Rob Johansen

u0531837

CS 4480 - PA 1 - Assignment Report

Design

My implementation of the caching proxy server for PA 1 is written in Java. It utilizes the ServerSocket class for accepting incoming connections, and a thread pool from ExecutorService for handling client requests in separate threads.

For caching, my proxy first creates a hash of the URI of each client request. These hashes are stored as keys in a shared, thread-safe ConcurrentMap instance. The value associated with each hash key is a unique Lock object. Thus if two clients simultaneously request the same URI, one client will acquire the lock associated with the request, while the other waits for the lock to become available. All new responses are written to disk in a directory (named after the hash) and used in response to subsequent matching requests.

One of my design tradeoffs was to detect whether each request is for text or an image. This allows my server to read/write text via character streams and images via byte streams.

My proxy does not implement the extra credit functionality.

Testing

I used three tools for testing during and after the development of my caching proxy server:

- telnet - This command-line approach to testing of both server and client was particularly useful for testing concurrency.

- IDE - With this approach I launched my server via my IDE so I could use breakpoints to pause and step through execution of my code. This allowed me to analyze the performance of my server at runtime, and helped me detect a few tricky bugs.

- Firefox 15 - I configured Firefox 15 to use my proxy server and HTTP 1.0. This helped me discover that my initial implementation was lacking support for images.

My server appears to work relatively well for most websites. However, I’ve noticed that Firefox 15 periodically sends empty requests and experiences broken pipes. I've also noticed infrequent cases where requests for extremely large HTML files don't succeed. Ironically, my server correctly returns the HTML for the simple page provided in the assignment (http://www.cs.utah.edu/~kobus/simple.html), however Firefox will not display it due to a compression/encoding issue.

Output

Below is sample output from client/server that shows the correct functioning of my proxy (including a response from the cache).

Server:

```
java WebProxyServer 6000
```

Client:

```
telnet localhost 6000
GET http://www.cs.utah.edu/~kobus/simple.html HTTP/1.0
```

Server:

```
No response for this request in cache: http://www.cs.utah.edu/~kobus/
simple.html
Writing response to cache: ./19b6e54e7bc8006a9cae6fccafb3aac8/cached
```

Client:

```
telnet localhost 6000
GET http://www.cs.utah.edu/~kobus/simple.html HTTP/1.0
```

Server:

```
Retrieving request from cache: http://www.cs.utah.edu/~kobus/simple.html
```