* 1. Hardware/software mapping – How will subsystems be assigned to hardware? You may use deployment diagrams.

These subsystems will not require any special hardware mapping. However, this project will be utilizing GitHub’s deployment system to allow for continuous integration from all team members. This way, when any one developer works on a given feature, the entire project will be universally updated as to keep from doubling up on work. GitHub also has some convenient features that would allow us to deploy our working releases to the Windows App Store as to have a convenient way to push bug fixes and updates.

* 1. Persistent data management – Identify the data which will be persistent. Describe the file system or database to be used, including a complete database design.

Neither of our subsystems will need to persist data as the user determines how the data needs to be analyzed at runtime. In the creation of the Nexus file, the user will be prompted to determine what analyses the Nexus file must be prepared for, and the Nexus file generating system will be hardcoded to handle the two possible scenarios (sequential or morphological data). Instead, this system will only hold the Nexus information at runtime using object models as to keep this system fast. After the file has been generated, the system will offer an option to save said file to the hard drive of the user’s computer, but that will be the only information saved.

The same will be true for the Nexus file analyzer – there will only be an option to save the output to the computer’s hard drive where the output is a jpg or png file of the visual representation of the desired analysis.

* 1. Boundary conditions – describe how the system will be started up, initialized and shut down. How will it respond to errors and exceptions?

This system will be initially downloaded through the Windows App Store as a Windows app for any computer running Windows. This will allow the Windows Store to handle any updates or bug pushes. Once the app is downloaded and opened, the user will be given the choice to create a Nexus file or analyze an existing file. Once a Nexus file is either obtained or created, the user can then be offered to analyze the new file or go back to the main menu, leaving the shutdown operation of this system to the user whenever they choose to close the program. As far as error and exception detection, this system will be heavily validated. In the event that an exception/error occurs that was not previously anticipated for, the user will be sent to a generic screen stating that this system is encountering technical difficulty, and an automatic email will be sent to the project developers stating the details of said error/exception. In the event that this occurs, the user will be forced to either restart the system, or go to the main menu, which will force the user to lose and redo whatever work they had begun on.