CS/SE 4352.501 — System/Software Architecture
Fall 2005

Project II: A Web Search Engine

Due: 11:00am Thursday October 27 (hardcopy in class; softcopy online)

The design of complex systems must blend the art of architecture with the science of engineering.

— "The Art of Systems Architecting".

I. Summary

As system/software architects of a renowned company, your team is to architect a web search engine, **Cyberminer**, using **KWIC** which you implemented as part of Project I. For this part of the project, you will continue to use an Object-Oriented architectural style in building a Java applet (or an equivalent), which should be accessible through your own web page (Each member of the team should maintain a course webpage).

II. Cyberminer - A web search engine

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, **KWIC** - a KWIC (Key Word in Context) index system, in order to efficiently maintain a database of URLs and the corresponding descriptions.

KWIC 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 list of all circular shifts of the descriptor parts of all lines in alphabetically ascending order, together with their corresponding URLs. No line in the output list shall start with any noise word such as "a", "the", and "of".

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

Non-Functional Requirements:

Cyberminer shall be easily understandable, portable, enhanceable and reusable with good performance. **Cyberminer** 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 off-line specifications (submission of one copy per team). You can choose to use an 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 course web site).

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 ambiguous. Clarify each non-functional term repeatedly as many times as you'd see necessary.

2. **Architecture** Describe both pictorially and textually, 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 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)** Your program specification, well documented and tested.

4. **User Manual** Describe how the user can access and use the system. Your description should include the addresses of each team member's web site where your applet (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 for subsequent steps that the user takes.