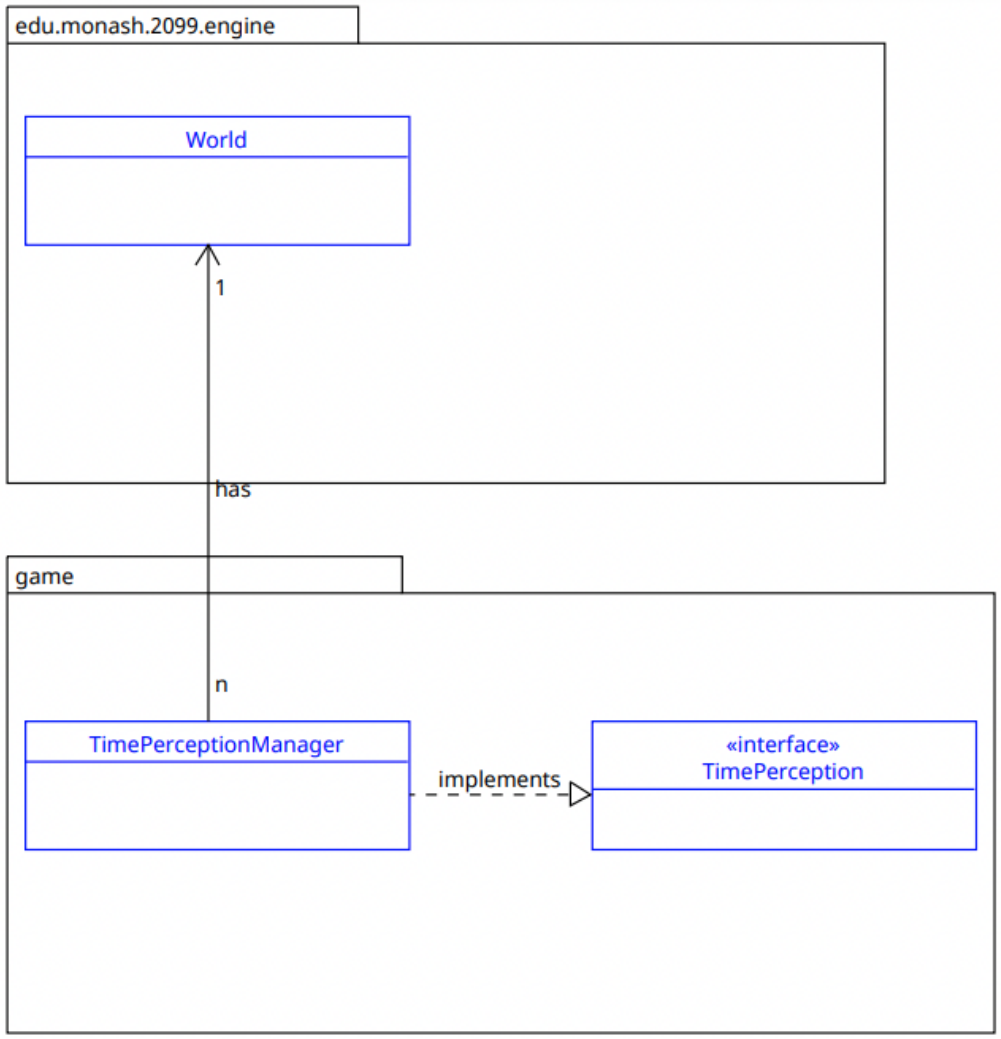
Assignment 1: Paul-Antoine Galliot REQ5 & REQ6

Requirement 5: Day and Night

Uml Diagram-



Design Rationale-

New classes to be added:

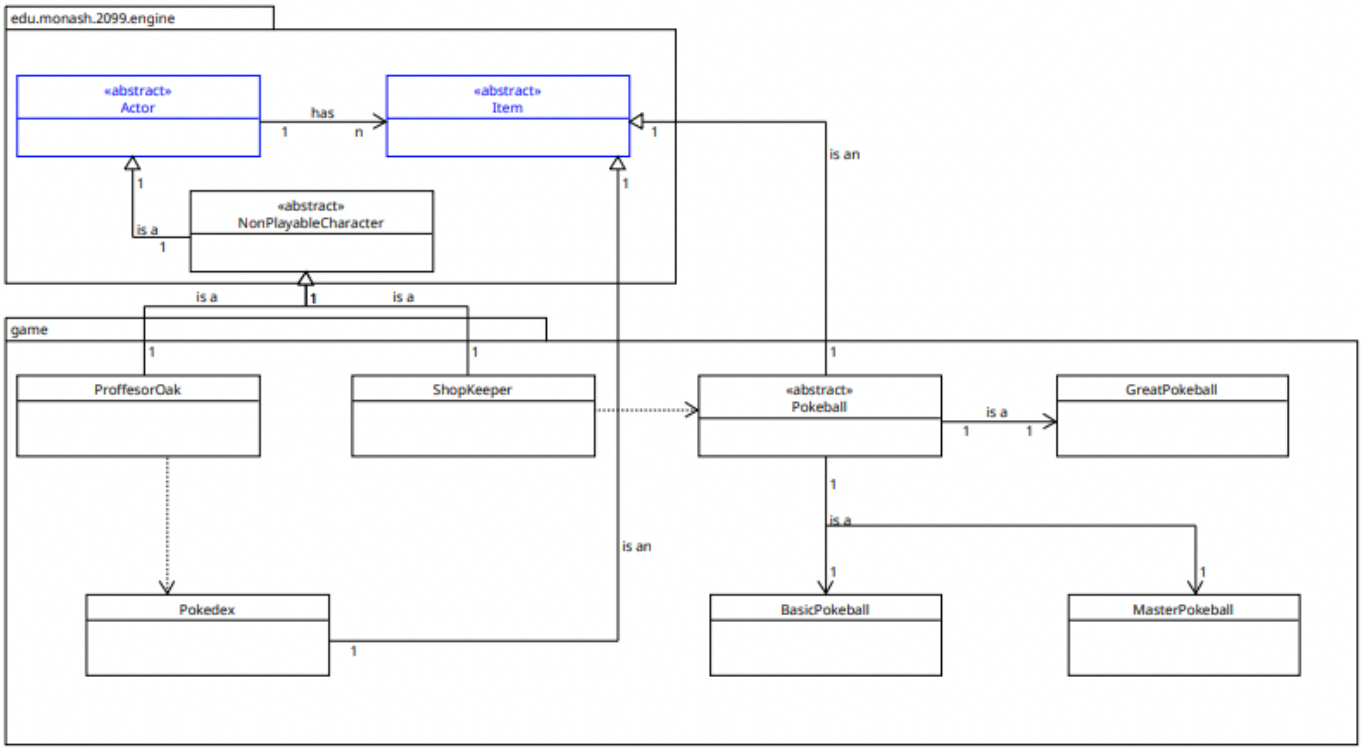
- None

Reasoning:

No extra classes should be added as they are not needed. Instead of adding a class all I need to do is create an association between the 'World' class and the 'TimePerceptionManager' so that the world is aware of what time of the day it is in order to apply the correct changes. Furthermore I need to implement the ‘TimePerception’ interface in the ‘TimePerceptionManager’ class so I can add the necessary functionality.

Requirement 6: NPCs

Uml Diagram-



Design Rationale-

New classes to be added:

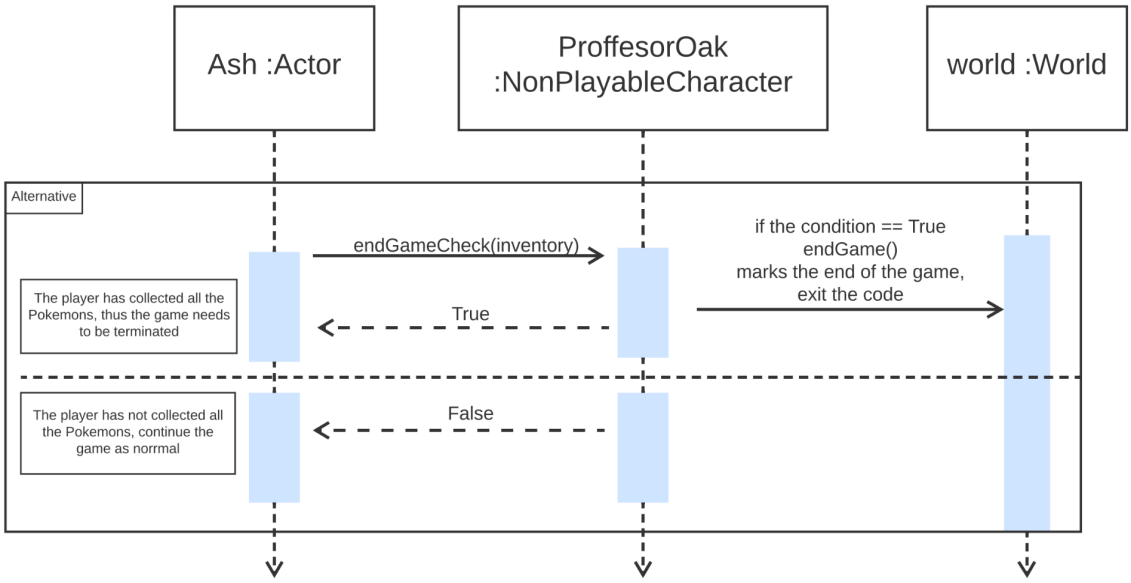
- NonPlayableCharacter <<abstract>>
- ProffesorOak
- ShopKeeper
- Pokeball <<abstract>>
- BasicPokeball
- GreatPokeball
- MasterPokeball
- Pokedex

Reasoning:

For the NPC requirement I decided to add a multitude of new classes which would aid our group in following SOLID principles. Firstly I added an abstract class called 'NonPlayableCharacter' which extend the ‘actor’ class and agrees with the open-closed principle as modifying the ‘actor’ class won't be needed when adding a NPC such as ProfessorOak or the ShopKeeper, thus I’m simply extending not modifying the ‘actor’ class. The purpose of making 'NonPlayableCharacter' abstract is for the new code that will be implemented to follow the dependency inversion principle by not inheriting from a concrete class. This new abstract class will outline the functionality that these NPCs will have and also acts as the blueprint for the two new subclasses that will be added that are 'ProfessorOak' and 'ShopKeeper'. As a group we also decided to add an abstract ‘Pokeball’ class which will serve a similar purpose of the 'NonPlayableCharacter’ class as it adheres to the SOLID principles in the same manner that the 'NonPlayableCharacter’ did. It

will extend the ‘Item’ class and three subclasses will extend it, they will be called ‘BasicPokeball’, ‘GreatPokeball’ and ‘MasterPokeball’. Even though creating a pokedex object is not necessary I decided to add one as it may be used in future assignments where we will have to implement its functionality and it only requires a insignificant amount of lines of code.

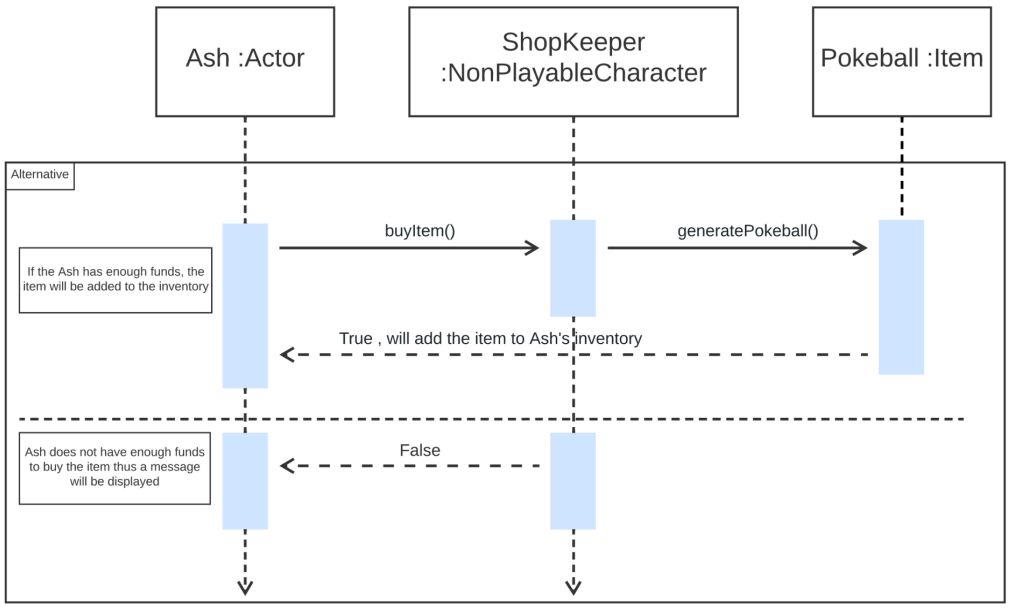
Interaction Diagram 1: Professor Oak and Ash



Interaction Diagram Information-

This interaction diagram depicts an interaction between ProffesorOak and Ash. The main point of their interaction is to check if Ash has caught all the pokemons thus signifying the end of the game. If that condition is met through endGameCheck it will call endGame in the World class which will terminate the game.

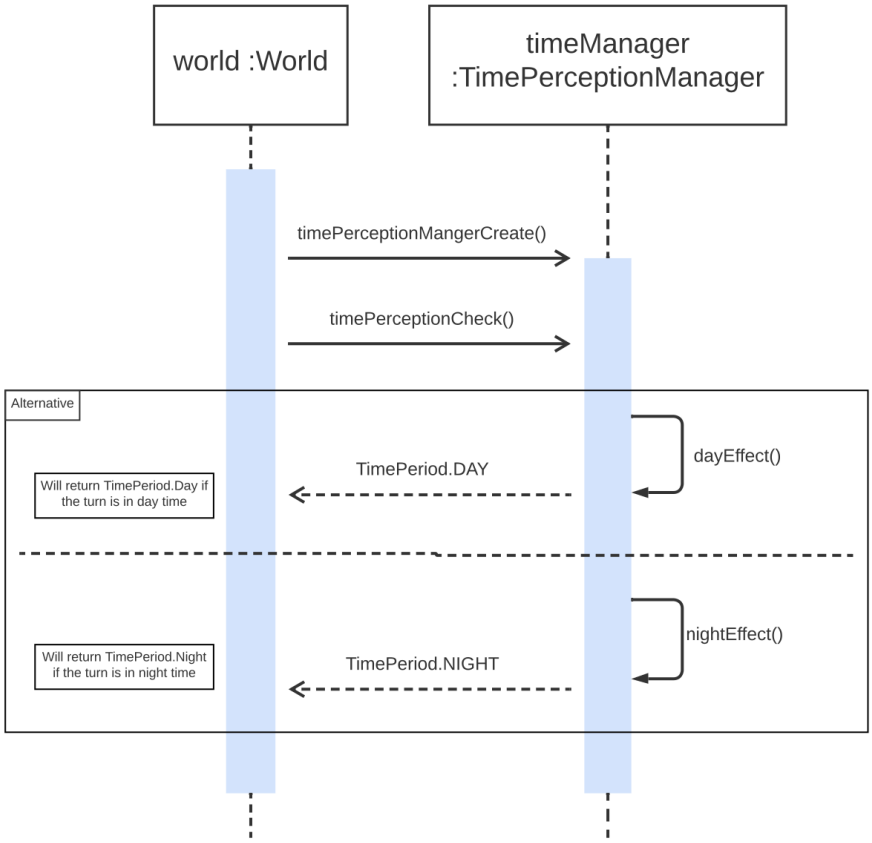
Interaction Diagram 2: ShopKeeper and Ash



Interaction Diagram Information-

The interaction diagram between Ash and the shopKeeper illustrates an interaction between the actor and the nonPlayableCharacter. Ash will try to buy an item through the ‘buyItem’ method. If Ash has enough funds to buy said item it will be generated through ‘generatePokeball’ which will return true and add the item to the actor’s inventory. In the case of there not being enough funds, an error message will be displayed and the method ‘buyItem’ will return false

Interaction Diagram 3: Day&Night



Interaction Diagram Information-

The relationship between time and the world is very basic. The world object needs to check whether it’s day or night time and then act accordingly by calling the two methods that are ‘nightEffect’ and ‘dayEffect’ which are part of the ‘TimePerception’ interface, it will also return the enumeration constant whether that be TimePeriod.Day or TimePeriod.Night .