Where's the File Readme
Philippe Clesca and Ronnie Jebara

USE LS.CS.RUTGERS.EDU LIKE BASICALLY ALL OTHER ILAB MACHINES WILL CAUSE SEG FAULTS AND WE SPENT 4 DAYS TRYING TO FIGURE OUT WHY AND WE COULDNT.

Compression

We were successfully able to make a system call to use tar to compress the files to make them easier to send across the socket. We were also able to use a system call to use tar to compress old project versions down to one file. We found using tar much easier then trying to make compression work with zlib.

Multithreaded

The server is set up to be multithreaded. It can accept essentially an endless amount of processes to do any command. We use a mutex lock though to stop the threads from trying to access the same items. We decided on one lock, that basically would block all other processes from accessing the ".repo" directory, which is where the server stores all the projects.

Project Storage

Like we said above the server stores all created projects in the ".repo" directory. Inside the projects is where it stores all the old version of the project as a tarred file. We did this for extra credit.

Network Protocol

We had a system of ways to send messages from the client to the server that made communication easier and shorter. For sending files over the socket, our networking protocol tarred the files all into one file to save on time and space. We also had short and easy commands to follow that the client would send to the server. Such as for "create" the client would send the server "mkdir". For "destroy" our client sends the server "rmdir".

Add Remove

Add and remove were quite tricky to get at first. We had to rebuild it several different times. We found ourself in a unique situation where no matter what we did we couldn't stop it from seg faulting at first. We fixed all our problems by building our our own string parser, string concatanator, and such. We found by having such control over these functions we were able to get everything t owork exactly the way we wanted to.

Checkout

Checkout was by far the hardest function to make. We had to spend a lot of time on it. We suffered the same problems that we did from add remove. We had to use a lot of the string functions we had to build and this is where we first started using tar to send files over the socket.

Testing

So we tested our code at first by testing one function at a time. We had really good results with that so we started to try fork and accessing the same resources at once. The locks caught that and it worked the way we expected. That's as far as the testing went. Since we have one lock that blocks access to the repo to one thread at a time there was no reason to go beyond that. We knew our lock worked and all our functions worked.