Hong Kong Baptist University

COMP 4047 Internet and World Wide Web

DESIGN AND IMPLEMENTATION OF A SEARCH ENGINE

Group #8: Marco Ambrosini, Robert Herman, Jonathan Rempel

Authors: Marco Ambrosini, Robert Herman, Jonathan Rempel

Group: No. 8

Prof.: Y. W. Leung

Date: 01. November 2016

Table of Contents

[1 Introduction 3](#_Toc465447456)

[2 Design 4](#_Toc465447457)

[2.1 Components 4](#_Toc465447458)

[2.2 Communication 5](#_Toc465447459)

[2.3 Classes 5](#_Toc465447460)

[2.4 Data structures 7](#_Toc465447461)

[3 Implementation 8](#_Toc465447462)

[3.1 Configuration 8](#_Toc465447463)

[3.2 Installation 8](#_Toc465447464)

[4 Appendix 9](#_Toc465447465)

[4.1 List of Figures 9](#_Toc465447466)

[4.2 List of Tables 9](#_Toc465447467)

[Participation Form 10](#_Toc465447468)

# Introduction

The group project of the course COMP 4047 Internet and World Wide Web comprises the design and implementation of a search engine which crawls the web from a specified starting point storing the information in a way that allows the user to search for specific keywords or phrases.

This document is to describe the design and implantation of said search engine.

# Design

The search engine is a complex software which makes use of various components. This section shall provide insight into the functions of each single component including the communication between each other.

## Components

The following list shall provide an overview of the most relevant components and their core function.

* web-crawler.jar  
  This java application is the heart of the software managing both the crawling of the web as well as searching the stored information for the requested keyword or term. The program is written in java and the section 2.3 Classes provides more detailed information of the individual classes. The executable jar file is created using maven to ensure that all dependencies to external libraries are up-to-date. A second version ‘web-crawler-jar-with-dependencies.jar’ incorporating all these dependencies is also created, which is the preferred version to be executed.
* Maven  
  Maven is a software…
* pom.xml & web-crawler.iml  
  These two files contain all the information required by maven to create the executable jar-files. While ‘pom.xml’ lists all external libraries used throughout the java application, ‘web-crawler.iml’ provides the information on relative folder paths. When installing the executables using maven these dependencies and paths are resolved correctly allowing to install and run the application to and from any arbitrary directory.
* config.properties  
  This configuration file allows to easily alter some parameters used in the java application before installation.
* IgnoreList.csv  
  This text file contains a list with all words which shall be ignored while crawling the web. Changes to this list will be effective upon restarting the application.
* CrawlerStorage.mv.db  
  This h2database is a simple in-memory database which comes with a build-in java api. We consider this database fit for purpose since it is easy to implement and handle. The database is used to a) store all the information gathered by the crawler, and b) retrieve any relevant links when searching for a keyword or term. Section 2.4 Data structures describes the used tables and their relation in more detail.
* start-server.cmd  
  This command is used to launch the application and has no other function then calling the python script ‘server.py’ described below.
* server.py  
  This python script is executed directly after launching the application and instructs the java application to crawl the web using a special keyword. The keyword and also the duration of the crawling can be modified using the configuration file. After the java application has finished crawling ‘server.py’ sets up a webserver and opens ‘index.html’, the homepage of the search engine, in the default browser.
* search.py  
  This python script is called from ‘index.html’ when the user clicks the search button. The keyword or term is passed to the java application which then returns a new webpage with the found results.
* Results.ftl  
  The above mentioned new webpage is based on the template ‘Results.ftl’ which is dynamically amended with the search results by the java application.

## Communication

The below flow chart / Event driven Process Diagram (EPC) written in Business Process Model Notation (BPMN 2.0) illustrates the comprehensive process flow of this software.

(use Adonis to illustrate the communication process – data flow or BPMN?)

The starting point is the command ‘start-server.cmd’ which is launches ‘server.py’…..

The below data flow UML diagram shows flow of information managed by the java application.

(data flow)

## Classes

The java application ‘web-crawler.jar’ consists of various classes which are shown in the following class diagram and explained in more detail below.



Figure 1: Class Diagramm – Crawler Model

* Crawler.java makes it so that the search engine ignores the most commonly used English words, so to make the search results more accurate. It also manages new URLs and adds them to the URL pool for processing. Lastly, it stores processed words and metadata.
* Http.java sets the timeout of the search to 5 seconds. It retrieves the words on a page, along with their position and number of occurrences. Additionally, it also does some error checking and prints error messages accordingly.
* PageResult.java stores and returns words and URLs with ArrayLists.
* PageResultException.java (I’m not sure what this does)
* SearchServer.java handles the initial and maximum number of URLs as well as the URL pool size.
* Storage.java manages the searched terms and uses tables to acquire the most accurate results.
* TemplateResponse.java (I’m not very sure what this does either, sorry…)
* WordResult.java manages the word searched, and how many times the word is used in a webpage.

## Data structures

tbc

# Implementation

## Configuration

Prior to installation, various parameters of the crawler can be changed inside the configuration file ‘config.properties’. This allows any users to easily customize this software to match their preferences. They can change the…

* initial URL which the crawler shall use as starting point.
* max number of URLs which shall be crawled
* max poolsize?
* template which shall be used to display the search results
* amount of days after which the crawler regards a stored URL as outdated and will remove it from the database
* keyword which is used to instruct the java application to crawl the web rather than performing a search against the available database.

## Installation

1. Extract the two folders contained in group8.zip into any location.
2. Optional: update ‘config.properties’ in ‘/web-crawler/src/main/resources’ to your preferences
3. Make sure Maven is installed
4. Open a new cmd window and set the directory to be ‘/web-crawler’
5. Type and call the cmd ‘mvn clean install’

# Appendix

## List of Figures

Abbildung 1: Class Diagramm – Crawler Model 2

## List of Tables

**Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.**

Participation Form

**Group No.: 8**

**Date: 01/11/16**

**Group Members:**

|  |  |  |
| --- | --- | --- |
| Student ID | Student Name | Signature |
|  | Jonathan Rempel |  |
| 16502019 | Marco Ambrosini |  |
| 16502299 | Robert Herman |  |

**Participation:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Assessment Criteria | | Percentage participation | | | |
| Student 1 | Student 2 | Student 3 | Total |
| Design | Design and data structures (20 marks) |  |  |  | 100% |
| Implementation | Gathering information (35 marks) |  |  |  | 100% |
| Web server setup (5 marks) |  |  |  | 100% |
| Serving requests (20 marks) |  |  |  | 100% |
| Documentation | Report (15 marks) |  |  |  | 100% |
| Comments in source codes (5 marks) |  |  |  | 100% |
| Supplementary information: | | | | | |