<u>Changes in design (Assignment 2)</u>

**This documentation only shows the changes in design and the reasons behind. For complete updated documentations, please see "Design Rationale".*

Class changes: Limb

- Instead of storing upgraded weapon as an attribute in this abstract class, this attribute is stored in respective child classes such as Arm class and Leg class.
- Instead of declaring the damage as a final static attribute, the damage for all the limb subclasses is initiated in its constructor by using the super() keyword.

Class changes: AttackAction

- A new method called dropWeapon(Actor, GameMap) is added to handle the situation where the actor has to drop everything from its inventory, this method is called either when an actor is dead or a zombie has lost both of its arms. It has a protected visibility because this method is needed to be accessed by its subclasses which is AttackZombieAction and AttackHumanAction.

Class changes: AttackZombieAction

- Instead of handling the probability of a zombie losing a limb in its execute method, the functionality is delegated to Zombie class in order to achieve encapsulation where only the Zombie will decide whether it will lose a limb or not after being attacked.
- The deduction of the number of limbs when a limb has been knocked off a zombie will also be handled by Zombie class so that it achieves the preferable "tell, don't ask" approach.

- The name of the private method dropAt() is changed to dropLimb() and it will be responsible for all the actions involved in removing a weapon from the Zombie's inventory and spawn that weapon onto the adjacent location of the Zombie. So it will not return a location which is suggested in previous dropAt() method but rather handling the whole process of dropping a limb.
- In previous design rationale, a private method called dropWeapon() will be created to take in the number of limb lost and returns a boolean on whether the Zombie will drop the weapon or not. Since the zombie is only allowed to drop one limb at a time, this method is no longer needed.

Class changes: Zombie

- Both loseArm() and loseLeg() methods are set to have a private visibility since only Zombie will determine if it will lose a limb or not after being attacked.
- The dropWeapon() method stated previously which handles the action of removing all the weapons from its inventory is no longer needed in Zombie class as this action is handled in AttackZombieAction. Since AttackZombieAction inherits from AttackAction which already has this dropWeapon() functionality, it can reuse the method and achieve the design principle "DRY".
- A new method call zombieIsAttacked() is added to determines if the Zombie will lose a limb after being attacked.
- The private attribute Behaviours has an array of behaviours that are arranged such that the Zombie will have a 10% chance of shouting something every turn regardless of the conditions the Zombie is in. Then it is followed by pick up weapon behavior and attack behaviour, so that the zombie will choose to pick up the weapon first before attacking a human nearby. The crippled behavior if placed right before hunt behavior and wander behavior so that the Zombie will move in alternating turns if it has only one leg or cannot move at all if it has no leg.

Class changes: CraftWeaponAction

- The constructor only takes in one parameter of type Limb instead of Actor and Item. This is because only weapons of type limb is upgradable and have the method needed to craft them into better weapons. The actor parameter is no longer needed as this action does not need to know which actor is carrying the weapon.

Since the interactions between classes will remain the same, the UML class diagram will not be changed.

Class Changes : PickUpWeaponBehaviour

- Add an if statement to make maximum size of zombie's inventory to 1. Therefore, zombie is allowed to pick up 1 weapon only unless it drops the previous weapon.

Class Changes: ShoutBehaviour

- Change an attribute type from ArrayList to Array. (From ArrayList<String> to String[]) It is to initialize the Array with elements in it before going into the method. (For example, `private String [] words = {"Brainnn"} instead of Private ArrayList<String> words = new ArrayList<String>()).`
- Check the probability 10% of shouting word in ShoutBehaviour instead of in ShoutAction. This is because check the probability in ShoutBehaviour makes the zombie to carry out other action if the probability 10% is not reached. As we know, if action returns null, the other action will be taking place.

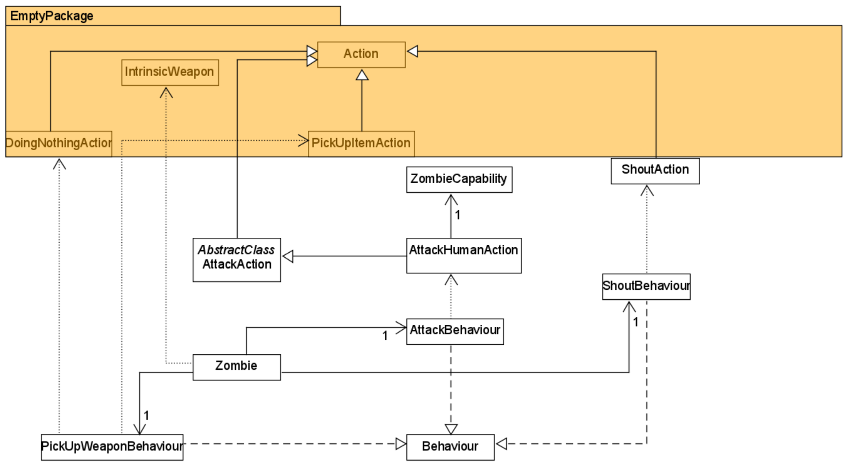
Class Changes : Word

- This class is removed because it is redundant. This is because it stores string of word while I can create an array of type String that stores string of word in ShoutBehaviour.

Class Changes : ShoutAction

- Probability of Shout is not checked in this class.

New UML as Word class is removed.



Class Changes: Farmer

-Changed the attribute from FarmBehaviour Class to Behaviour[] behaviours, which store not only the Farmbehaviour but also the other behaviour that Human has so that when human has new behaviour, Farmer will also get these behaviours since Farmer is also a human.

Class Changes: Farmbehaviour

- Changed the getAction in behaviour to return actions such as SowAction, FertiliseAction, HarvestAction or Null.

Removed FarmAction and Added three different Action classes such as SowAction, FertiliseAction, HarvestAction.

New class added: SowAction,FertiliseAction,HarvestAction.

-Each Action class now does their own implementation in super class's execute method. SowAction checks the surrounding exit if it is standing next to a patch of dirt and has a 33% chance for Farmer to sow Crops. FertiliseAction checks if Farmer is standing on a unripe crop and will fertilise it. HarvestAction checks for the adjacent location to see if there exists a ripe crop.

Class Changed: Crop

-spawnWheat method changed to cropRipens method instead of creating a new object type Wheat, it will turn crop displayChar to 'C' to represent a ripe crop.