CS 5700 - Homework 1 Report

Ligia R. Frangello

September 14, 2017

Strategy Pattern Design

Strategy Pattern defines a family of algorithms, encapsulates each one, and makes them interchangeable. This design pattern lets the algorithm very independently from clients that use it. In this case implementing the Strategy pattern allowed multiple Matching Algorithms, Importers, and Exporters, to be implemented separately but following the same parameters from a corresponding interface. Encapsulating each algorithm was important because the project" design requirements require extensibility.

UML Design

This project was different from what I have been conditioned to do in the Computer Science program. I spent a good amount of time not coding this assignment; which in other classes I would have just counted that time as "starring out of the window not being productive," but this assignment required mostly design time or "starring out of the window" time. I first started by trying to define the class names, their data members, and their methods. That part was relatively fast, and after 30 minutes or so I had come up with a partial UML design which can be seen below.

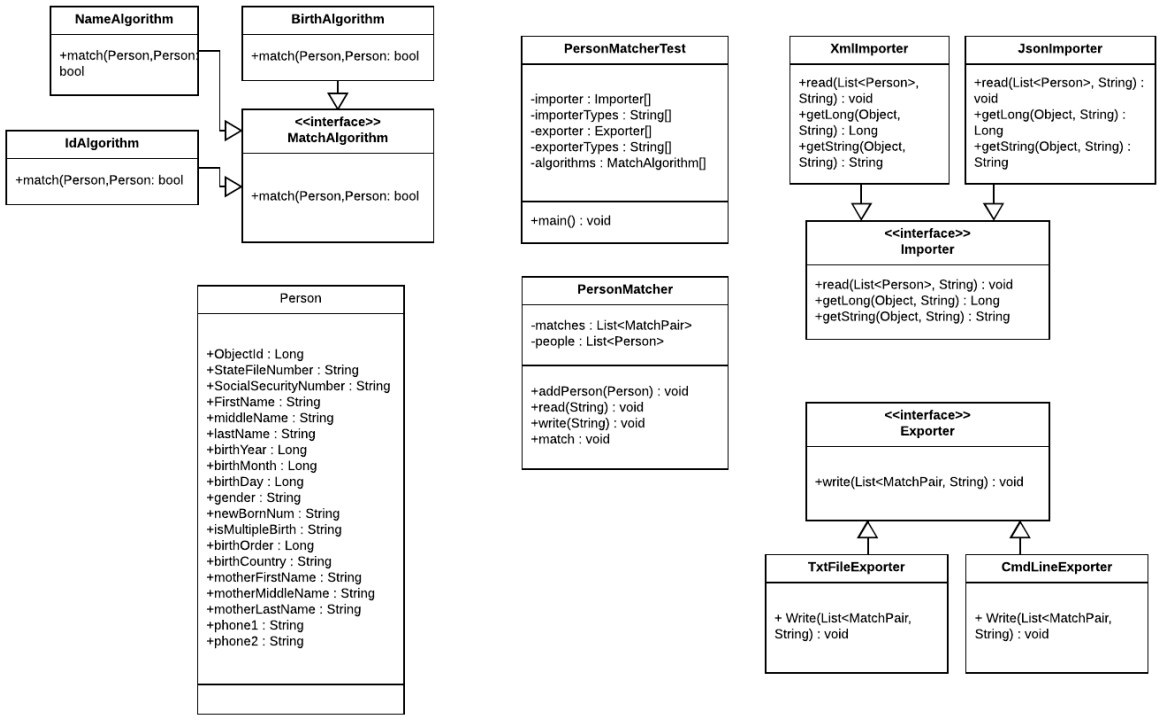

Figure 1: Initial UML design

Then it came the most difficult of part of this project, connecting the classes. Classes relationships is important to make sure no class owns information that it doesn't need and that classes don't implement something it doesn't need. I didn't fully decide on every relationship on my UML design until the very end, mostly because I am still a little confused to what each arrow means. But overall I implemented a

good relationship on my code and just added them to my UML design, which follows Strategy and good design patterns. The final version of the my UML design is bellow:

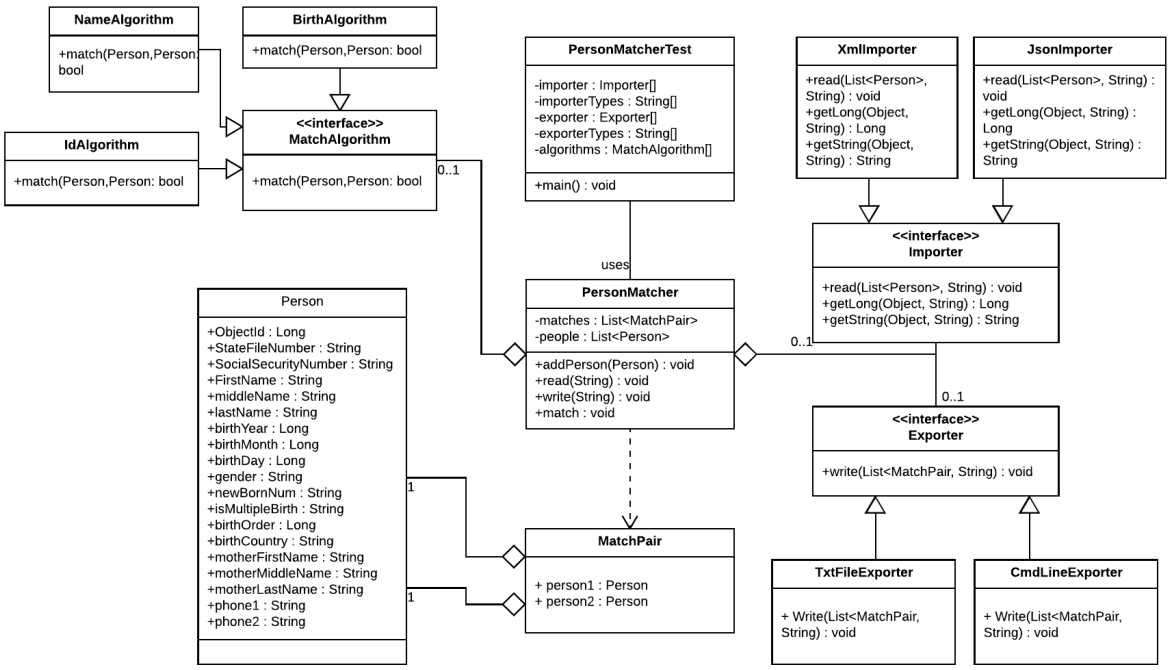

Figure 2: Final UML design

Lessons Learned

- When writing a UML diagram, not every box means a different a class, for example you can have a box for a data member.
- Simple constructors with getters and setters can make the program more modular and extensible.
- Strategy pattern with Arrays to hold the possible algorithms can replace many if/else statements.
- When doing unit tests, don't assume what has happened before, cover all possibilities.