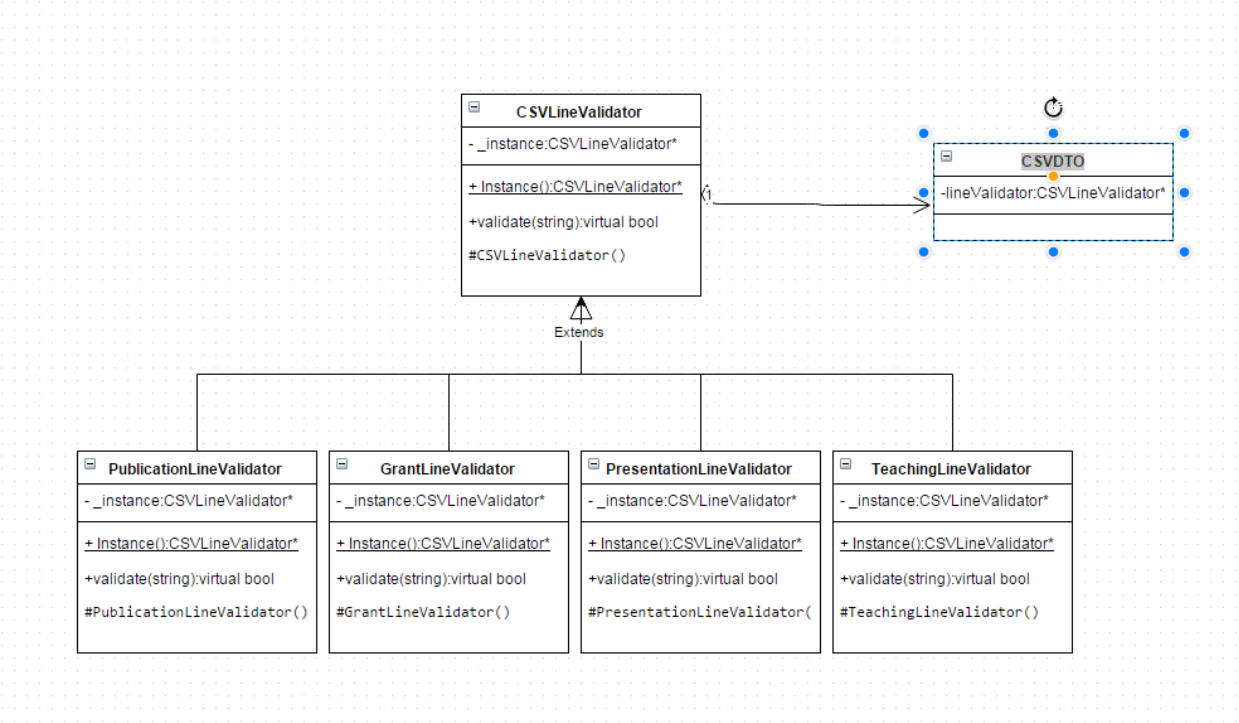
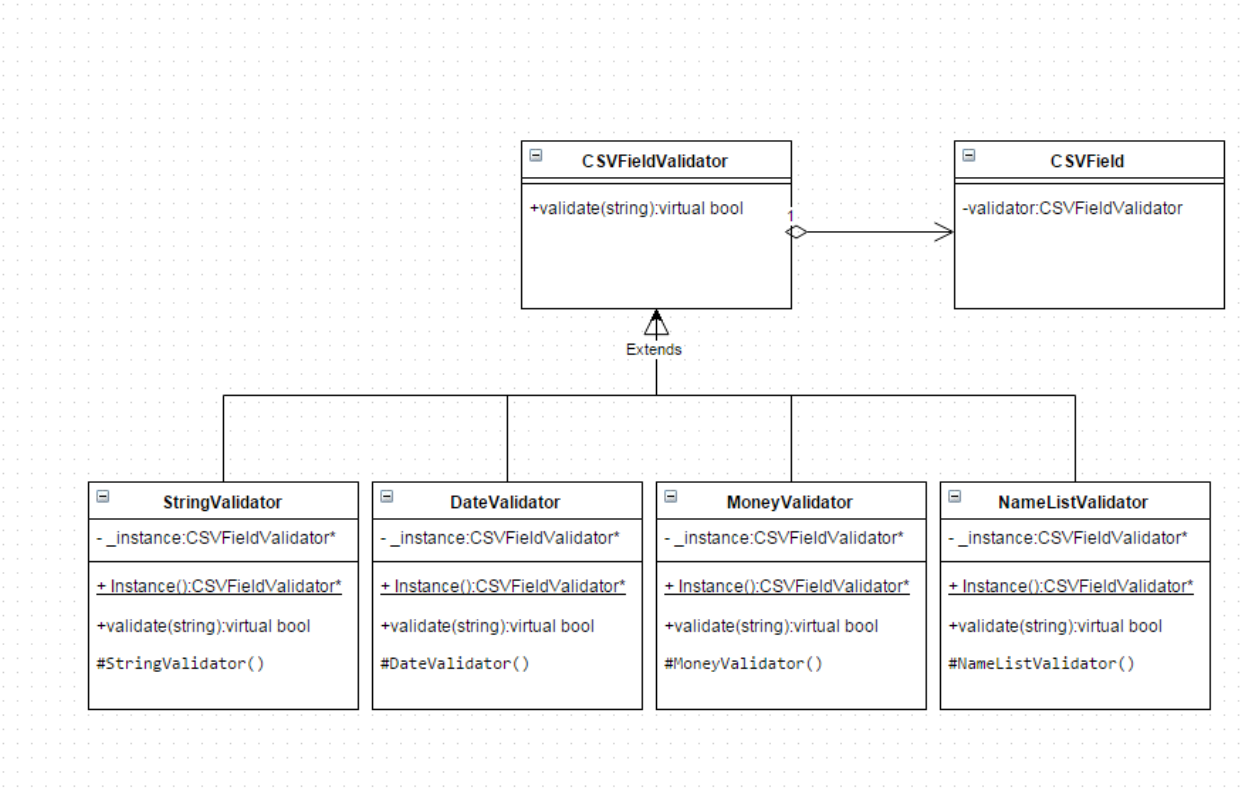
Design Patterns

In the parsing of the CSV of use a strategy pattern for both field validation (that a field contains the correctly formatted data) and for line validation (that each line makes sense as a whole ie. Start year <= end year). Each strategy is implemented with a singleton pattern as the need no shared data between the families of algorithms. Each singleton is accessed using a factory method. (Other methods not related to the patterns not shown)

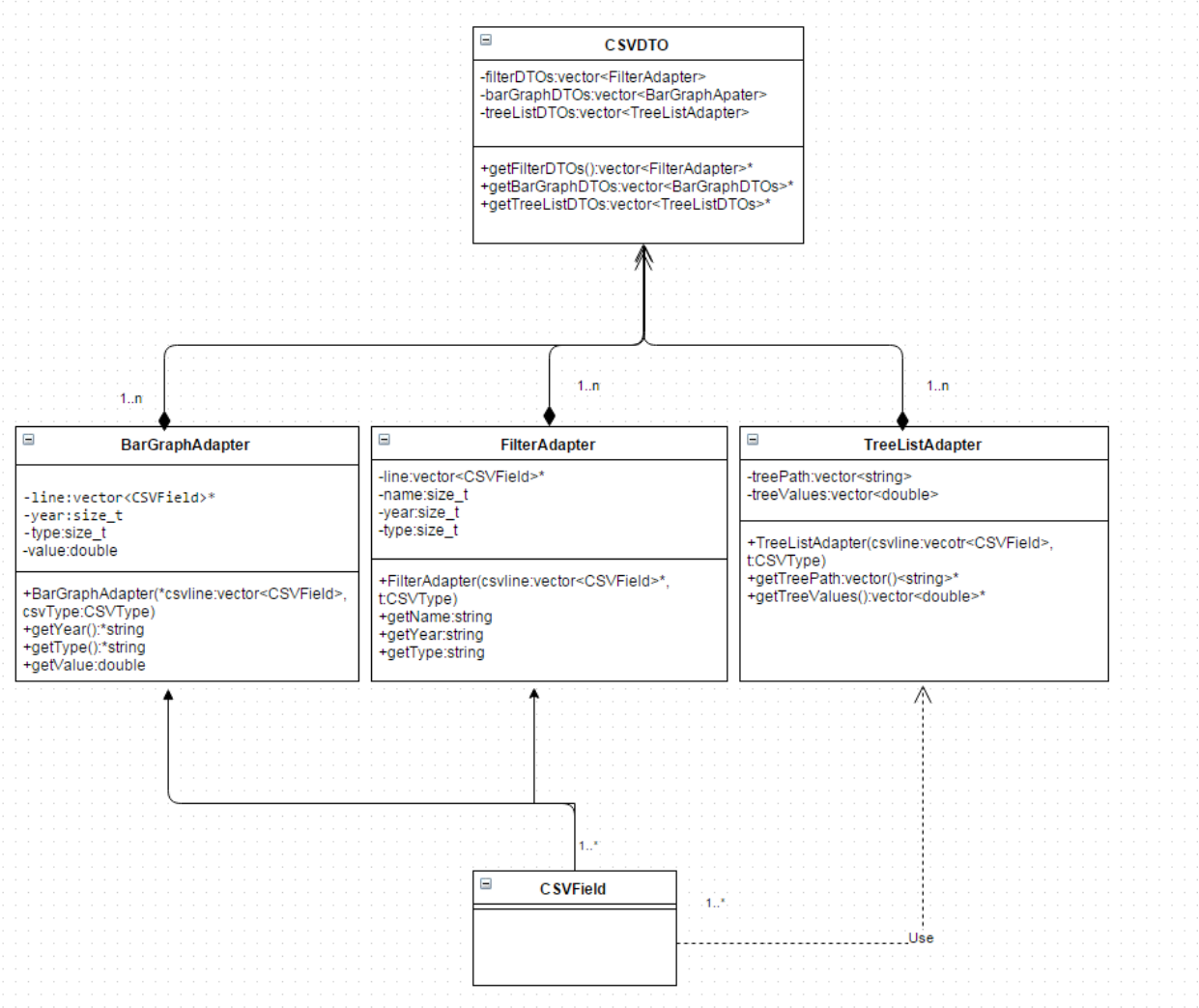
Line Validation strategy



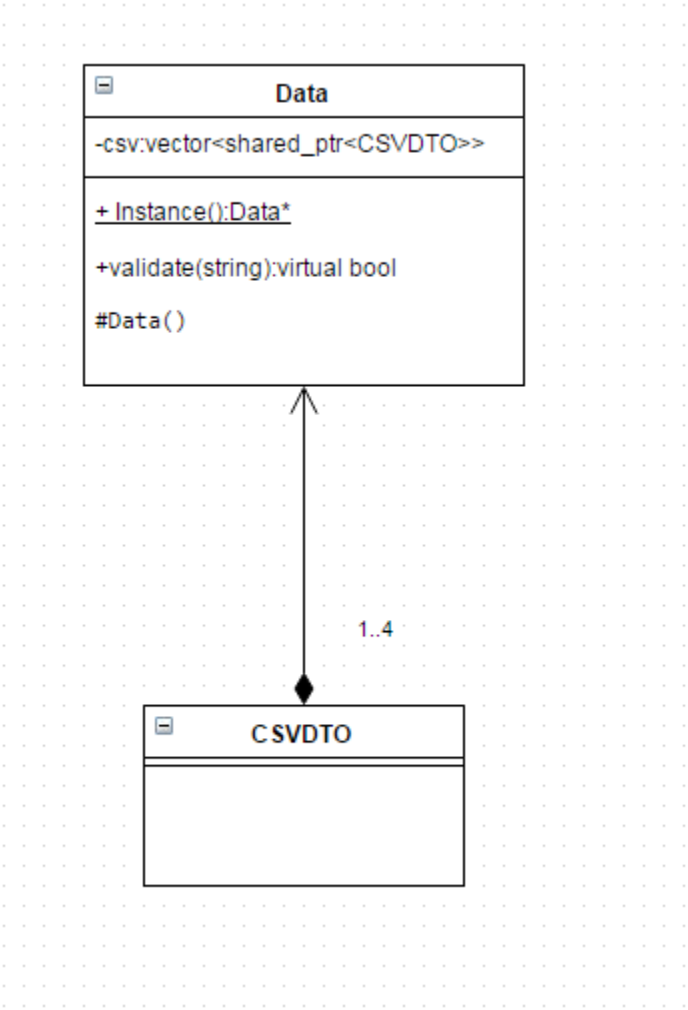
Field Validation Strategy



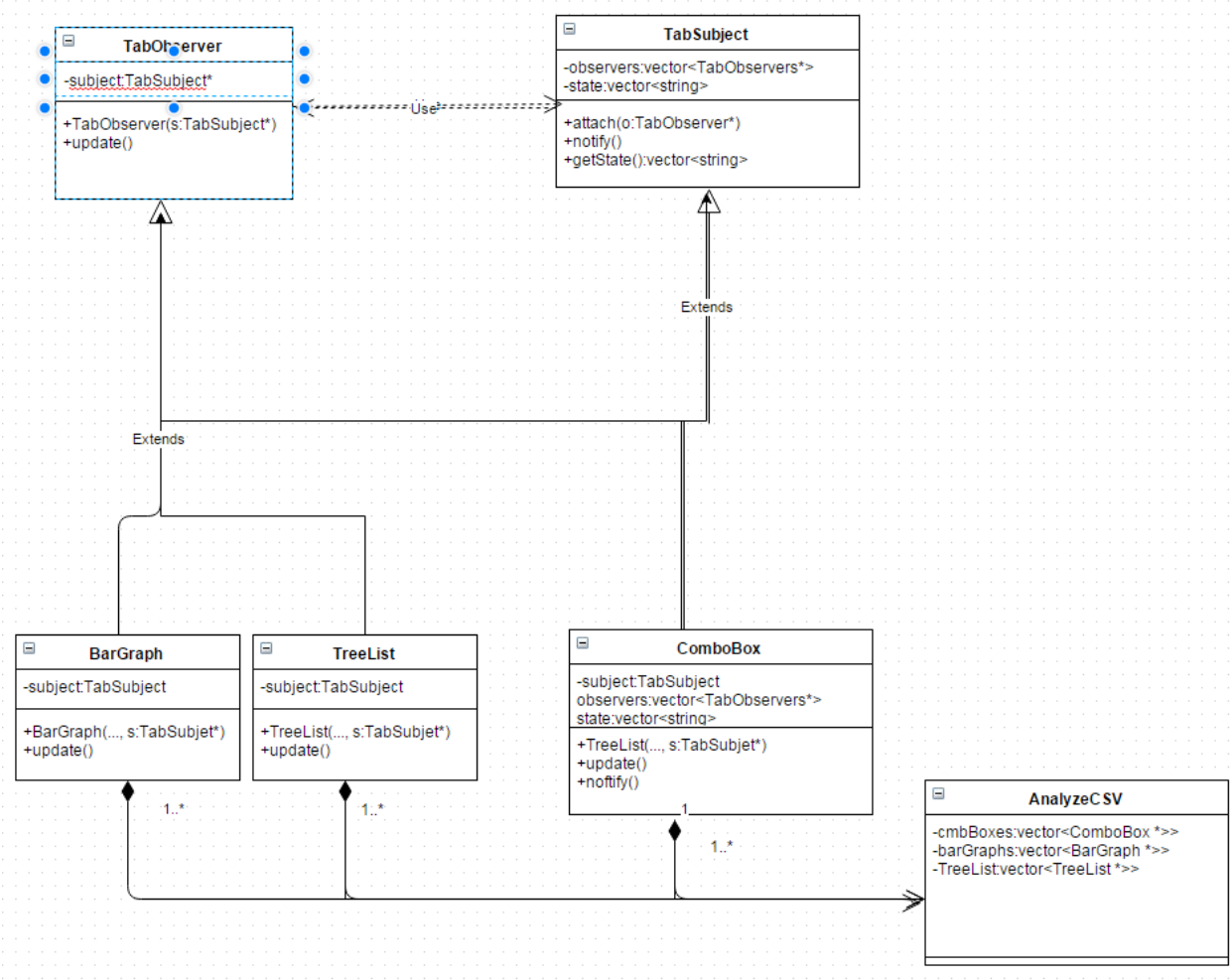
To make the parsed lines of text work well with our user interface we implemented 3 adapter class. One for each of tree list, bar graph and filter. The filter adapter class holds the necessary information to tell if a line of the text fits the selected filter the user has chosen in the ui. The bar graph adapter holds the necessary information to add a line of the csv as a data element to the bar graph, and the tree list adapter holds the path the csv line would take in the display tree from the root to the child as well as the values associated with that line.



The main storage of parsed data from the csv is held in a singleton Data class that holds up to 4 CSVDTO field and as above the Adapters for each dto as well as methods to manipulate them



The main logic in the analyze page is handled with an observer subject pattern. When the user changed the selection of a combo box, all dependent combo boxes (left to right, up to down order) are notified. Then the bar graph and tree list are also updated if the type or end year was changed respectively.



The ComobBox class uses a strategy pattern to determine which value from the filter adapter that it should populate. Each of these is again a singleton accessed with a factory method.