Project 7

Final Submission

Project Name:
- WW2 Strategy Game
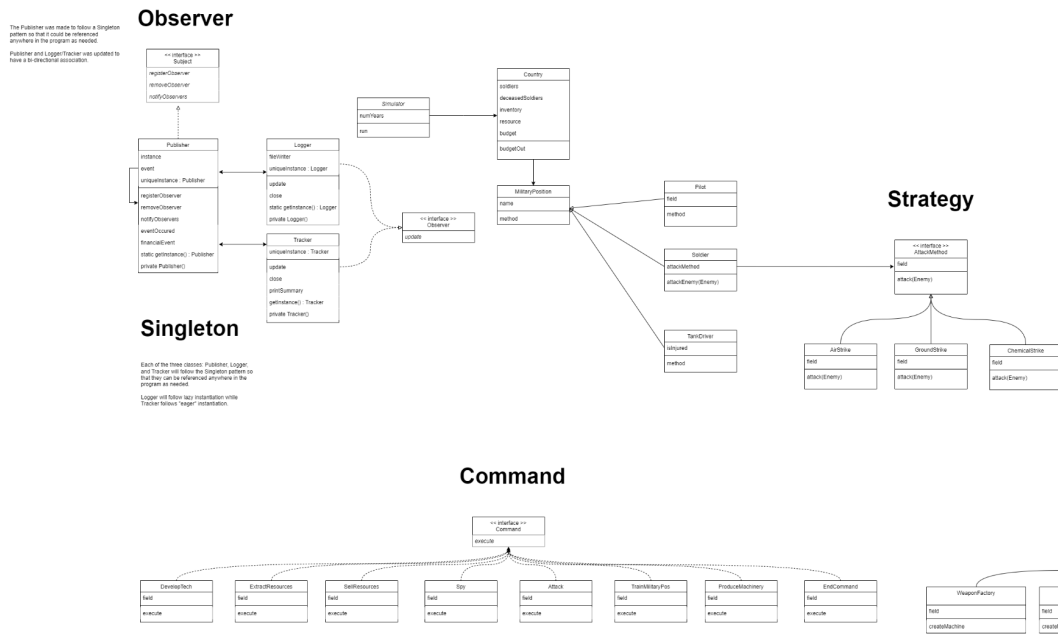
Names:
- Jonathan Taing
- Sultan Alshakrah

Final State of System Statement:
- The final state of our system implements a World War 2 strategy game where players can choose a country to play as. It can be either the USA or Germany.
  - The game has the following features:
    - Main Menu that includes options such as extracting resources, selling resources, spying on the other country, training military units, and attacking other countries.
    - A hub to manage resources and the training of military units.
    - A publisher to record the player's records and progress throughout the game.
  - Features that were not implemented:
    - Although the logger was set in place. It was not implemented fully to capture the player's records for each move (year) made by the player. We considered this a redundant feature because the player's records have already been recorded in the SimResults.txt file instead.
  - What changed from Project 5 and 6
    - Between Project 5 and 6, we moved the responsibility of the attack methods from the Soldier class to the Hub class itself.
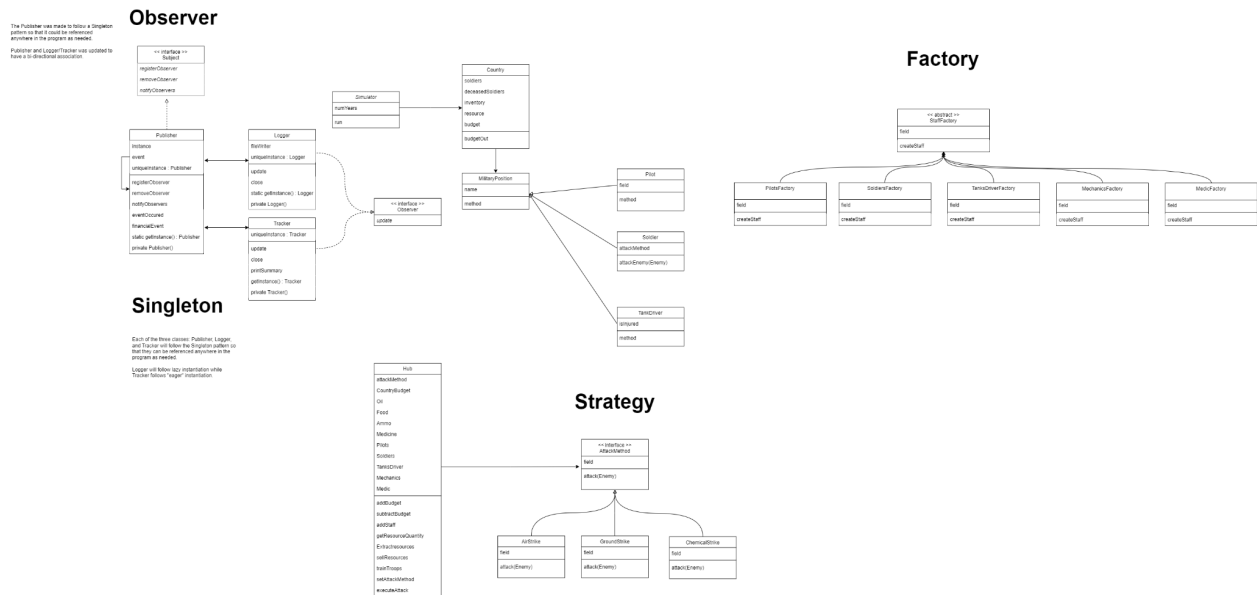
Final Class Diagram and Comparison Statement:
- Class diagram submitted in Project 5

UML for Project 5

Observer

The Publisher was made to follow a Singleton pattern so that it could be referenced anywhere in the program as needed.

Publisher and Logger/Tracker was updated to have a bi-directional association.

Singleton

Each of the three classes: Publisher, Logger, and Tracker will follow the Singleton pattern so that they can be referenced anywhere in the program as needed.

Logger will follow lazy instantiation while Tracker follows "eager" instantiation.

Strategy

Command

Factory

```
<< interface >>
Subject
--------------------
registerObserver
removeObserver
notifyObservers
```

```
Simulator
--------------------
numYears
--------------------
run
```

```
Country
--------------------
soldiers
deceasedSoldiers
inventory
resource
budget
budgetOut
```

```
Publisher
--------------------
instance
event
uniqueInstance : Publisher
--------------------
registerObserver
removeObserver
notifyObservers
eventOccured
financialEvent
static getInstance() : Publisher
private Publisher()
```

```
Logger
--------------------
fileWriter
uniqueInstance : Logger
--------------------
update
close
static getInstance() : Logger
private Logger()
```

```
<< interface >>
Observer
--------------------
update
```

```
Tracker
--------------------
uniqueInstance : Tracker
--------------------
update
close
printSummary
getInstance() : Tracker
private Tracker()
```

```
MilitaryPosition
--------------------
name
--------------------
method
```

```
Pilot
--------------------
field
--------------------
method
```

```
Soldier
--------------------
attackMethod
--------------------
attackEnemy(Enemy)
```

```
<< interface >>
AttackMethod
--------------------
field
--------------------
attack(Enemy)
```

```
TankDriver
--------------------
isInjured
--------------------
method
```

```
AirStrike
--------------------
field
--------------------
attack(Enemy)
```

```
GroundStrike
--------------------
field
--------------------
attack(Enemy)
```

```
ChemicalStrike
--------------------
field
--------------------
attack(Enemy)
```

```
<< interface >>
Command
--------------------
execute
```

```
DevelopTech
--------------------
field
--------------------
execute
```

```
ExtractResources
--------------------
field
--------------------
execute
```

```
SellResources
--------------------
field
--------------------
execute
```

```
Spy
--------------------
field
--------------------
execute
```

```
Attack
--------------------
field
--------------------
execute
```

```
TrainMilitaryPos
--------------------
field
--------------------
execute
```

```
ProduceMachinery
--------------------
field
--------------------
execute
```

```
EndCommand
--------------------
field
--------------------
execute
```

```
<< abstract >>
MachineFactory
--------------------
field
--------------------
createMachine
```

```
WeaponFactory
--------------------
field
--------------------
createMachine
```

```
AirplaneFactory
--------------------
field
--------------------
createMachine
```

```
TankFactory
--------------------
field
--------------------
createMachine
```

- UML class diagram representing the final set of classes.

UML for Project 7

https://drive.google.com/file/d/1Ch8rut1E__2hXjVCl5-7GxyEPG3qoibe/view?usp=sharing

Between Projects 5 and 6, there were some key changes that were made for the final iteration of the project. We removed the need to implement the need for the Command Design Pattern for our CLI menu. We also created a Hub class to manage all the resources for the player's country and military units. Because of this, it made the most sense to move the responsibility of the attack methods from the Soldier class to the Hub class itself.

Third-Party code vs. Original Code Statement

The majority of the code was original with some of the design elements for the logger (Singleton and Observer) inspired from our earlier project work throughout this course.

Statement on the OOAD process for your overall Semester Project

- The Hub class ended up handling a lot of tasks. They may have resulted in high coupling between different functionalities and low cohesion within the class.
- There wasn't any testing or validation done on the code. This wasn't a problem for a project this small, but would have been wise if our project grew in the future.
- We were very satisfied with the usability and the user experience. The design ended up being easy to navigate.