Student: Orazgul Garyagdyyeva

ISTG 6010-01

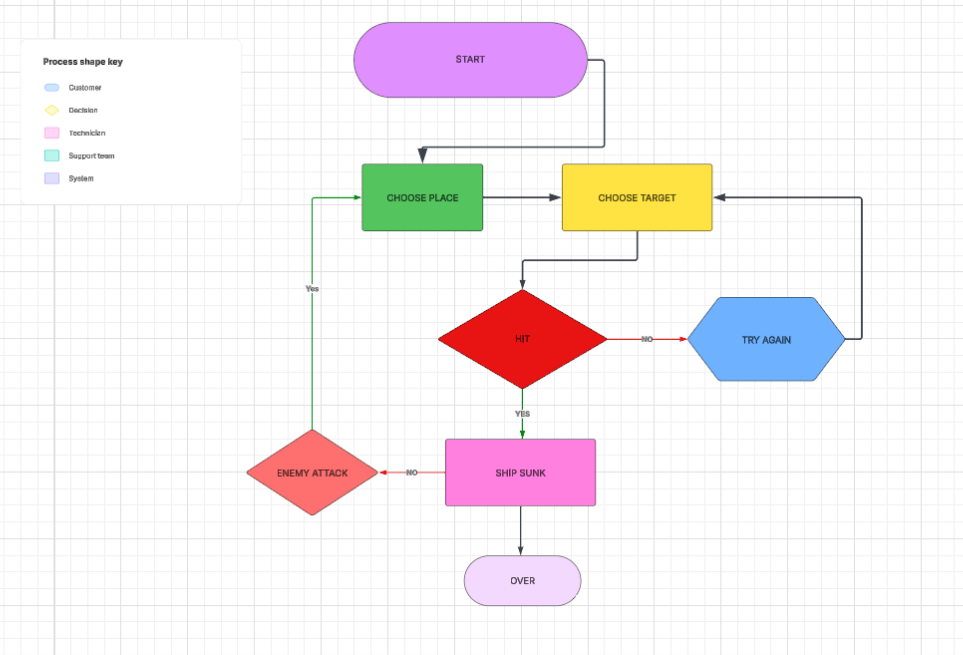
Object Oriented Systems

Prof Updyke, Dustin

9 April 2025

Battleship Game

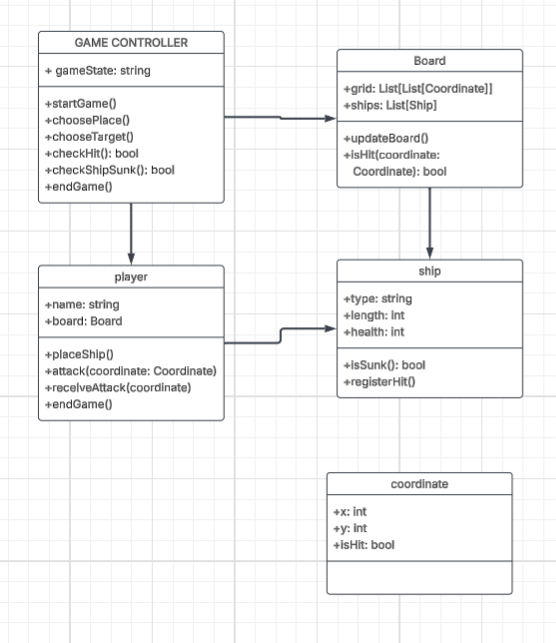
Flow Diagram



This Flow Diagram above represents a simplified use case flow for a RED player in the game Battleship. We better understand following a step-by-step sequence:

1. Start – The game begins, initializing the environment.  
 2. Choose Place – The player chooses where to position their ships on their grid.  
 3. Choose Target – Once setup is complete, the player selects a target location to fire at the enemy's grid.  
 4. Hit Evaluation – The system checks whether the selected target hits an enemy ship.  
 5. If Missed – The player must try again.  
 6. If Hit – The system checks if the ship has sunk.  
 7. Ship Sunk / Not Sunk – If a ship is sunk, the system records it. Otherwise, the enemy attacks.  
 8. Game Over – When one player's fleet is completely destroyed, the game ends.

UML DIAGRAM



This UML diagram is a blueprint of how the software system might be structured using object-oriented programming. It includes five core classes:  
   
1. Game Controller (Control Object) – Coordinates the game flow, including setup, turns, and checking for win conditions.  
 2. Player (Boundary Object) – Represents each player in the game. It interacts with the user or opponent AI and manages the player’s board.  
 3. Board (Entity Object) – Holds a player's grid and ship placements.  
 4. Ship (Entity Object) – Represents individual ships with attributes like length and health.  
 5. Coordinate (Entity Object) – Represents positions on the grid with hit status.

Technical Object-Oriented Analysis

Entity: Ship, Board, Coordinate – represent game data.  
 Boundary: Player – interface between user actions and system logic.  
 Control: GameController – orchestrates the flow of the game.  
   
 The class structure supports principles of object-oriented programming:  
 Encapsulation – Each class keeps its behavior and data contained.  
 Abstraction – The diagram hides inner mechanics but shows clear roles.  
 Reusability – Code built on this model can be reused in variations of the game.