CS/SE 6362 – Advanced Software Architecture and Design Summer 2011

Project Phase II: Cyberminer Software Architecture

Due Date:- July 14 (Thursday) – Interim Project II submission
July 26/July 28 (Tuesday/Thursday) – Final Project II submission and presentation

> ***All the other arts were obedient and submitted to the discipline of architecture.*** – *The Art of Systems Architecting.* Victor Hugo

I. Summary

As system/software architects of a renowned company, your team is to architect a web search engine, *Cyberminer*, using the simple KWIC software system which you implemented as part of Project I. For this part of the project, you will use an Object-Oriented architectural style, and build a Java applet (or an equivalent), which should be accessible through your team's web page.

II. Cyberminer – A growing web search engine

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 use another software system, the KWIC system, as a component, in order to efficiently maintain a database of URLs and the corresponding descriptions.

The KWIC system shall accept an ordered set of lines, where each line consists of two parts:

- *The URL part*, whose syntax is:

  URL ::= 'http://' identifier '.' Identifier '.' ['edu' | 'com' | 'org' | 'net']

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

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

  digit ::= ['1' | '2' | … | '9' | '0']

- *The descriptor part*, whose syntax is:

  identifier {' ' identifier}*

The descriptor part of any line shall be “circularly shifted” by repeatedly removing the first word and appending it at the end of the line. The KWIC index system shall output a listing of all circular shifts of the descriptor parts of all lines in ascending alphabetical order, together with their corresponding URLs. No line in the output list shall start with any noise work such as “a”, “the”, and “of”.

The KWIC system shall allow for two modes of operation: i) for building an initial KWIC indices; and ii) for growing the indices with later additions.

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*;
- *Setting the number of results to show per page, and navigation between pages*; and also possibly
- *Autofill*.

Note that if Cyberminer uses the Java Event Model, one of the above features can be omitted.

II.2 Non-Functional Requirements

Cyberminer shall be easily understandable, portable, enhanceable and reusable with good performance. Cyberminer shall also be user-friendly, responsive, and adaptable. Also, Cyberminer shall run with recent versions of popular web browsers.

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. Requirements specification

The functional requirements specification is incomplete (e.g., where should the input come from, and the output go?). 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.

2. Architectural specification

Describe both pictorially and textually the constituents of the architecture, i.e., the architectural style, components and connections. Your deliverable should also discuss the rationale in terms of the advantages and disadvantages of your architecture, in consideration of several scenarios whenever appropriate. Also describe all the constraints and patterns, if any. You should establish traceability between the requirements specification and the architectural design specification.

3. Specification of a Java applet or an equivalent, and a prototype implementation

Your program specification, well documented and tested, and an implementation of a prototype system.

4. 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.