Evaluating Martus

At the start of our project we had the goal of downloading, building, and running the “Martus” codebase. Martus was designed to be an extremely safe data storage service for sensitive personal issues, primarily documenting domestic abuse. We downloaded and tried to build the java project in eclipse, but we were immediately met with incompatibility errors and limited documentation to help us navigate them. After researching some more, we realized that the last updates to the project were almost 4 years ago and that the servers that supported the service were no longer online.

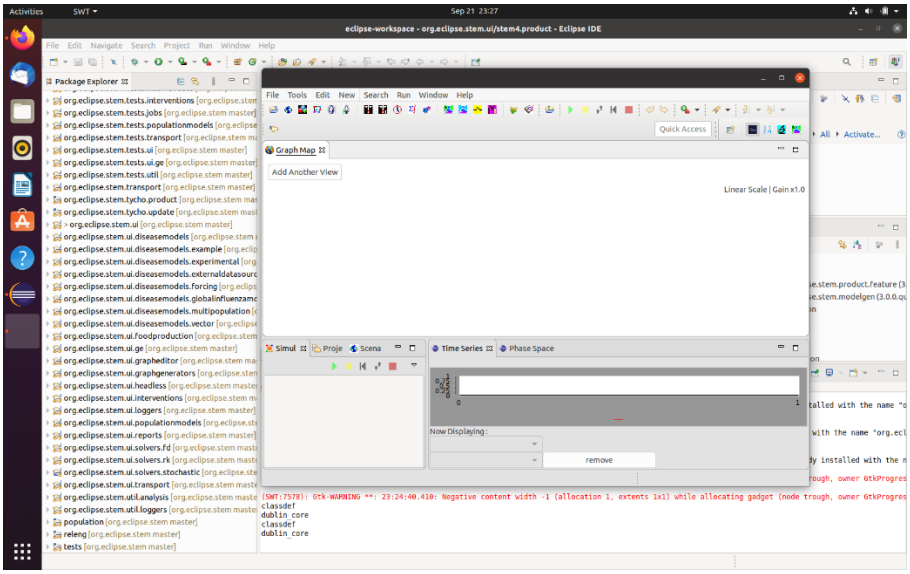
What did that mean for our project? Could we still even work with the code? Technically it was possible for us to work with the code if we had more time and resources to troubleshoot the failed builds. Even if we were to overcome these issues, however, we would have to either dedicate one of our machines to run as a server or all run individual private servers. Running local servers would make it difficult to ensure we were working with the same data set, but running a live server from one of our homes was not a viable option for our group either. We decided that the project had too many hurdles for us to navigate in a short time and decided it would be a better idea to focus our energy on a project with more recent updates, and one that was more optimized for newer hardware and software.

Exploring STEM (The Spatiotemporal Epidemiological Modeler)

After the difficulty Martus provided, our group was worried moving into The Spatiotemporal Epidemiological Modeler (STEM). Luckily for us, STEM is still being updated regularly and has an in-depth guide to getting it set up in Eclipse. This made it far easier to get working in Linux. A step-by-step guide is provided on the project wiki, including download links for necessary installers and detailed instructions for ensuring version comparability and dependencies.

Following these guides, we had a thankfully uneventful process of getting the source code downloaded, installed, and ready to run in Eclipse. There were some build errors in our group, but these were resolved by reinstalling the latest version of Java. We didn't see any tests on the wiki, but there were several demos. Upon running STEM initially, we were concerned about another error we received, but interestingly this was part of the design. This was in order to specify and set up initial configurations for the local STEM instance.

Continuing to follow the tutorial, we eventually got STEM built:



Once the code was ready on our devices, we decided to start running some of the tutorials to test out how the user experiences the code. The tutorials gave us a good insight to how the code is supposed to be run and how the input and output looks so we can automate the inputs to send any range of values into the program and harness the outputs to test their accuracy based on the results.