For our project we focused more on functionality than form due to the constraints placed on us: quick deadlines, large group, program requirements. We currently use parameters for most of our visualization in the ui, but time permitting an observer pattern will be implemented with a singleton pattern being updated when the ui elements are changed. We didn’t directly implement a chain of responsibilities but we would implicitly be using one with a print manager. Currently we have accessor methods in our DTOs that return the required values (visualization objects) we need for our other class, but we have branched and are working on converting that to a set of Adapter classes. We are not currently using a factory pattern due to limitations with our chosen CSV parsing library. A custom csvparser using some of that libraries functionality is currently being created at which point we can refactor our CSVDataAssembler to use a factory implementation and a strategy pattern. A strategy pattern has been used in a non release branch (merging requires refactoring of other classes). We do not use a composite pattern