Software Architecture Project 3 Documentation
Improved Web Search Engine

Group 20 - Jason Switzer, Cliff Halasz

1. Process Architecture

Our team met regularly for brief meetings after lectures to perform overall architecture design and to break down task responsibilities, and then the google talk protocol was used to communicate on a day-to-day basis. The main task with this project was to break out the previous project into a client and server architecture. We debated different architectures and came to an agreement on using J2EE as neither of us had experienced it before and we desired to attempt something new. Individually we then researched the platform and then worked in parallel to implement it. We then worked closely together to test the system and finally write this documentation.

2. Architecture Specification

Our previous, single machine implementation used a heavily object-oriented approach. This was very beneficial in applying the client / server pattern; we already had self-regulating data structures and all the shifting and sorting in separate methods. Using the J2EE platform for implementing client server meant that all we had to do was wrap the functions we already had and then provide an html interface for both the 'client' and the 'server', both of which act as clients upon the J2EE server. The 'client' is run remotely and connects via browser to the server, and then makes a call through J2EE and the server methods are invoked. Similarly the 'server' is just a local client with permissions to also use server methods for modifying data. All of the data is persisted on a database

Our designs are forward-thinking, by using an object-oriented design before it made refactoring to the client and server easier than a more embedded system would have been. This project didn't require much new real functionality from the previous, most was merely moving previous functionality to either the client or the server interfaces. The largest new requirement was persistence in data, this was achieved by tying the data storage to a database on the server, because we already had a client / server architecture, this did not introduce any new dependency, and again is a forward-thinking design allowing for more functionality to be added in the future. Removal of data for instance would be very easy to add, we would just have to add a method to search and remove from the database, and then just add the removal function to the client or server interfaces.

Our architecture primarily uses the repository design pattern, where both the client and server are in reality both thin clients utilizing a server which performs all computations and stores all data in a database that it manages. This keeps all data managed and controlled,

and allows for a very large number of clients to work efficiently with the same data. For this project, in deployment one would expect a very very large table of URLs to manage, so the repository is a strong approach. Data is held in IndexedString objects which are sent to a CircularShifter which acts as both a factory building the many shifted IndexedStrings off a single input, and then operating as an iterator which gives and interface for methods to access the IndexedStrings. The CircularShifter also uses an EnglishCollator as a strategy which provides one implementation of how to sort the strings; so alternative strategies could be implemented in the future trivially.

3. Program Specification

Our program is built on the J2EE platform. SearchController is a SimpleFormController servlet that holds a SearchService which performs the search remotely on the database. This utilizes the IndexedString object from the previous project, which manages the url and title strings, which are held in a CircularShifter just as before. An Admin page allows for adding entries to the database while a search page only allows the search service to be called. Once the service is running, any browser can be used to open the admin or search pages and then perform actions.

4. User Manual

To deploy the server, you must have a running java server. Upload the web.war file and then start it.

To manage the server, open a browser window and navigate to <localhost>:<port>/web/admin.htm. This will open the server window, which will present you with a form to add entries to the index. There is a box for entering a text of the form "<decriptor> <URL>" and when complete you may press the OK button to enter the data to the database. You will then be taken to a form showing the data entered, and you may view the resultant circular shifts in the database.

To use the client, navigate any browser to <server IP>:<port>/web/search.htm and you will be presented with a search box and an OK button. You may enter many lines of text in the box and a scroll bar will appear if neccesary, and then when finished press the OK button to perform the search. Text entered on the same line will be considered within quotations, or the search will be performed for strings containing all of those words; separate lines of text will be searched independently. You will then be taken to a page of results in alphabetical order where you can mouse over for the URL or click the title to link to the page.