During this iteration, the development team really focused on separating the domain logic from the GUI. More specifically, a DisplayController class was added to communicate with the MainController. The goal was to ensure that no PModel object exists above the domain logic level (MainController), and that all GUI components (JComboBoxes and Lists) are generated by DisplayController via strings and integers which it requests from MainController. The mapping of GUI elements to the underlying objects is done in MainController, with minimal error checking in the GUI (such not accepting empty strings) and DisplayController (such as not accepting negative values).

A notable exception to this are the Analysis GUIs: GANTT, Pert, and Earned Value Analysis have access to the project that they are being generated upon. This is done primarily because they belong to the business logic level, and as such are not true GUI elements. Only their output gets displayed. The other reason is to avoid cumbersome and excessive argument passing.

MainController now coordianates with three helper objects which communicate with the database: a DataLoader, DataDeleter, and DataUpdater. This is done to increase some of the cohesion in MainController, as well as to alleviate its God class status. In the previous iteration document, we stated that we are not using a pure MVC design. In fact, we are not using the pattern at all; we are relying on message passing from GUI components (JFrames and JDialogs) to DisplayController, which in turn communicates with MainController, which delegates to the 3 data objects. If we were to redo the project all over, we would have implemented the pattern properly. However, given that a complete rewrite with our existing code base was out of the question, so we adapted as we could and learned a great deal in the process.