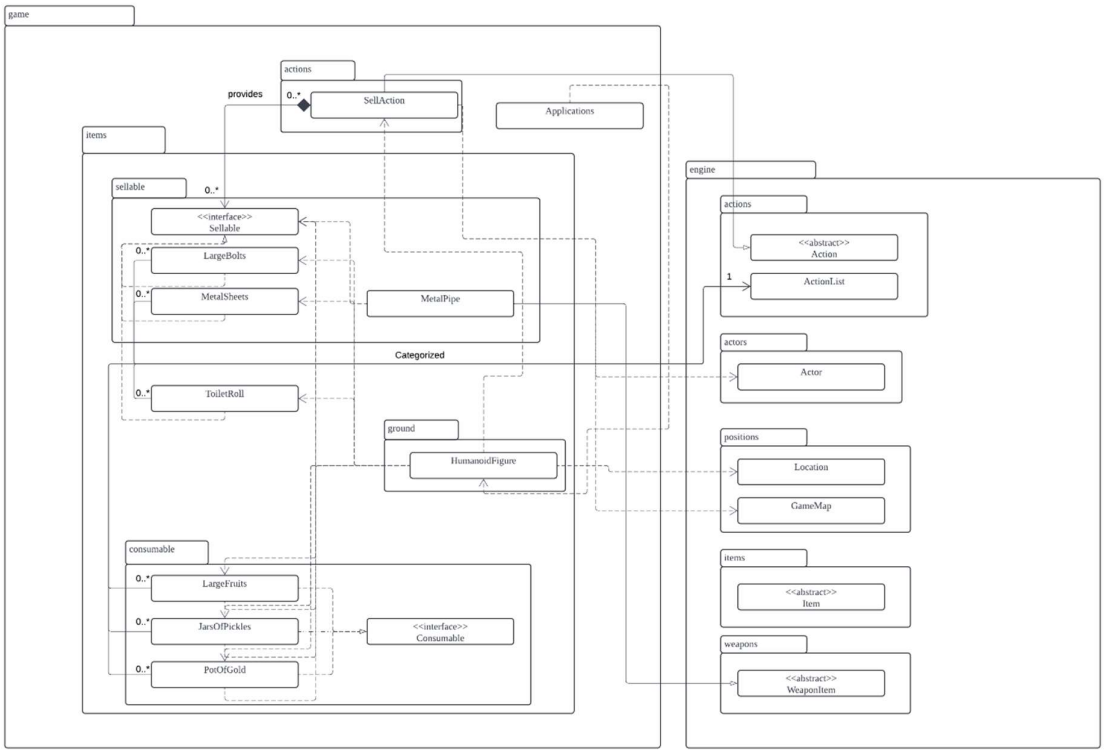
Assignment 3 Design Rationale REQ 2

UML



Design Goal

The design goal for this is to allow selected items to be sellable objects that can be sold for credits from a Humanoid Figure located in factory's spaceship parking lot. Some of items may have specific conditions when selling the object.

Design Decision

The approach for requirement 2 was to implement a HumanoidFigure Class, Sellable interface and SellAction Class.

The HumanoidFigure Class was implemented with the purpose of allowing the player to sell applicable items in the player's inventory.

Sellable interface was implemented with methods such as sell() and getSellingPrice(), to allow all the sellable objects to be able to override the methods. With this interface, the applicable items that can be sold such as MetalSheets, MetalPipe, PotOfGold, JarsOfPickles, LargeFruits, LargeBolts and ToiletPaper all implement the interface with their own specific working conditions.

SellAction class was implemented and also extended to the Action Class with the purpose of allowing the items to be sold in the Humanoid Figure. To allow all the applicable items to be sold, the Status Enum Class is also altered with an addition Enum named “SELLABLE” which can be added as the item’s capabilities.

Alternative Design

An alternative design may be implemented using only Humanoid Figure only, avoiding the use of Selling Action.

Analysis of Alternative Design

With this alternative design, the design violates the SRP within the SOLID Principle. The main problem with letting Humanoid Figure handling all the operations which heavily violates the SRP principle. Besides that, DRY principle would be violated as the conditions to sell items would be repeated constantly for all the required items.

Final Design

The implementation follows the following principles.

Single Responsibility Principle (SRP)

Each class designed was implemented with only one specific objective. The Sellable interface was implemented with the use of providing sellable items with methods with conditions to sell each item. For example, the LargeBolt implements the Sellable interface which allows it to override the methods from the interface to make LargeBolt become a sellable object.

Open/Closed Principle (OCP)

OCP is implemented with the purpose of promoting easy extension with no modification towards existing classes. The MetalPipe Class adheres to this principle as it is open for extension but closed for modification. This may be explained as the WeaponItem Class is not modified while it is still open till extension and implementation such as the Sellable interface, inheriting the methods.

Interface Segregation Principle (ISP)

Interface Segregation Principle emphasizes on avoiding necessary dependencies on method that a class don’t use. The MetalSheet Class solely implements Sellable interface which fits

the principle. This allows MetalSheet to focus on behaviours relating to selling only, which in turn improves extensibility and maintenance.

Liskov Substitution Principle (LSP)

Liskov Substitution Principle promotes the ability which allows parent class objects to be replaced by sub class objects with no alteration of code, improving flexibility. All the sub classes that implement the Sellable interface can be used to replace Sellable interface without any code modification, which promotes flexibility. For example, in SellAction where it expects a Sellable. If we put any of LargeBolt, MetalSheet, and ToiletPaper through it, the code will accept it with no errors.

Conclusion

In conclusion, the code provides high modularity and extensibility, and maintenance capabilities due to its good adherence towards the SOLID Principle such as SRP, OCD, and ISP,. This solution is feasible due to its modularity for future improvement if needed.