CS/SE 6359 – Object-Oriented Analysis & Design Summer 2023

Project Phase II: Cyberminer – A web search engine –

MUST: Class, Use Case, and Sequence Diagrams + State Transition + Activity Diagrams

Due Date: July 13(Thursday)– Interim Project II submission
August 1 (Tuesday)– Final Project I submission & Presentation

I. Summary

As system/software engineers of a renowned company, your team is to develop a simple web search engine, here called *Cyberminer*. For this project, you will use an Object-Oriented Analysis and Design, and build an Object-Oriented Program, which should be accessible through your team's web page or run on your laptop.

II. The Cyberminer System

II.1 Functional Requirements

Cyberminer shall accept a list of keywords and return a list of URLs whose descriptions contain any of the given keywords.

Cyberminer shall check each URL is syntactically correct, in the following spirit:

URL ::= {['http://'|'https://']} {www.} Identifier '.' ['edu' | 'com' | 'org' | 'net'|'gov']

identifier ::= {letter | digit}$^{+}$

letter ::= ['a' | 'b' | … | 'y' | 'z' | 'A' | 'B' | … | 'Y' | 'Z']

digit ::= ['1' | '2' | … | '9' | '0']

Cyberminer shall allow for:

- *Case sensitive search*: The system shall store the input as given and retrieve the input also as such;
- *Hyperlink enforcement*: When the user clicks on the URL, which has been retrieved as the result of a query, the system shall take the user to the corresponding web site;

- *Specifying OR/AND/NOT Search*: A keyword-based search is usually an OR search, i.e., a search on any of the keywords given. The system shall allow the user to specify the mode of search, using “OR”, “AND” or “NOT”;
- *Multiple search engines*: to run concurrently;
- *Deletion of out*-of-date URL: and corresponding description from the database;
- *Listing of the query result in ascending alphabetical order*; *most f*requently *accessed order, or per payment,*
- *Setting the number of results to show per page, and navigation between pages*;
- *Autofill*, while correcting typographical errors,
- Filtering out symbols that are not meaningful, according to the user configuration.

II.2 Non-Functional Requirements

```
Cyberminer shall be easily understandable, portable, enhanceable and reusable
with good performance. The KWIC system shall also be user-friendly,
responsive, and adaptable.
```

III. The Deliverable

Your description should be elegant and comprehensible. Your deliverable should be available as both on-line (one URL per team member) and offline specifications (submission of one copy per team). You can choose to use an (extended) IEEE-style format for the deliverable, in which the major sections typically include: Introduction, Main Body (items below, for this project), Glossary (Definitions and Acronyms) and References (See, for example, "Document Templates - general IEEE" on the Internet or course web site).
N.B. Your team’s role play should be described in the updated Project Plan, among other things.

1. Process

Describe the process your team takes and the artifacts your team develops for every phase/step of the process. Your team should use the Unified Process (UP) as a basis, as suggest by C. Larman or a reasonably good variant of UP. Since your project involves implementation and testing, the UP diagram should be extended accordingly. Also, for Phase II, your team should address both Software Architectural Design and Component/Low-Level Design.

Describe both pictorially and textually. Your team’s deliverable should establish traceability among all the phases of UP, from your domain to the requirements, and to the (architectural and component) design specification (including design patterns), to implementation and testing.

Additional specifics will be discussed in class.

2. Domain Modeling & Requirements specification

Your team should consider who the stakeholders of the system are, whether different kinds of users (e.g., organizational vs individual users, advertisers, and Cyberminer adminstrators) would need different kinds of features, etc.

The functional requirements specification is incomplete (e.g., where should the input come from, and the output go?). What and how inputs and outputs should be handled? Describe any extensions, or clarifications, to the requirements specification.

The non functional requirements specification is also ambiguous. Clarify each non-functional term repeatedly as many times as you'd see necessary.

3. Design specification

3.1 Architectural Design specification

Per class discussions, you'd need to consider, among other things, using the MVC architecture.

3.2 Component Design specification

Per class discussions, you'd need to consider using some design patterns, such as the Singleton, Façade and Observer design patterns, for realizing the MVC architecture.

4. A Prototype implementation

Your program specification of a prototype implementation, well documented and tested (should include test cases with test data).

5. User Manual

A (preliminary) user manual should be developed, which should become more complete and consistent at the end of the 2nd phase of the project. Describe how the user can access and use the system, including the URL of each team's web site where your applet, or its equivalent (and all other deliverables), can be accessed. Also briefly describe essential scenarios --- the typical interactions between the user and the system, e.g., what are the steps the user has to follow in using the system. Use screenshots to show how the system looks like initially as well as to show subsequent steps that the user might take.