Design Rationale for req2

The diagram represents an object-oriented system for the purchase, drop and the runes managing.

Three classes extend Actor abstract class since there share common attributes and methods. This is also the same for the DeathAction and PurchaseAction class where both of the classes extend the Action abstract class. It obeys the DRY principle to reduce the repetition.

Trader class is introduced which allows player to purchase or sell items. This class has an association with WeaponItem class since Trader class has to manage the available weapon list. It also adds the PurchaseAction into the action list when player is nearby by calling the methods from PurchaseAction class. The trader is the object that gives the player an action to be caught. This helps with the Reduce Dependency Principle.

RuneManager class is also created to manage the runes of the player. It has an association with the player since each player has an instance of the rune manager to keep track of their runes. The DeathAction class is called when the enemy is killed by the player and it will drop runes to the player. Hence, reducing dependency between the DeathAction and RuneManager class.