 2014 г.

Оглавление

[1. Актуальность темы исследования 3](#_Toc394998435)

[1.1. Структура организации (подразделения) 3](#_Toc394998436)

[1.2. Состав решаемых задач. 4](#_Toc394998437)

[1.3. Состав автоматизированных задач 5](#_Toc394998438)

[2. Анализ существующих ПО и обзор методов 5](#_Toc394998439)

[3. Обзор методов 6](#_Toc394998440)

[Парсинг с использованием регулярных выражений 6](#_Toc394998441)

[Парсинг с использованием библиотеки BeautifulSoup и ей подобных 6](#_Toc394998442)

[Парсинг с использованием движка Webkit на языке Javascript 7](#_Toc394998443)

[4. Постановка задачи 8](#_Toc394998444)

[4.1. Потребность в автоматизации (задачи) 8](#_Toc394998445)

[4.2. Требования к видам обеспечения 8](#_Toc394998446)

[4.3. Наименование разработки 8](#_Toc394998447)

[4.4. Цель и назначение работы 9](#_Toc394998448)

[4.5. Задачи работы 9](#_Toc394998449)

[4.6. Решение задач 9](#_Toc394998450)

[5. Выводы, результаты и перспективы работы 16](#_Toc394998451)

1. Актуальность темы исследования

1.1. Структура организации (подразделения)

В 1985 году кафедра «Системы автоматизированного проектирования и поискового конструирования» (САПР и ПК) была выделена из состава кафедры вычислительной техники.

С 1991 г. и по настоящее время кафедрой руководит заслуженный деятель науки РФ, доктор технических наук, профессор Валерий Анатольевич Камаев.

На кафедре созданы следующие структурные подразделения, осуществляющие платное обучение: компьютерный центр, центр дополнительного образования детей (детская школа-студия компьютерной анимации), второе высшее образование.

К видам профессиональной деятельности выпускников кафедры относятся исследование, проектирование, производство и эксплуатация автоматизированных систем и программных комплексов в различных сферах науки, техники и бизнеса, поиск методов решения сложных задач. Кафедра является основоположником в высшей школе России учебных дисциплин, связанных с изучением методов технического творчества. За время своего существования кафедра выпустила более 800 специалистов, работающих во многих крупнейших компаниях страны. 85 из них получили дипломы с отличием, 34 отмечены медалями, дипломами и премиями Международных, Всесоюзных и Российских конкурсов.

## 1.2. Состав решаемых задач.

В настоящее время на кафедре подготовится к созданию системы «E-patent Examiner».

Основными целями создания системы являются:

* обеспечение автоматического позиционирования заявочных материалов на получение патента на изобретение в глобальном патентном пространстве на основе статистико-семантических подходов;
* оптимизация технологических процессов обработки материалов связанных с проведением экспертизы изобретений на основе результатов автоматического позиционирования заявочных материалов на получение патента на изобретение в глобальном патентном пространстве на основе статистико-семантических подходов.

Задачи, которые необходимо решить для достижения целей:

1. Провести анализ предметной области и разработать структуру глобального патентного пространства.
2. Оценить варианты позиционирования Заявки в глобальном патентном пространстве.
3. На основе сравнительного анализа выявить требования к статистико-семантическим методам и моделям.
4. Разработать архитектуру и требования к системе «E-patent Examiner».
5. Провести эксперименты для оценки эффективности разработанной концепции.
6. Разработать макет системы с применением «облачных» технологий и методов параллельных вычислений.

## 

## 1.3. Состав автоматизированных задач

В ряде задач для создания системы имеется задача: парсинг сайта Google Scholar (Академия Гугл), сервиса для поиска научной литературы, и получение текстовой информации. Цель задачи заключается в создании парсера сайта, т.е. программы, которая на вход бы принимала произвольный текст запроса, а на выходе выдавала бы его текстовую информацию и записывала ее в базу данных.

В рамке работы создана программа на C# для парсинга Google Scholar, получения и записи в базу данных и файл текстовой информации.

Тема практики – получение информации о научной литературе.

1. Анализ существующих ПО и обзор методов

Для реализации задачи были рассмотрены следующие ПО:

1. Библиотека для Python “scholar.py”;
2. Библиотека для C# “HTML Agility Pack” для парсинга web-страниц.

Google Scholar API для работы с сервисом не предусмотрены корпорацией Google.

Анализ:

Библиотека на Python не была выбрана из-за ее недоработанности. Из-за отсутствия стандартного API от компании Google была выбрана библиотека для языка C# под названием “HTML Agility Pack”, которая является самой популярной, удобной и обладающей большим функционалом библиотекой

1. Обзор методов

### Парсинг с использованием регулярных выражений

Регулярное выражение (regular expression, regexp, регэксп) - механизм, позволяющий задать шаблон для строки и осуществить поиск данных, соответствующих этому шаблону в заданном тексте. Кроме того, дополнительные функции по работе с regexp'ами позволяют получить найденные данные в виде массива строк, произвести замену в тексте по шаблону, разбиение строки по шаблону и т. п. Однако главной их функцией, на которой основаны все остальные, является именно функция поиска в тексте данных, соответствующих шаблону, описанному в синтаксисе регулярных выражений.

Где используются регулярные выражения? Сейчас регулярные выражения используются многими текстовыми редакторами и утилитами для поиска и изменения текста на основе выбранных правил. Многие языки программирования уже поддерживают регулярные выражения для работы со строками. Например, Perl и Tcl имеют встроенный в их синтаксис механизм обработки регулярных выражений. Набор утилит (включая редактор sed и фильтр grep), поставляемых в дистрибутивах Unix, одним из первых способствовал популяризации понятия регулярных выражений. Регулярные выражения можно использовать при парсинге сайтов, особенно, если нужно извлечь и проанализировать конкретную информацию статической HTML страницы. Такой страницей может служить какой-либо отчёт, размер которого превышает 10Мб.

### Парсинг с использованием библиотеки BeautifulSoup и ей подобных

Этот вид парсинга основан на построении полного дерева синтаксического разбора (DOM) в оперативной памяти и удобной навигации по нему с помощью встроенных в библиотеку функций. Так как на построение полного дерева уходит достаточно большой объем оперативной памяти и ресурсов компьютера, то в сравнении с предыдущим методом требует больше ресурсов, но позволяет более точно задать критерии поиска необходимой информации, а также предоставляет удобный интерфейс для программиста.

### Парсинг с использованием движка Webkit на языке Javascript

Парсинг с помощью движка Webkit также включает в себя и предыдущий вид парсинга, т.е. построения полного дерева DOM в оперативной памяти. Типичным представителем данного вида парсинга является PhantomJS. Фактически, это полноценный браузер в консоли, позволяющий с помощью встроенного языка Javascript взаимодействовать с Web-страницами, а также с ее интерактивными элементами, отслеживать данные, загружаемые на вебстраницу с помощью технологии AJAX, а также контролировать выполнение скриптов. Это снимает практически все ограничения по получению информации с сайта для систем автоматизации.

Парсинг с использованием HTML Agility Pack

Это парсер HTML, который строит доступный для чтения / записи DOM и поддерживает простой XPATH или XSLT. Это .NET библиотека, которая позволяет работать с HTML файлами «вне сети». Анализатор терпим к «реальному» неправильному HTML. Объектная модель очень похожа на ту, что предлагает System.Xml, но для HTML документов (или потоков).

XPath (XML Path Language) — язык запросов к элементам XML-документа. Разработан для организации доступа к частям документа XML в файлах трансформации XSLT и является стандартом консорциума W3C. XPath призван реализовать навигацию по DOM в XML. В XPath используется компактный синтаксис, отличный от принятого в XML.

1. Постановка задачи

## 4.1. Потребность в автоматизации (задачи)

Имеется сайт поиска научной литературы Google Scholar. Необходимо сделать запрос, распарсить и обработать полученную текстовую информацию, записать ее в базу данных и файл.

4.2. Требования к видам обеспечения

Для поддержания работы модуля визуализации должна быть использована операционная система Linux (Ubuntu 13.10) или Windows 7/8. Других специальных системных средств и систем программа использовать не должна. При запуске программы под ОС, не входящие в вышеперечисленные, разработчик не несет ответственности за корректную работу разрабатываемого изделия.

Язык разработки программы - C#.

Среда разработки - Microsoft Visual Studio в OS Windows.

Для реализации системы будут использована готовая C#-библиотека HTML Agility Pack для парсинга web-страницы.

4.3. Наименование разработки

Разработке подлежит программный модуль для парсинга определенного сайта. Полное наименование продукта - «Поиск научной информации». Краткое наименование - «ПНИ».

4.4. Цель и назначение работы

Программы создаются для парсинга сайта. Результаты программ будут использованы в системе кафедры САПРиПК «E-patent Examiner».

4.5. Задачи работы

Для реализации программ парсинга сайта должны выполнять следующие задачи:

- разработка алгоритмов составления и отправки запроса;

- разработка алгоритмов для вытаскивания тестовой информации из DOOM-объектов;

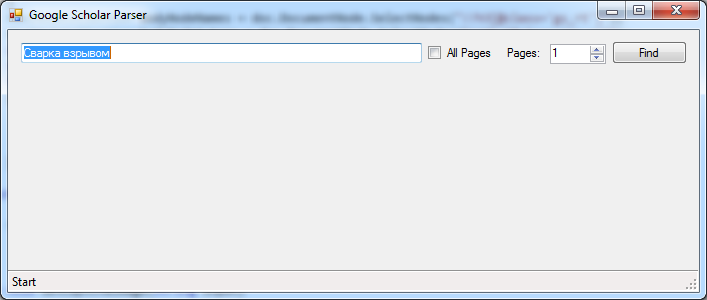
- разработка алгоритмов для обработки нужной информации;

- разработка алгоритма записи в базу данных.

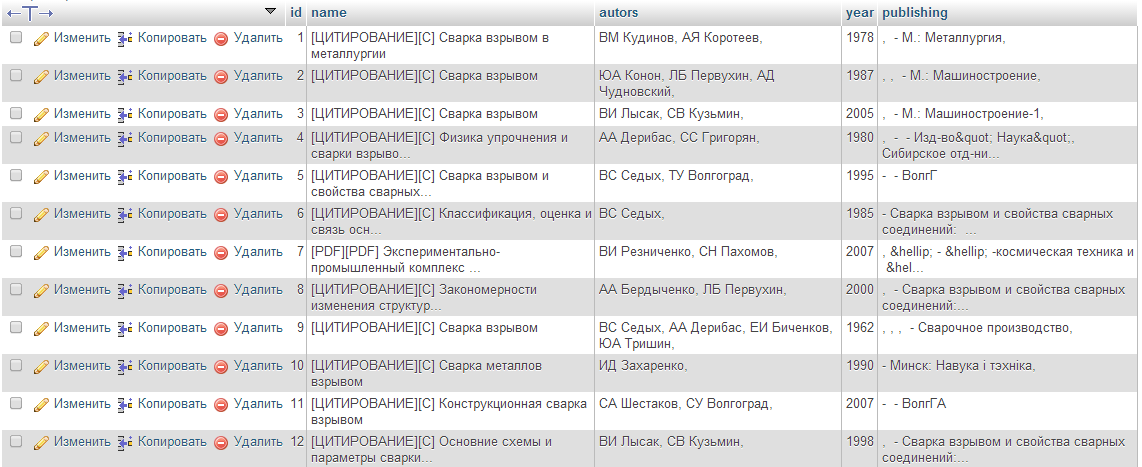
- разработка алгоритма записи в файл.

4.6. Решение задач

**Входные данные**



**Выходные данные** – база данных после записи.



**Код программы**

MainForm:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading;

using System.Windows.Forms;

using System.Net;

using System.IO;

using HtmlAgilityPack;

using HtmlDocument = HtmlAgilityPack.HtmlDocument;

using MySql.Data.MySqlClient;

using System.Configuration;

namespace GoogleScholarParser

{

public partial class MainForm : Form

{

private const string url = "https://scholar.google.com";

private int index;

private int count;

public MainForm()

{

InitializeComponent();

}

private void buttonFind\_Click(object sender, EventArgs e)

{

toolStripStatusLabelResponse.Text = "";

Thread threadParse = new Thread(Parse);

threadParse.Start();

}

private void checkBoxCount\_CheckedChanged(object sender, EventArgs e)

{

if (checkBoxCount.Checked)

{

labelCount.Enabled = false;

numericUpDownCount.Enabled = false;

}

else

{

labelCount.Enabled = true;

numericUpDownCount.Enabled = true;

}

}

private void toolStripMenuItemDatabaseSettings\_Click(object sender, EventArgs e)

{

DatabaseSettingForm databaseSettingForm = new DatabaseSettingForm();

databaseSettingForm.ShowDialog();

}

private void Parse()

{

File.Delete("log.txt");

File.Delete("log.html");

index = 0;

count = 0;

do

{

try

{

HttpWebRequest request = HttpWebRequest.CreateHttp(GetRequest(index, textBoxRequest.Text));

request.Credentials = CredentialCache.DefaultCredentials;

//request.UserAgent = i.ToString();

request.ContentType = @"text/html; charset=windows-1251";

request.Headers.Add(HttpRequestHeader.AcceptLanguage, @"ru-RU,ru;q=0.9,en;q=0.8");

WebResponse response = request.GetResponse();

toolStripStatusLabelResponse.Text = "Success";

Stream dataStream = response.GetResponseStream();

StreamReader reader = new StreamReader(dataStream, Encoding.Default);

string responseFromServer = reader.ReadToEnd();

File.WriteAllText("log.html", responseFromServer, Encoding.Default);

ParseHtmlDocument(responseFromServer);

}

catch (WebException ex)

{

toolStripStatusLabelResponse.Text = "Error";

Stream dataStream = ex.Response.GetResponseStream();

StreamReader reader = new StreamReader(dataStream, Encoding.Default);

string responseFromServer = reader.ReadToEnd();

GetCaptchaImage(responseFromServer);

File.WriteAllText("log.html", responseFromServer, Encoding.Default);

}

index++;

count--;

}

while((count > 0 && checkBoxCount.Checked) || (count > 0 && index < numericUpDownCount.Value && !checkBoxCount.Checked));

}

private void ParseHtmlDocument(string input)

{

HtmlAgilityPack.HtmlDocument doc = new HtmlAgilityPack.HtmlDocument();

doc.LoadHtml(input);

try

{

if (count == 0)

{

HtmlNode bodyNodeCount = doc.DocumentNode.SelectSingleNode("//div[@id='gs\_ab\_md']");

string c = bodyNodeCount.InnerText;

string b = "";

int l = 0;

while (c[l] != '(')

{

if (Char.IsDigit(c[l]))

{

b += c[l];

}

l++;

}

count = int.Parse(b) / 10;

}

//запись

DataResults dataResults = new DataResults();

HtmlNodeCollection bodyNodeNames = doc.DocumentNode.SelectNodes("//h3[@class='gs\_rt']");

HtmlNodeCollection bodyNodeDatas = doc.DocumentNode.SelectNodes("//div[@class='gs\_a']");

string[] names = new string[bodyNodeNames.Count];

Data[] data = new Data[bodyNodeDatas.Count];

for (int i = 0; i < bodyNodeNames.Count; i++)

{

names[i] = bodyNodeNames[i].InnerText;

data[i] = dataResults.GetData(bodyNodeDatas[i].InnerText);

}

SaveData(names, data);

WriteData(names, data);

}

catch

{

File.WriteAllText("log.html", input, Encoding.Default);

}

}

private void GetCaptchaImage(string input)

{

HtmlAgilityPack.HtmlDocument doc = new HtmlAgilityPack.HtmlDocument();

doc.LoadHtml(input);

string captcha = "";

try

{

HtmlNode bodyNode = doc.DocumentNode.SelectSingleNode("//img");

CaptchaForm captchaForm = new CaptchaForm(url + bodyNode.Attributes["src"].Value);

captchaForm.ShowDialog();

captcha = GetRequest(index, textBoxRequest.Text) + "&captcha=" + captchaForm.Captcha + "&submit=Отправить";

}

catch

{

File.WriteAllText("log.html", input, Encoding.Default);

}

ParseCaptchaDocument(captcha);

}

private void ParseCaptchaDocument(string input)

{

try

{

HttpWebRequest request = HttpWebRequest.CreateHttp(input);

request.Credentials = CredentialCache.DefaultCredentials;

//request.UserAgent = i.ToString();

request.ContentType = @"text/html; charset=windows-1251";

request.Headers.Add(HttpRequestHeader.AcceptLanguage, @"ru-RU,ru;q=0.9,en;q=0.8");

WebResponse response = request.GetResponse();

toolStripStatusLabelResponse.Text = "Success";

Stream dataStream = response.GetResponseStream();

StreamReader reader = new StreamReader(dataStream, Encoding.Default);

string responseFromServer = reader.ReadToEnd();

File.WriteAllText("log.html", responseFromServer, Encoding.Default);

ParseHtmlDocument(responseFromServer);

}

catch (WebException ex)

{

toolStripStatusLabelResponse.Text = "Wrong Captcha";

Stream dataStream = ex.Response.GetResponseStream();

StreamReader reader = new StreamReader(dataStream, Encoding.Default);

string responseFromServer = reader.ReadToEnd();

File.WriteAllText("log.html", responseFromServer, Encoding.Default);

}

}

private string GetRequest(int numb, string input)

{

string[] arr = input.Split(' ');

string req = "";

for (int i = 0; i < arr.Length; i++)

{

req += arr[i];

req += "+";

}

req = req.Remove(req.Length - 1);

return url + "/scholar?start=" + (numb \* 10).ToString() + "&q=" + req + "&btnG";

}

private void SaveData(string[] names, Data[] data)

{

string conStr = "server=" + ConfigurationManager.AppSettings["server"] + ";user=" +

ConfigurationManager.AppSettings["user"] + ";database=" +

ConfigurationManager.AppSettings["database"] + ";password=" +

ConfigurationManager.AppSettings["password"] + ";";

using (MySqlConnection con = new MySqlConnection(conStr))

{

try

{

con.Open();

for (int i = 0; i < names.Length; i++)

{

string sqlResults = "INSERT INTO results\_table (id, name, autors, year, publishing) VALUES (null, @Name, @Autors, @Year, @Publishing);";

MySqlCommand cmdResults = new MySqlCommand(sqlResults, con);

//создаем параметры и добавляем их в коллекцию

cmdResults.Parameters.AddWithValue("@Name", names[i]);

cmdResults.Parameters.AddWithValue("@Autors", data[i].autors);

cmdResults.Parameters.AddWithValue("@Year", data[i].year);

cmdResults.Parameters.AddWithValue("@Publishing", data[i].publishing);

cmdResults.ExecuteNonQuery();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

private void WriteData(string[] names, Data[] data)

{

for (int i = 0; i < names.Length; i++)

{

File.AppendAllText("log.txt", names[i] + "\n" + data[i].autors + "\n" + data[i].year + "\n" + data[i].publishing + "\n\n");

}

}

}

}

CaptchaForm:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace GoogleScholarParser

{

public partial class CaptchaForm : Form

{

private string captcha;

public string Captcha

{

get { return captcha; }

}

public CaptchaForm(string input)

{

InitializeComponent();

pictureBoxCaptcha.Load(input);

captcha = "";

}

private void buttonOk\_Click(object sender, EventArgs e)

{

captcha = textBoxCaptcha.Text;

this.Close();

}

}

}

DatabaseSettingForm:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Configuration;

namespace GoogleScholarParser

{

public partial class DatabaseSettingForm : Form

{

public DatabaseSettingForm()

{

InitializeComponent();

textBoxServer.Text = ConfigurationManager.AppSettings["server"];

textBoxUser.Text = ConfigurationManager.AppSettings["user"];

textBoxDatabase.Text = ConfigurationManager.AppSettings["database"];

textBoxPassword.Text = ConfigurationManager.AppSettings["password"];

}

private void buttonSave\_Click(object sender, EventArgs e)

{

Configuration currentConfig = ConfigurationManager.OpenExeConfiguration(ConfigurationUserLevel.None);

currentConfig.AppSettings.Settings["server"].Value = textBoxServer.Text;

currentConfig.AppSettings.Settings["user"].Value = textBoxUser.Text;

currentConfig.AppSettings.Settings["database"].Value = textBoxDatabase.Text;

currentConfig.AppSettings.Settings["password"].Value = textBoxPassword.Text;

currentConfig.Save(ConfigurationSaveMode.Modified);

ConfigurationManager.RefreshSection("appSettings");

this.Close();

}

private void buttonCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. http://htmlagilitypack.codeplex.com/  
   Сайт разработчиков библиотеки HTML Agility Pack
2. http://msdn.microsoft.com/ru-ru/  
   Сеть разработчиков Майкрософт
3. Выводы, результаты и перспективы работы

В результате прохождения практики даются следующие выводы:

- участвовав в разработке проекта кафедры САПР и ПК, выполнил все поставленные задачи. В особенности разработаны программы для парсинга сайта Google Scholar;