FIT2099 Assignment 1 Star War: Design Rationale

Loi Chai Lam (2813 6179)
Vinitha Raj Rajagopal Muthu (2824 9542)

1. Leave Affordance
   First, we create a class Leave under the package starwars.actions as Leave is an action. The class Leave allows the actor to put down the object they are currently carrying. The class Leave is inherit from SWAfordance as it is kind of Affordance. In the SWEntity classes, we need to add "Leave" into affordance so that when actor picks up the item, they can leave it. The Lightsaber and Blaster are inherited from SWEntity. When the actor leaves the item, we also need to remove the "Leave" affordance in the "act" method. The class Leave also modified the SWActor. In the class SWACtor, we need to create a method dropItemCarried to set the itemCarried by actor to NULL to show that the actor is holding nothing. Also, we need to use the class of SWWorld to get the current location of the actor, so that when they put down the object, the object will in the location of actor when it was done. We will get the location of the actor through method getEntityManager.whereIs(actor). The SWActor has implemented the interface of SWEntityInterface while SWActor also has its own SWWorld.

2. Force ability
   We create a new class TheForce under startwars package as TheForce is not an Entity nor Actor nor Action. SWActor has TheForce class, which means every actor has TheForce. If the actors have the ability to use Force, their forceAbility will set to True. Inside TheForce class, we have two attributes: Boolean forceAbility which indicates the actor's ability to use the force, int forceNumber which is the force of the actor. We have the getter and setter method for the forceNumber. Also, we can reset the forceAbility of the actor. Ben and Player can use the Force as specified in the requirement, so, they are shown in the UML diagram.

3. Lightsabres
   As specified in the requirement, people with a lot of Force ability can wield and use a Lightsabre as a weapon, hence, we create a new enum FORCEABLE in the class Capability. Lightsabre has capability; we change the capability of Lightsabre to FORCEABLE to indicate that Lightsabre is a weapon which needs Force. Also, we mention the forceNeeded by the Lightsabre in the LightSabre class. Lightsabre inherites from SWEntity and SWEntity implements interface SWEntityInterface. We need to modify the conditions in act method in Attack class so that only the people who have a lot of Forceability can use the Lightsabre to attack others. Class Action has SWActor; we need to use SWActor to check what the itemCarried by the actor. Ben and Player who have the Forceability are inherited from the SWActor. SWActor has class T heForce so that Ben and Player have their own Force.

4. Ben Kenobi

   For Ben Kenobi, we create a method “train (SWActor)” in the class SWLegend. As there could be many legends who can train Player.However, all legends will inherit the class SWLegend, so we decided to create the method in the parent class to avoid duplicate. Also, SWLegend and Player inherit from SWActor. From the UML, SWActor has their own SWWorld, hence, we can get the current location of the legend and player. If their locations are same, we will increase player’s force number since each SWActor has TheForce.

5. Droids

   For droids we a new class called droids was created which inherits from SWActor since droid is an actor too. A dependency was also created between droids and SWActor to signify that a droid is owned by an SWActor. A new method called isOwner was introduced to SWActor so that it can be known if an actor is an owner or not. Also, a dependency was created between SWActor and Move as a droid can only move in accordance with the owner’s location.