<u>Network Code Editor</u>

Ryan Yosua, Eric Panulla, Shane Eckenrode, Jose Ponte

Our main purpose with this project was to build a collaborative code editing application geared toward helping developers to collectively code their projects. Similar to the way in which users experience Google Docs, our application allows users to edit a file in real-time while also allowing them to send messages to each other using the chat window. Using the Java.net package for networking, our application allows users to choose an alias name and establish connections to a “room” based on an IP address. Once connected, the application’s multi-threaded interface allows users to concurrently edit code using the main editor window. All users may choose to save a copy of the file being worked on by using the save button, and a file that is already loaded is autosaved every 30 seconds. Any user may load a file into the editor that everyone else may edit.

Compared to our proposed application, it was necessary to adjust and adapt in a few areas to improve the application overall. We initially proposed that clients would be able to choose if they wanted to start or join a session. In the current design, each session has its own server, so we ask users to input a server IP to connect to upon starting the application. This implementation allows for different servers being able to be set up for multiple groups of users. User authentication through a database would have been more secure, but our application still has many use cases where authentication is not an issue including open source software development, and educational use. If a server is hosted behind a firewall on a private network, than authentication is also unnecessary. Also we simplified the idea of sharing entire directories and files among users in the application. Only one loaded file is shared at a time, but if the user loads a directory, they can easily switch between files by clicking on a file from the list on the left.

We initially wanted to use the Java database for data persistence but due to time we opted to use saving and loading of files directly to a users pc. This method of data persistence allows any user to load a file if they want or just start from a blank page. Once the file has been worked on, each user may save a copy of the file onto their own pc using the save button, and the data will be the same for every user as long as new changes aren’t made after clicking the save button. Because each session has it’s own server, the database of users did not add as much value.

Furthermore, we also managed to add an extra chat feature. Although not in the original proposal, a chat feature was added to to improve collaboration while editing a file. We saw fit that this kind of feature could be very crucial in an application where multiple users interact with a document. This implementation was based on material we learned through class exercises and added to increasing the usability of the application to developers.

The overall usability of the NCE (Network Code Editor), as originally proposed, was almost fully implemented throughout the application. The few features such as the “Room Login” and the “Directories Sharing” not included in our final application, were replaced by more efficient and time-saving features which are fully functional. Many of those features could be a goals for further development of our current working application.