**Using Patterns to Improve Localization of Design Decisions and Performance** -Princy Jain

**Introduction:**

This assignment mainly focuses on using patterns to improve design decisions and performance. I selected the part b of the given assignment, i.e., Sudoku Solver. The patterns implemented in this project are Template Method Pattern which we were supposed to use. Other than this I implemented Strategy Pattern.

**Template method design pattern** is to define an algorithm as skeleton of operations and leave the details to be implemented by the child classes. The overall structure and sequence of the algorithm is preserved by the parent class.

In **Strategy pattern**, we create objects which represent various strategies and a context object whose behavior varies as per its strategy object. The strategy object changes the executing algorithm of the context object.

**Project Design:**

I have implemented in total four algorithms to solve Sudoku. The program reads size of the puzzle, domain and the puzzle to be solved from an input file. After this, the puzzle is passed to the validator which validates whether the puzzle is valid or not, for example it checks whether the size is a perfect square or not and also if the puzzle contains any other value rather than “-“, numbers and alphabets.

After this, the puzzle is passed to the algorithm selected by the user. If in case, the algorithm fails to find the solution, the puzzle is then passed to another algorithm.

Algorithm Used:

* Backtracking
* Stochastic Search
* Only one Value
* Depth First Search

Patterns Applied:

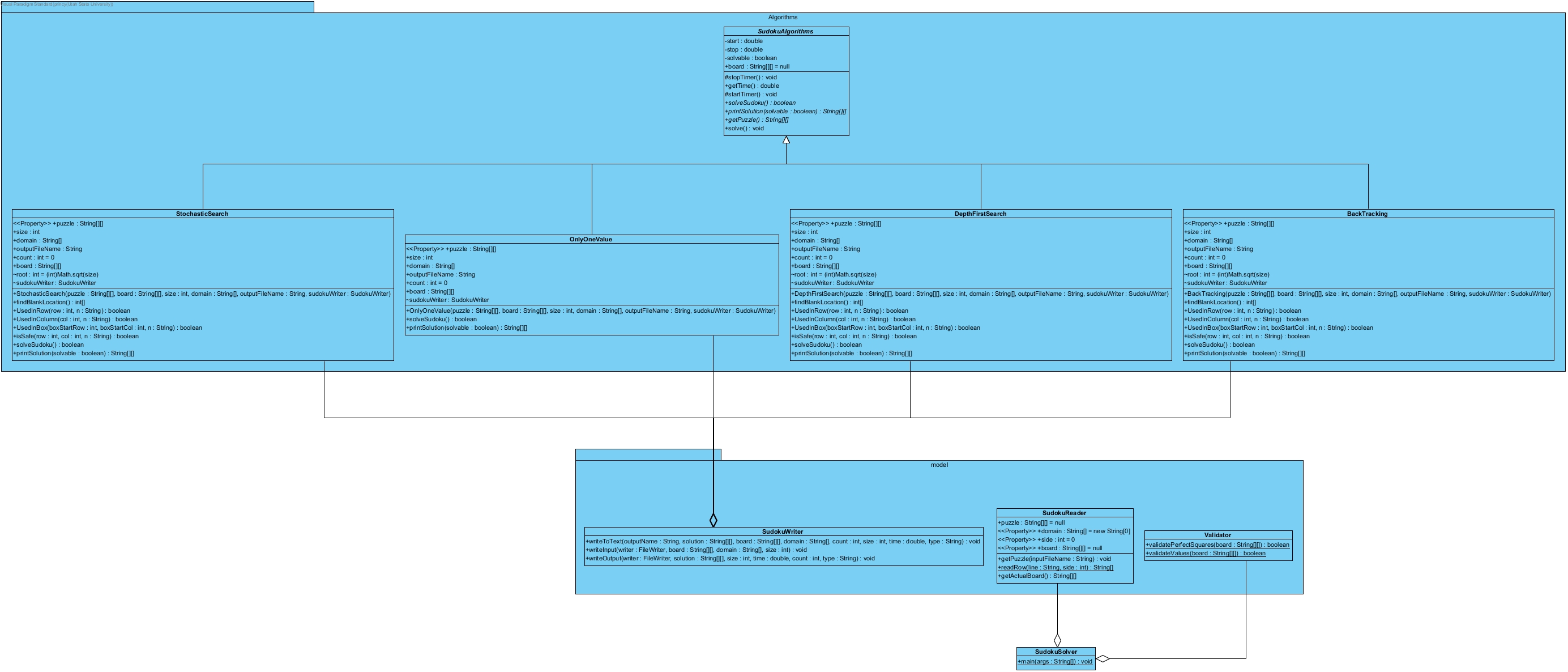
Template Method Pattern: It is used in the Sudoku Algorithms which defines the template for the steps which are followed to solve an algorithm.

Also, I have applied it to the Sudoku Writer which defines the fixed pattern on what to write first.

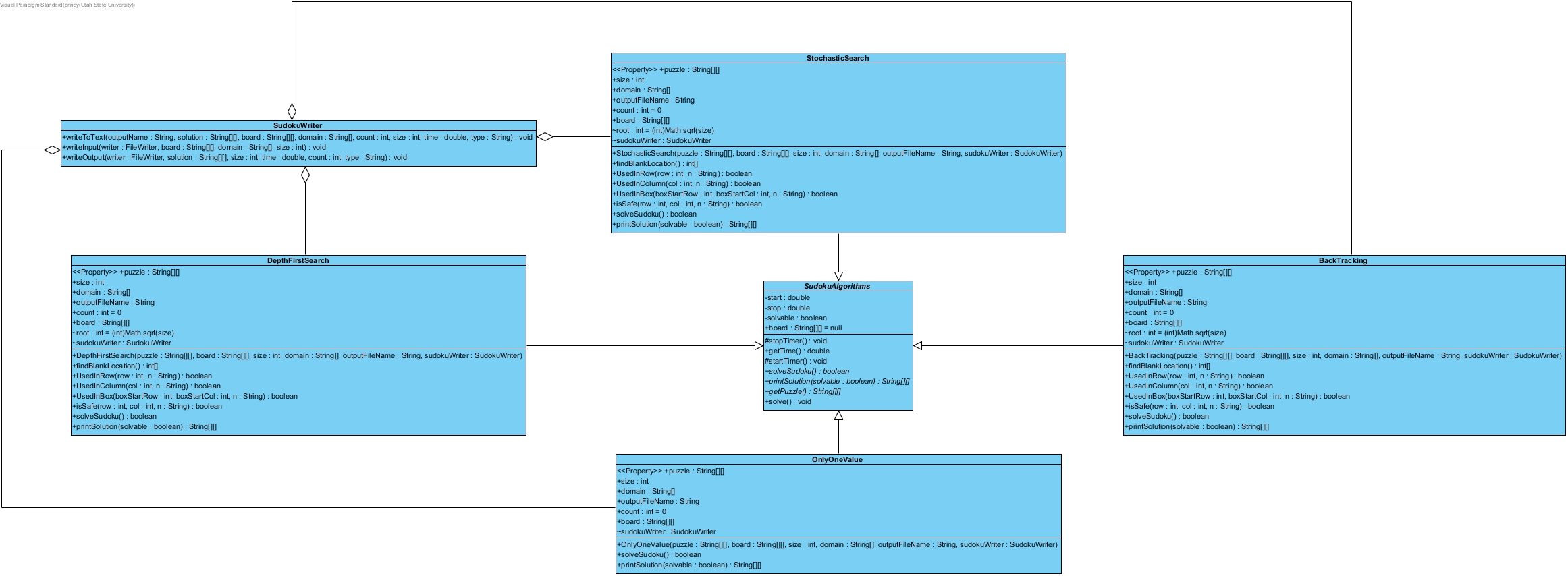
Strategy Pattern: This pattern is applied on the Algorithms. Based on users choice the object of the particular algorithm is selected at the run time.

Class Diagram:

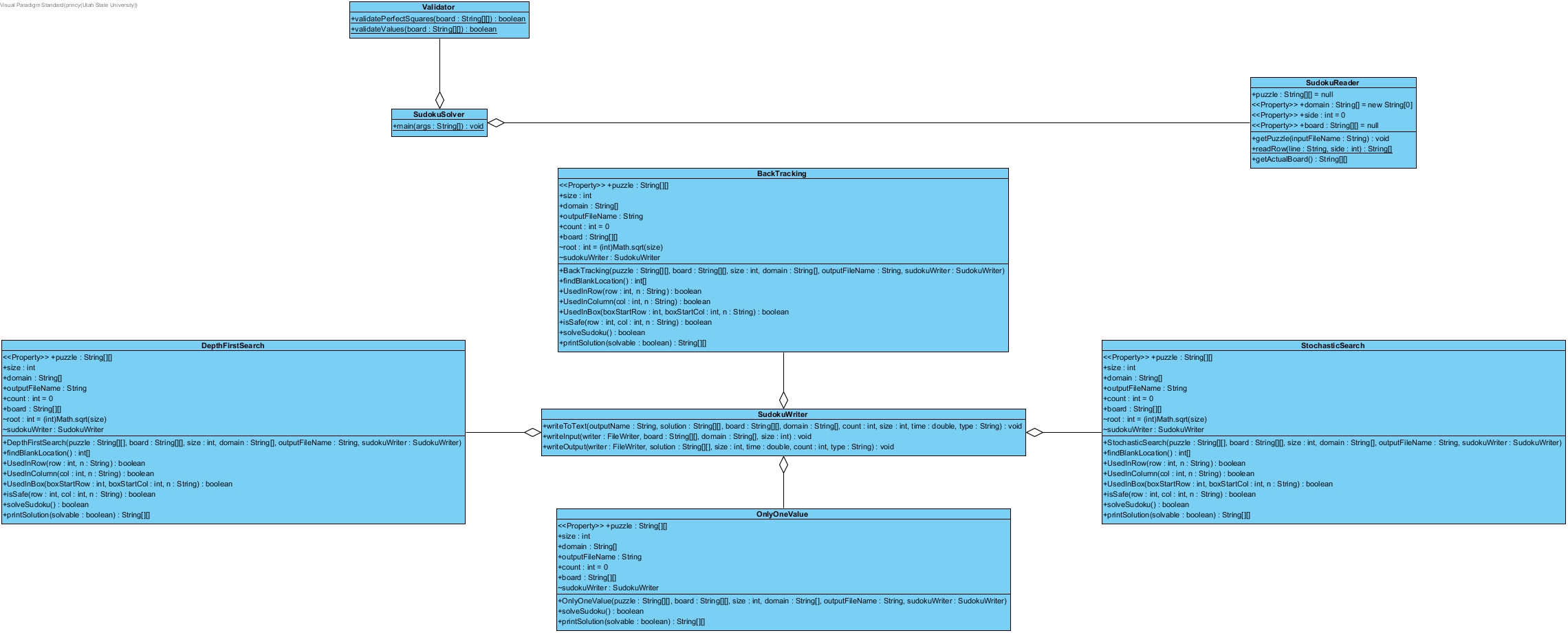
Diagram with all the Relationships shown among the classes:



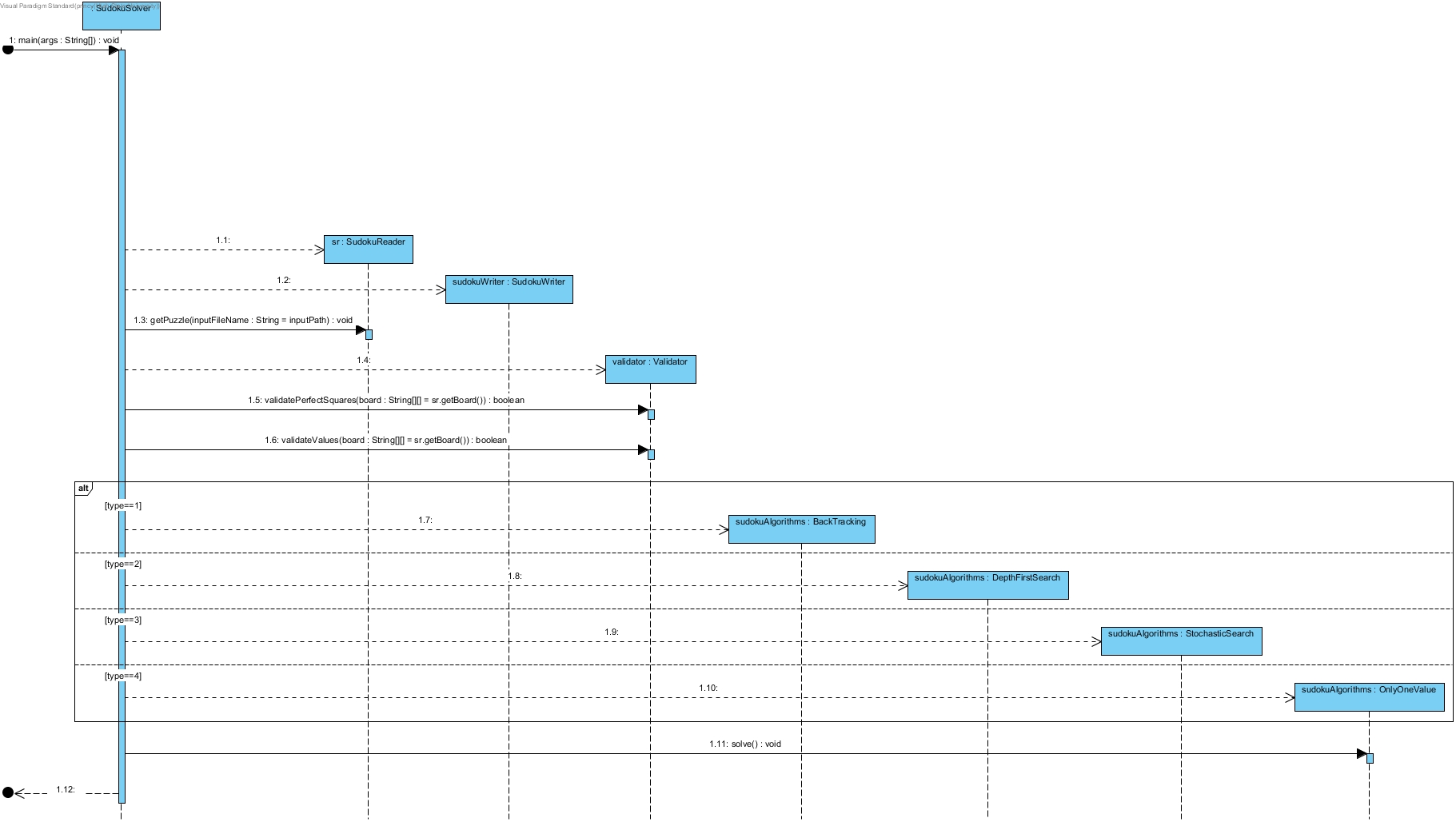
Algorithm Package:



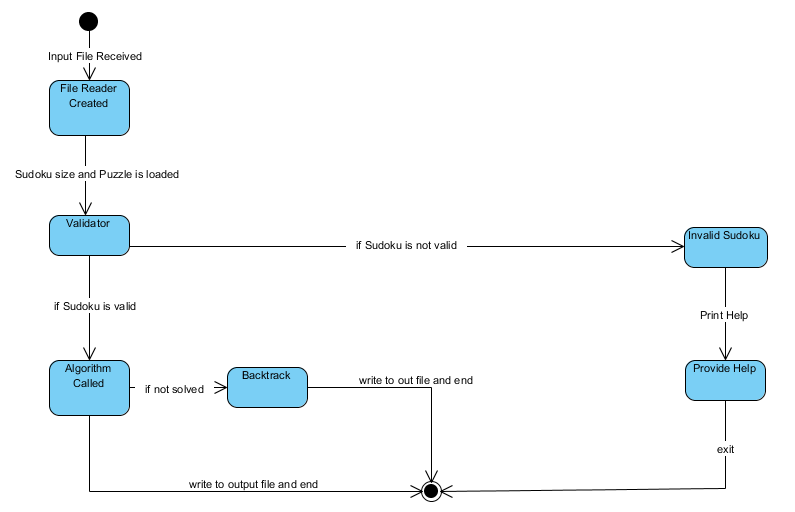
Model Package:



Interaction Diagram:

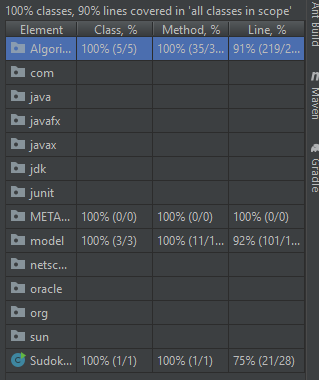


State Chart Diagram:



Unit Testing:

In unit all the classes and methods are covered with 100% coverage:



Insights Uncovered:

* Unit Testing is made much clear this time
* Template Method Pattern
* Strategy Pattern
* Learned different techniques to solve sudoku