The Legend of Beowulf Game Application

Alfonso Avila
Esteban Rodriguez
Iacopo Nohea Lenzi
Vladimir Veillard

Function Specification

Platform: Java Swing
OS : Windows

1. **Application Initialization**
    - When the application starts, the Main Menu options are displayed:
        - New Game
        - Load Game
        - Settings
    -
    - Loading screens could be an image of Beowulf in action

2. **Main Menu**
    - The main menu panel displays buttons for:
        - Buttons Width x Height = (250px, 100px)
        - Starting a new game.
        - Loading a save file
        - Settings
        - Exit Game

3. **New Game Setup**
    - When the player selects “New Game,” a new game setup panel opens.
    - Player can set:
        - Player name
        - Game difficulty
    - Once configured, the new game setup panel closes.

4. **Load from Save File**
    - When the player selects “Load from Save File” the game will continue from the last checkpoint.
    - If there are no saved files, present “No saved game” error message.

5. **Settings Menu**
    - Player will have option to change settings such as
        - Music Volume
        - Game Sound Volume
6. **Exit Game**
    - Closes the application

7. **Gameplay Panel**
    - Upon making a selection the gameplay panel is displayed
    - The gameplay panel contains:
        - Animations
        - Tracking of player inputs

8. **In Game Options Drop Down**
    - In the gameplay panel there will be a drop down menu where the player can:
        - Save the current checkpoint
        - Exit program
        - Edit volume settings

9. **Input Handling**
    - During gameplay, input is tracked for:
        - Player movement
        - Interactions with chests
        - Player interactions with enemies (attacks)
    - Main player movement controls:
        - w:: Up - updates and reprints character sprite on (x axis - 1)
        - a:: Left - updates and reprints character sprite on (x axis -1)
        - s:: Down - updates and reprints character sprite on (y axis +1)
        - d:: Right - updates and reprints character sprite on (x axis +1)
    - Main player interact controls:
        - e:: Interact

    - Main player Attack controls:
        - o :: Attack
        - p :: Special Attack
    - Inventory Open

        - m :: Inventory
    - Drop Item
        - n :: drop

10. Player Information Panel

- At the top of the screen, a panel displays player information:
    - Health Bar
    - Weapons in Inventory
    - Game difficulty
    - In game currency

Use Cases

Menu

Use Case - Starting New Game

<<Actor is Player>>

1. Window displays Main Menu options are displayed
    - New Game
    - Load Game
    - Settings
2. Player selects New Game
3. System displays character creation window displays
    - Text Field: *Enter Name*
    - Button selection : “Easy” , “Medium”, “Hard”
    - Button : “Start Journey”
4. Player enters name and selects difficulty then pressed “start journey”.
5. Gameplay is initialized

Use Case - Load From Save File

<<Actor is Player>>

1. Window displays Main Menu options are displayed
    - New Game
    - Load Game
    - Settings

2. Player selects Load Game
3. System plays Loading Screen while Gameplay is initialized
4. Gameplay starts

Use Case - Load From Save File Var#1 Save File Not Found

<<Actor is Player>>

1. Starts at Use Case - Load From Save File step 2.
2. No existing Save Files found, Load Game button turned off.
3. Start at Use Case - Load From Save File Step 1

Use Case - Change Settings

<<Actor is Player>>

1. Window displays Main Menu options are displayed
   - New Game
   - Load Game
   - Settings
2. Player selects Settings
3. Settings Panel is initialized and displays
   - Music Icon
   - -+Game Sound Icon
4. Player clicks Music Icon
5. Icon shows muted Music Icon, music volume turns off.
6. Player clicks Game sound Icon
7. Icon shows muted Game Sound Icon, Game Sound turns off.
8. Player exits settings menu clicking on “Done” button

Player Movement

Use Case - In Game Movement

<<Actor is Player>>

1. Player uses arrow keys or WASD keys to move the character in the game world.
2. Player presses ‘w’:: Up - updates and reprints character sprite “upward facing” on (x axis - 1)
3. Player presses ‘a’ :: Left - updates and reprints character sprite “left facing” on (x axis -1)

4. Player ‘s’:: Down - updates and reprints character sprite “downward facing” on (y axis +1)
5. Player ‘d’ :: Right - updates and reprints character sprite “right facing” on (x axis +1)
6. Character's position is updated on the screen based on player input.
7. Collision detection checks for obstacles or boundaries to prevent the character from moving through walls or off-screen.
8. Character animation changes based on movement direction.

Player Attack

Use Case - In Game Attack

<<Actor is Player>>

1. Player initiates an attack action by pressing the designated key: o.
2. Game calculates the attack's accuracy and damage based on character stats and target.
3. Animation and visual effects are displayed to represent the attack.
4. Damage is applied to the target's health points (HP) if the attack is successful.

Player Interactions

Use Case - In Game Interaction with NPC

<<Actor is Player>>

1. Player approaches a non-player character (NPC)
2. Player presses ‘e’ to interact, key tracking for movement, toggles off
3. Interaction options are displayed, such as "Talk" or "Take", “Nevermind”
4. In option selection Player can toggle between options using ‘a’ and ‘d’ , then pressed enter
5. Player selects ‘Talk’ , dialogue is displayed, dialogue closes , game is resumed, key tracking for movement toggles back on.
6. Step 3 Player selects ‘Take’, item added to inventory., game is resumed, key tracking for movement toggles back on.
7. Step 3 Player selects “Nevermind” , game is resumed, key tracking for movement toggles back on.

Use Case - In Game Interaction with NPC Var#1 No Item To Take

<<Actor is Player>>

1. Starts at Use Case - In Game Interaction with NPC step 6.

2. NPC does not have an item to take, displays dialogue “Nothing to Take”.
3. Game is resumed, key tracking for movement toggles back on.

Use Case - In Game Interaction with NPC Vendor Var# Player Buys Item

<<Actor is Player>>

1. Player Character Sprite approaches a non-player character (NPC) vendor
2. Player presses ‘e’ to interact, key tracking for movement, toggles off
3. Interaction options are displayed, such as "Talk" or "Shop", “Nevermind”
4. In option selection Player can toggle between options using ‘a’ and ‘d’ , then pressed enter
5. Player selects ‘Talk’ , dialogue is displayed, dialogue closes , game is resumed, key tracking for movement toggles back on.
6. Step 3 Player selects Shop, Shop panel is initialized and displays:
    - Buy
    - Sell
7. Player selects ‘Buy’, NPC inventory is displayed with prices and Player can browse items.
8. Player selects an item and item amount, then confirms purchase.
9. Item is added to Player inventory and in-game currency is removed from Player Wallet / Inventory
10. Player selects ‘Exit Shop’, game is resumed, key tracking for movement toggles back on.

Use Case - In Game Interaction with NPC Vendor Var# Player Sells Item

1. Player Character Sprite approaches a non-player character (NPC) vendor
2. Player presses ‘e’ to interact, key tracking for movement, toggles off
3. Interaction options are displayed, such as "Talk" or "Shop", “Nevermind”
4. In option selection Player can toggle between options using ‘a’ and ‘d’ , then pressed enter
5. Player selects ‘Talk’ , dialogue is displayed, dialogue closes , game is resumed, key tracking for movement toggles back on.
6. Step 3 Player selects Shop, Shop panel is initialized and displays:
    - Buy
    - Sell
7. Player selects ‘Sell’, Player inventory is displayed.
8. Player selects Item to sell and NPC offers a price, Player confirms sale of Item.
9. Item is removed from Player inventory, and in-game currency is added to Player Wallet / Inventory.
10. Player selects ‘Exit Shop’, game is resumed, key tracking for movement toggles back on.

Use Case - In Game Interaction with NPC Vendor Var# Player Doesn’t Shop

1. Player Character Sprite approaches a non-player character (NPC) vendor
2. Player presses ‘e’ to interact, key tracking for movement, toggles off
3. Interaction options are displayed, such as "Talk" or "Shop", “Nevermind”
4. In option selection Player can toggle between options using ‘a’ and ‘d’ , then pressed enter
5. Step 3 Player selects “Nevermind” , game is resumed, key tracking for movement toggles back on.

Use Case - In Game Interaction with Item

<<Actor is Player>>

1. Player Character Sprite approaches Item
2. Player presses ‘e’ when facing item
3. Item is added into Players inventory, Item removed from game only exists in inventory

Use Case - In Game Interaction with Chest

<<Actor is Player>>

1. Player Character Sprite approaches Item, only interactable when front facing
2. Player presses ‘e’ to interact with Chest.
3. Dialogue displays “You found ‘Item’”, Item placed into inventory,
4. Player pressed enter to close Dialogue, game resumes

Use Case - In Game Interaction with LuckyChest

<<Actor is Player>>

1. Player Character Sprite approaches Item, only interactable when front facing
2. Player presses ‘e’ to interact with Chest., Chest returns random Item
3. Dialogue displays “You found ‘Item’”, Item placed into inventory,
4. Player pressed enter to close Dialogue, game resumes

In Game Menu

Use Case - Save Game (Game Menu)

<<Actor is Player>>

1. Player presses “escape” key, brings up in Game Menu, Game Menu displays buttons
   - Save Game
   - Settings
   - Exit Game
   - Return to Game
2. Player selects "Save Game" from the in-Game Menu.
3. System displays a list of available save slots or allows the player to choose a save file name.
4. Player selects a save slot or enters a file name and confirms the save.
5. Game state, including player progress and current position, is saved to the chosen file.
6. Confirmation message is displayed to the player. Returns to in-Game Menu
7.

Use Case - Exit Game (Game Menu)

<<Actor is Player>>

1. Player presses “escape” key, brings up in Game Menu, Game Menu displays buttons
   - Save Game
   - Settings
   - Exit Game
   - Return to Game
2. Player selects "Exit Game" from the in-Game Menu.
3. System displays a confirmation prompt “Are you sure you want to Exit?, Any unsaved data will be lost”
4. System displays button options “Yes” , “No”
5. Player presses “Yes”.
6. Game is closed, and the player returns to the main menu.

Use Case - Exit Game (Game Menu) Var# Player Doesn’t Want to Exit

<<Actor is Player>>

1. Starts at Use Case - Exit Game (Game Options) step 4
2. Player selects “No”
3. Returns to Game Options menu

Use Case - Change Settings (Game Options)

<< Actor is Player>>

1. Player presses “escape” key, brings up in Game Menu, Game Menu displays buttons
    - Save Game
    - Settings
    - Exit Game
    - Return to Game
2. Payer selects “Change Settings” from the in-game menu.
3. Settings Panel is initialized and displays
    - Music Icon
    - Game Sound Icon
4. Player clicks Music Icon
5. Icon shows muted Music Icon, music volume turns off.
6. Player clicks Game sound Icon
7. Icon shows muted Game Sound Icon, Game Sound turns off.
8. Player exits settings menu clicking on “Done” button

Use Case - Game Menu Not Used

<<Actor is Player>>

1. Player presses “escape” key, brings up in Game Menu, Game Menu displays buttons
    - Save Game
    - Settings
    - Exit Game
    - Return to Game
2. Player presses Return to Game , Game resumes

Design Specification

Alfonso Avila
Esteban Rodriguez
Iacopo Nohea Lenzi
Vladimir Veillard

CRC Cards :

MainSystem -

- Initializes game

- Shows the display : MainDisplay
- Runs Gameplay loop :GamePlay

MainDisplay -

- Displays the Main Menu : MainMenu
- Displays the GamePlay : GamePlay
- Displays the GameMenu : GameMenu

MainMenu -

- Handles Option Selection
- Displays the NewGame : NewGame
- Displays the LoadGamee : LoadGame
- Displays the Settings : Settings
- Takes Option for Game Exit

NewGame -
- Takes Input for the New Player :Player
- Takes the Input Selection for the
  Game difficulty :GamePlay
- Starts GamePlay

LoadGame -
- Initializes gameplay from save data : SaveData
- Takes input from save data for the Player : Player
- Starts GamePlay : GamePlay

Settings -
- Takes input selections for volume : Volume

Volume -
- Handles change in Music sound and Game Sound

SaveData -
- Writes data into a file
- Reads data from a file

GamePlay (JPanel) -

- Handles Gameplay loop : Player

- Handles inputs from keyboard : Movement

GameMenu -

- Handles input form keyboard : Movement
- Handles Change in Volume   : Volume
- Handles Save Selection   : SaveData
- Handles Exit

Player -

- Takes Input from Keys : Movement
- Updates the Player Sprite

Movement -

- Handles inputs from keyboard
- Notifies Player of Position : Player

Attack -

- Initiates attack decisions based on player input: Player
- Calculates accuracy and damage based on character stats and target: Enemies
- Manages animation and visual effects for attacks
- Applies damage to the target's health points (HP) if the attack is successful : Enemies

Interactions -

- Handles interactions between the player and various in-game elements: Player, NPCs, Enemies, Item, Chest
- Manages dialogues and options during interactions: Player
- Processes player choices and updates the game state accordingly: Player

NPCs -

- Represents non-player characters in the game world: GamePlay, Enemies
- Provides dialogues and interaction options to the player: Player, Interactions
- Manages NPC behavior and responses based on player interactions: Player, Interactions

Enemies -

- Represents enemy characters in the game world   : NPCs, GamePlay
- Manages enemy behavior, movement, and attacks: Interactions, Attack
- Calculates damage received from player attacks   : Attack
- Provides rewards upon defeat: Chest

Item -

- Represents in-game items that can be collected, bought, or sold
- Manages item properties, such as name, description, and effects
- Handles item interactions, including adding/removing from player inventory: Player, Interactions

Weapon -

- Changes player damage output : Player

Chest -

- Represents in-game chests that contain items or rewards: Item
- Handles chest interactions, including opening and obtaining items: Player, Interactions, Item

Inventory -

- Manages items that the player has : Item

KeyPad -

- Takes the inputs and sets the uses : Interaction
  : Attack
  : Inventory
  : Movements

WeaponSelect -

- Toggles through weapons : weapon

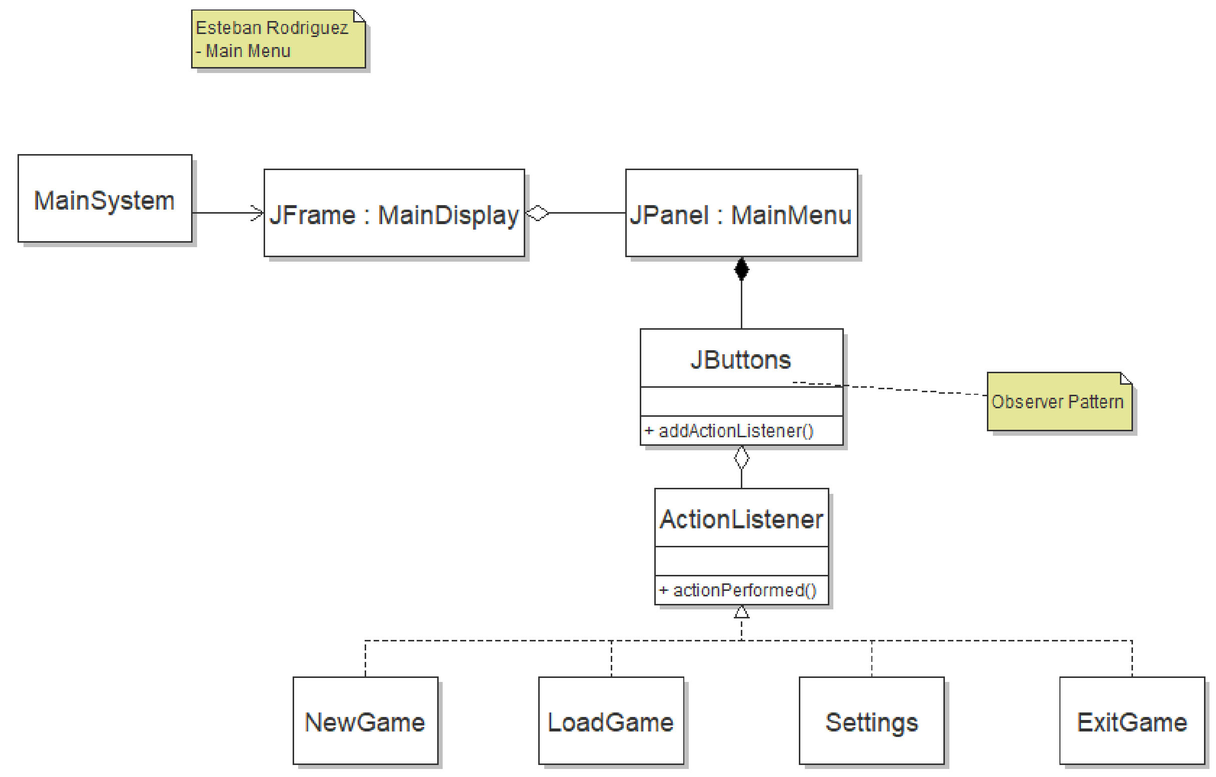
Levels

- Contain map/bounds
- Progresses story

UML Diagrams:

Class Diagrams:

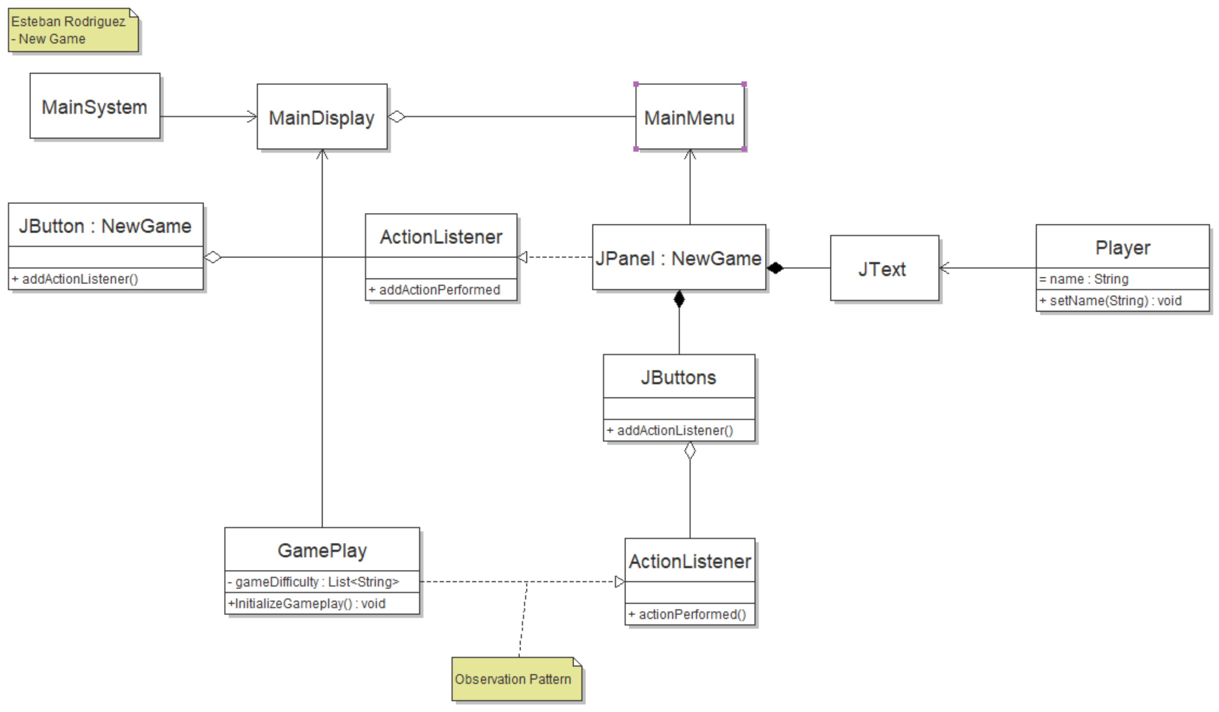
MainMenu Class Diagram -



Observation Pattern

subject	JButton
observer	ActionListener
concrete observer	NewGame, LoadGame, Settings
attach()	addActionListener
notify()	actionPerformed e

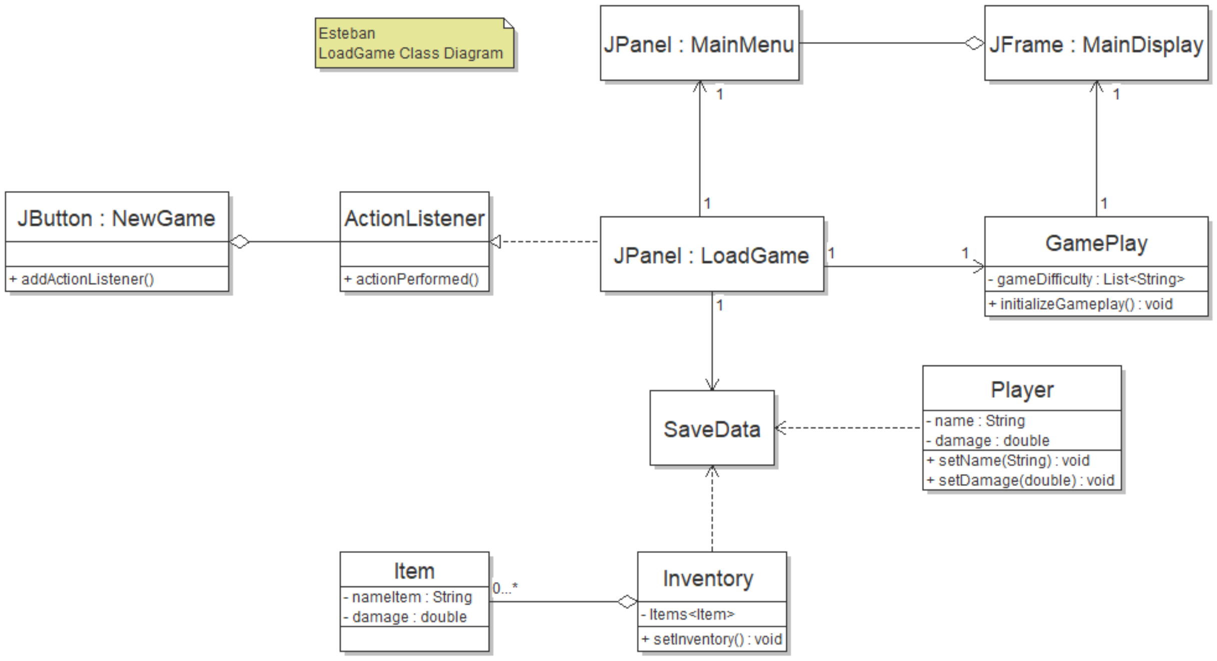
NewGame Class Diagram -



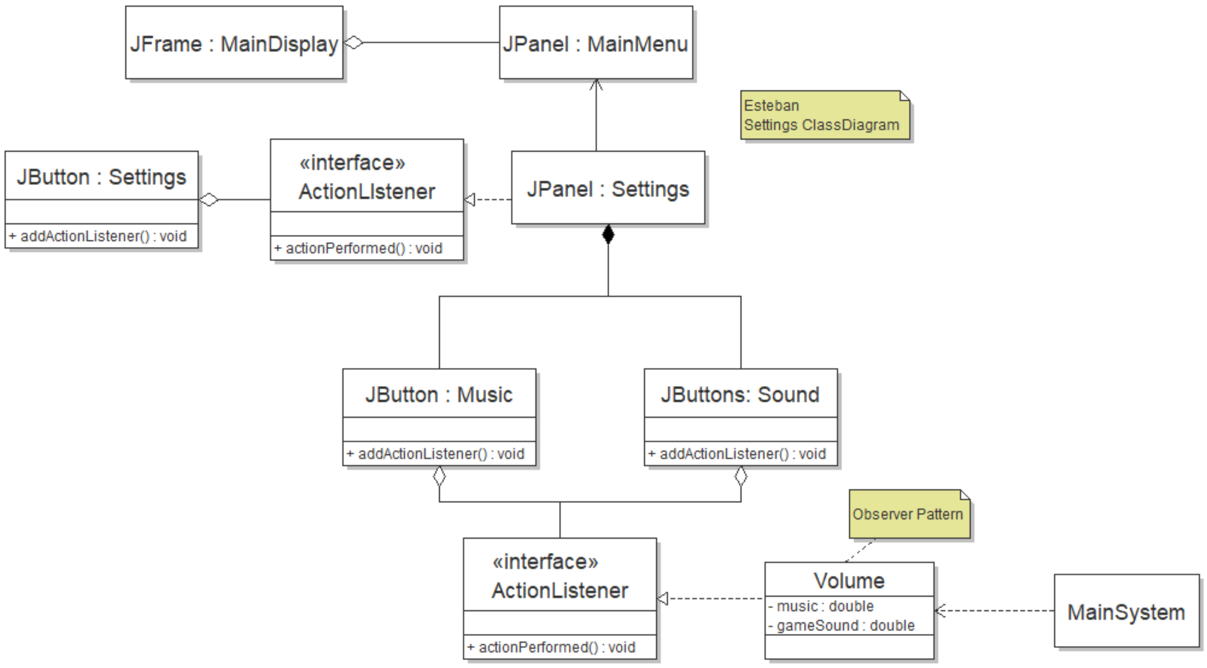
Observation Pattern

Subject	JButton
Observer	ActionListener
Concrete observer	Gameplay
attach()	addActionListener
notify()	actionPerformed e

LoadGame Class Diagram -



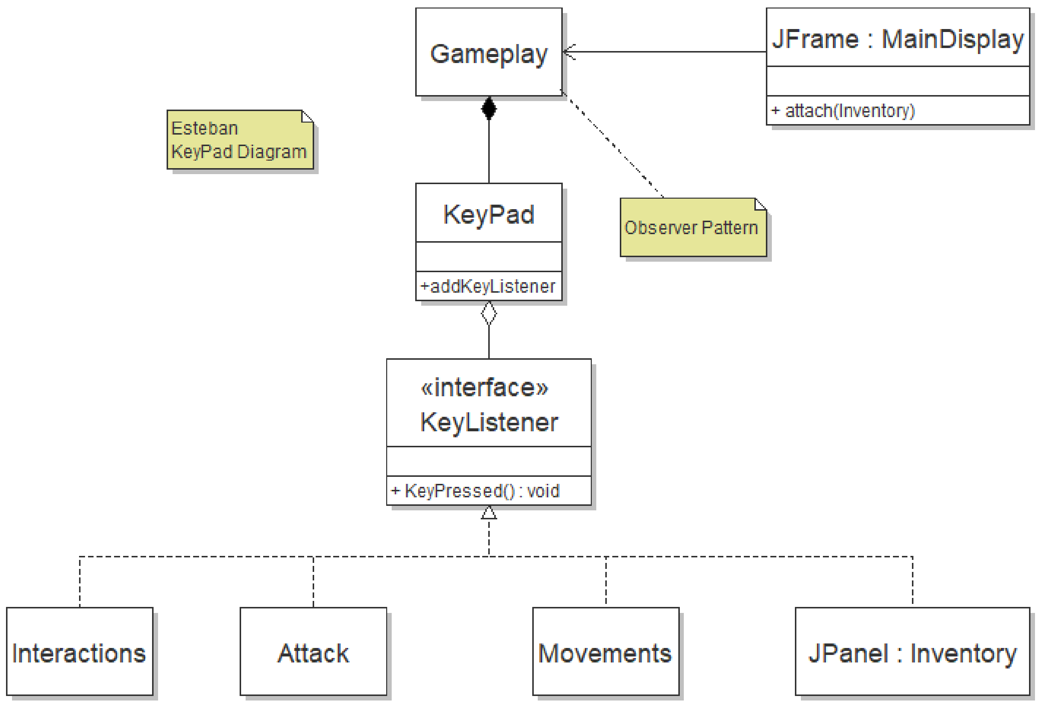
Settings Class Diagram -



Observation Pattern

Subject	JButton: sound, music
Observer	ActionListener
Concrete observer	Volume
attach()	addActionListener
notify()	actionPerformed e

KeyPad Class Diagram -

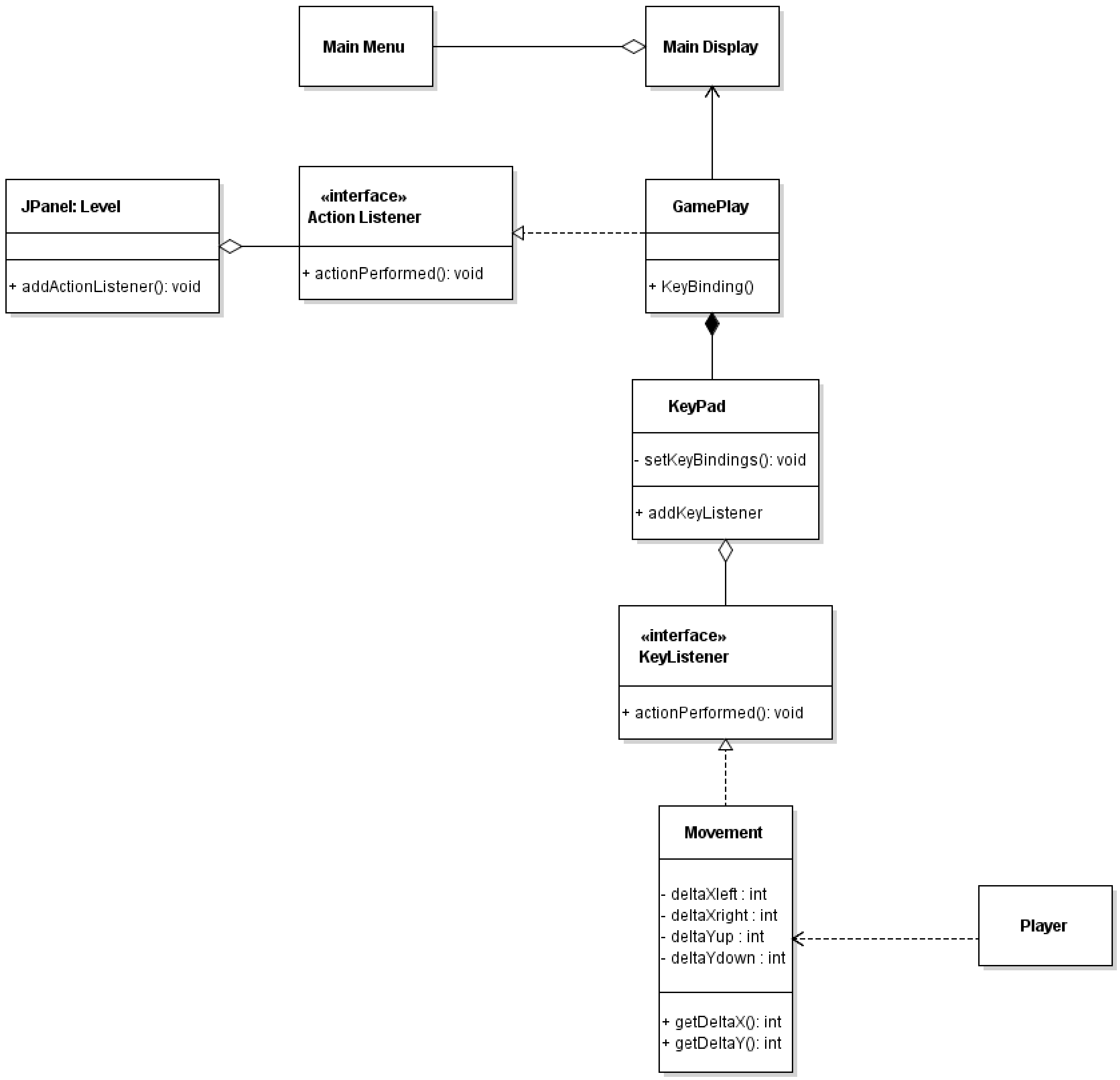


Observation Pattern

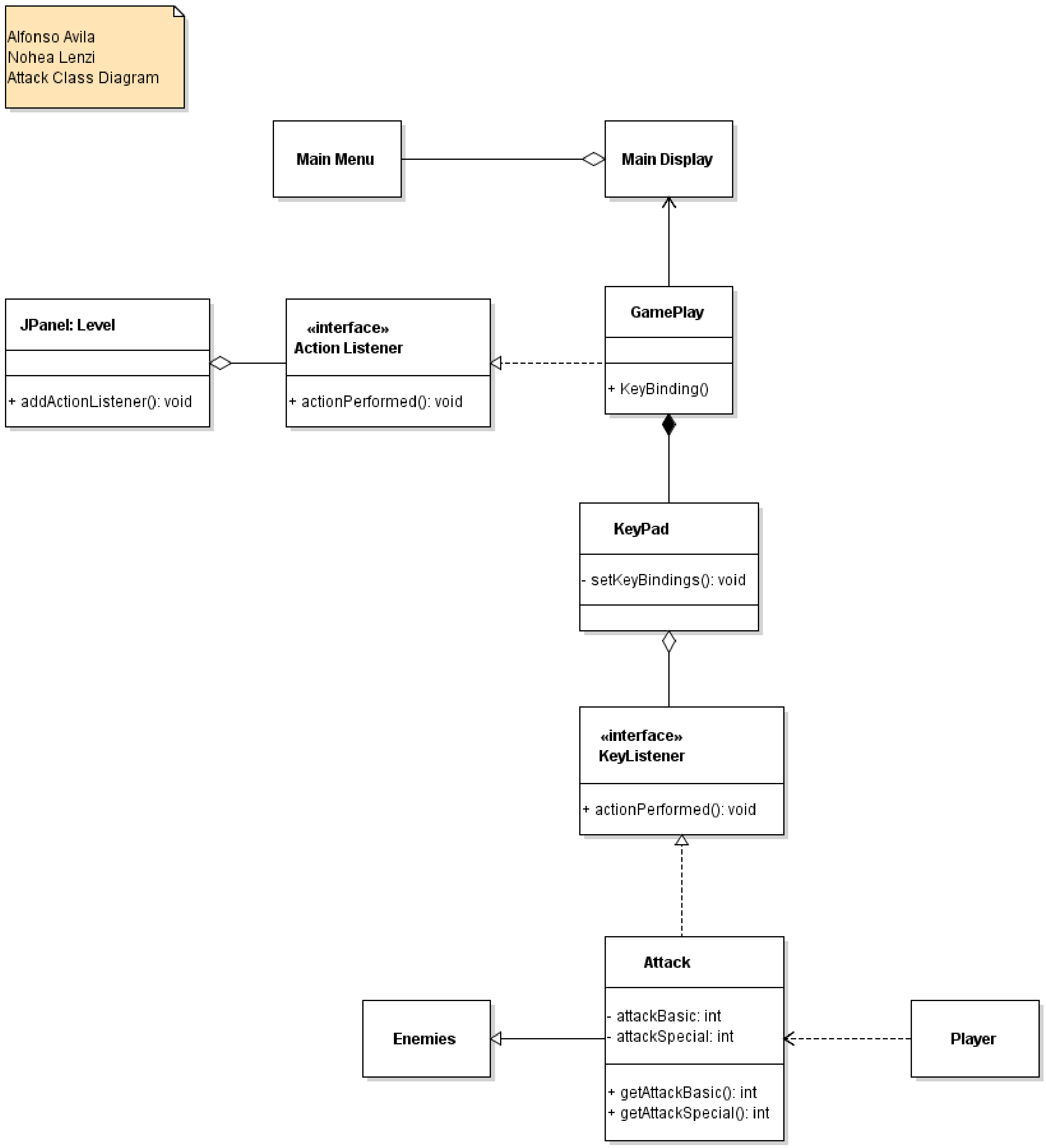
Subject	JButton : sound, music
Observer	ActionListener
Concrete observer	Volume
attach()	addActionListener
notify()	actionPerformed e

PlayerMovement Class Diagram -

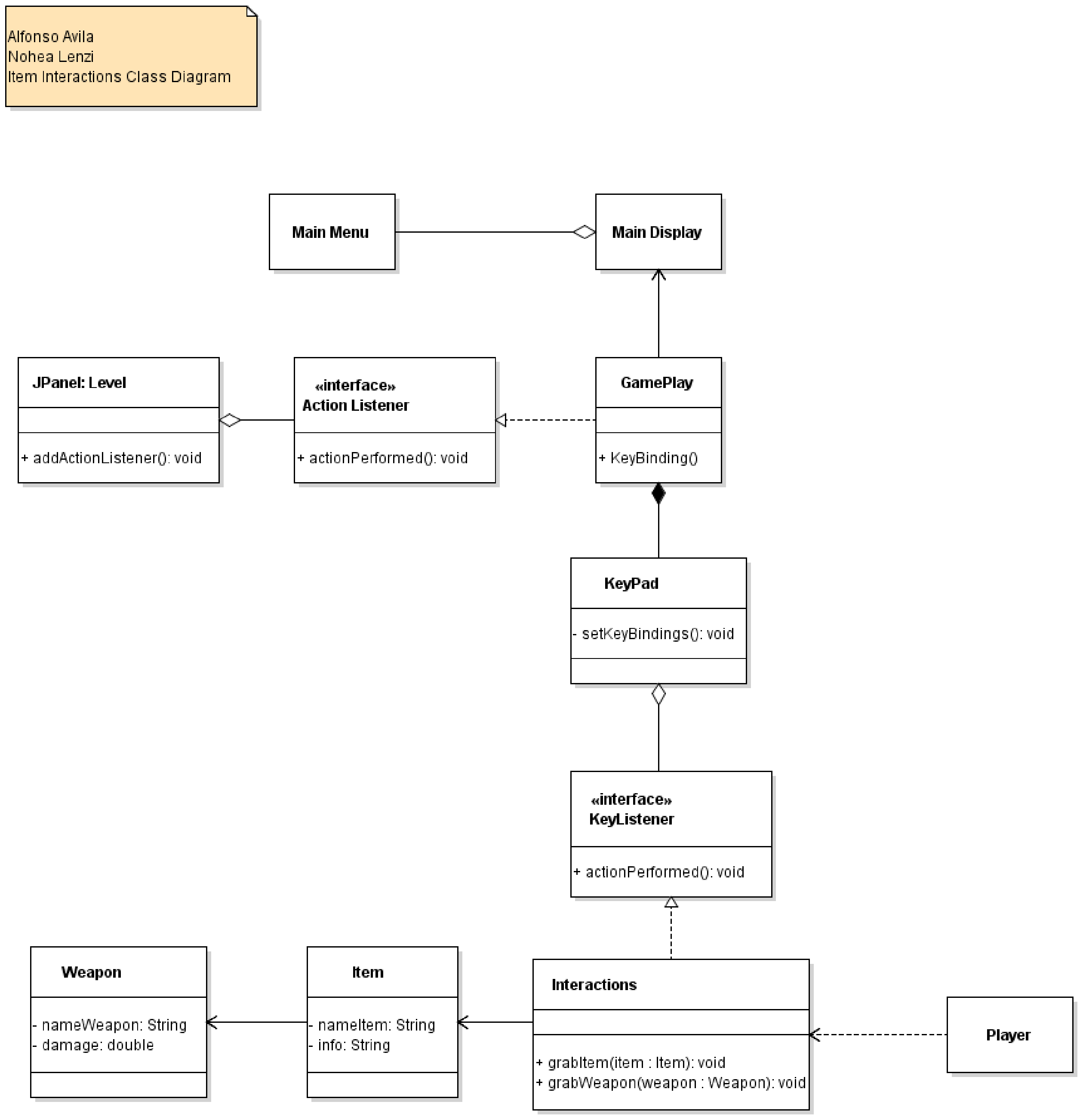
Alfonso Avila
Nohea Lenzi
Player Movement Class Diagram



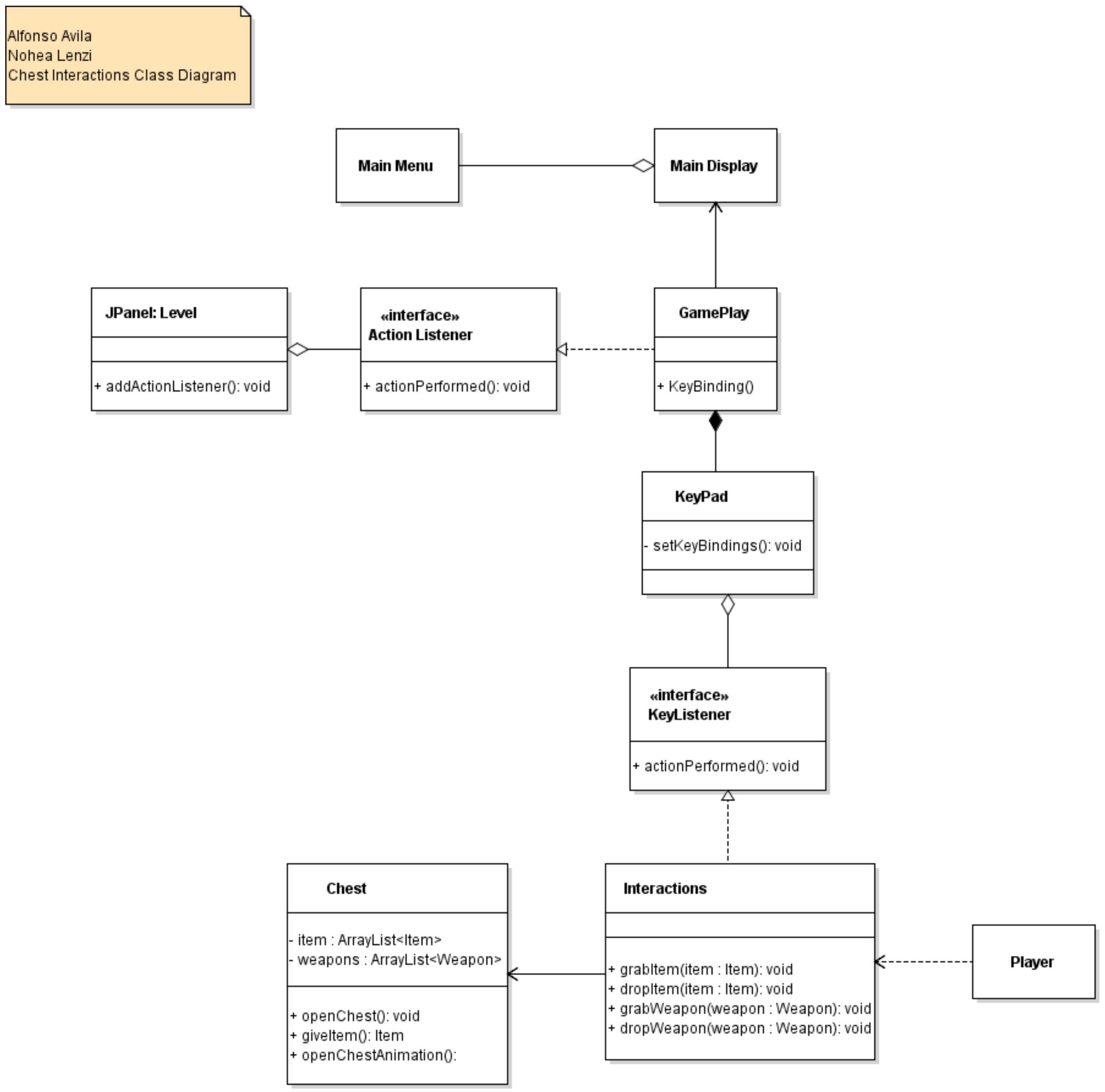
Attack Class Diagram -



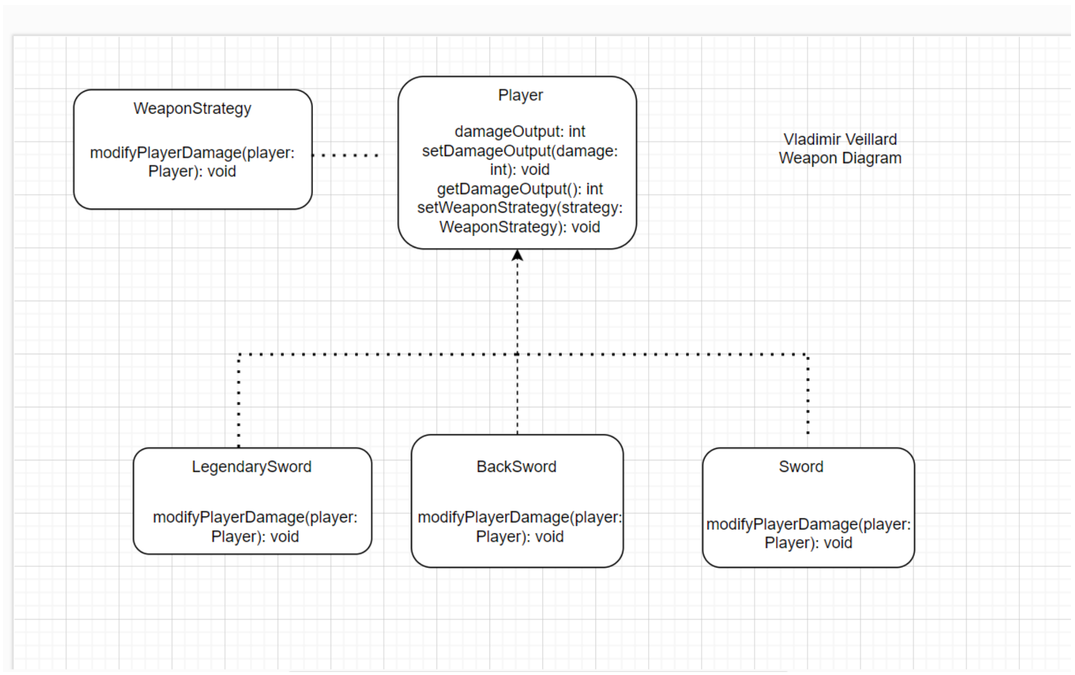
Item Interaction Class Diagram -



Chest Interaction Class Diagram -


Alfonso Avila
Nohea Lenzi
Chest Interactions Class Diagram
Main Menu
Main Display
JPanel: Level
+ addActionListener(): void
«interface» Action Listener
+ actionPerformed(): void
GamePlay
+ KeyBinding()
KeyPad
- setKeyBindings(): void
«interface» KeyListener
+ actionPerformed(): void
Chest
- item : ArrayList<Item>
- weapons : ArrayList<Weapon>
+ openChest(): void
+ giveItem(): Item
+ openChestAnimation():
Interactions
+ grabItem(item : Item): void
+ dropItem(item : Item): void
+ grabWeapon(weapon : Weapon): void
+ dropWeapon(weapon : Weapon): void
Player

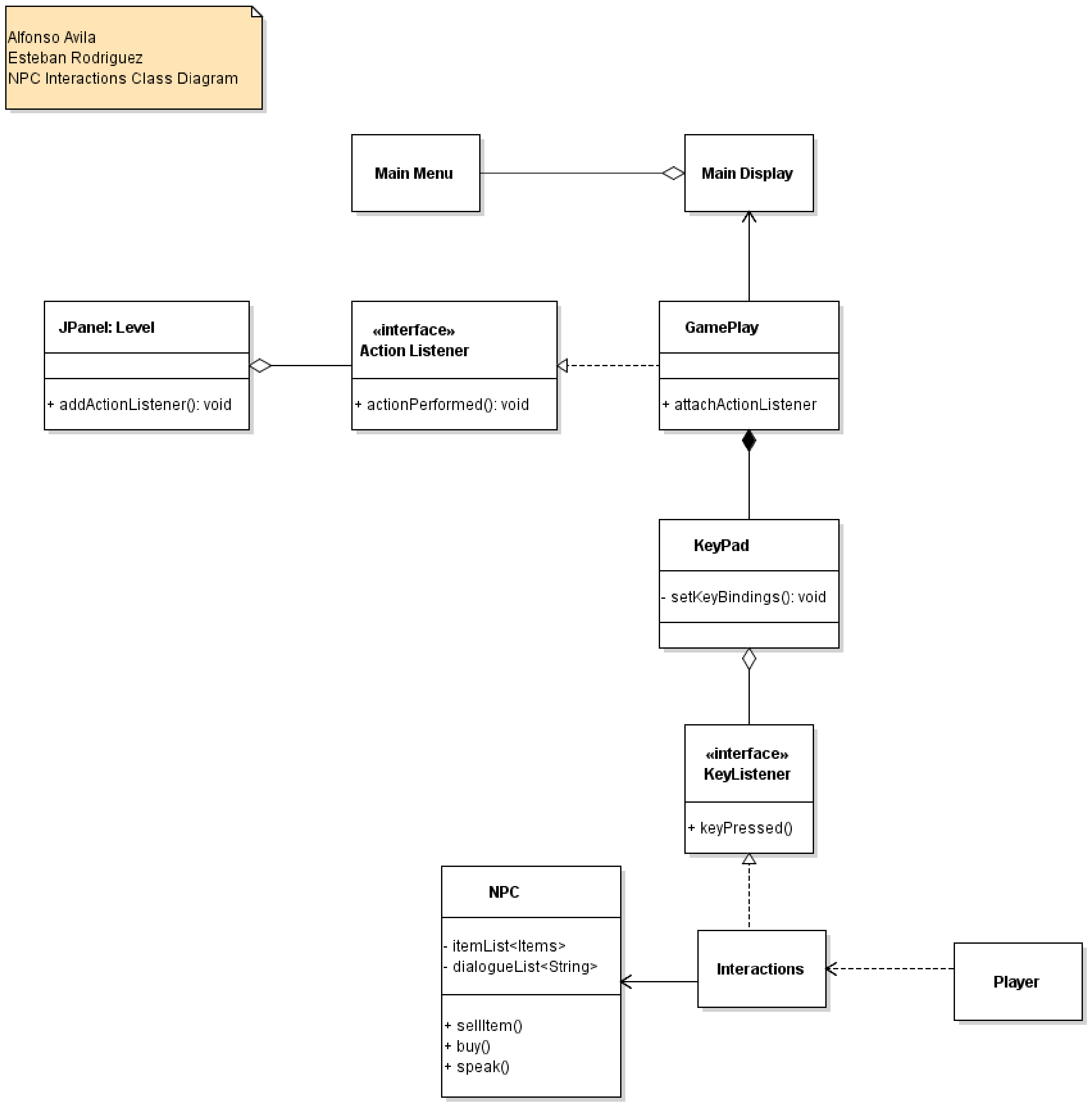
Weapon Class Diagram:


WeaponStrategy
modifyPlayerDamage(player: Player): void
Player
damageOutput: int
setDamageOutput(damage: int): void
getDamageOutput(): int
setWeaponStrategy(strategy: WeaponStrategy): void
Vladimir Veillard
Weapon Diagram
LegendarySword
modifyPlayerDamage(player: Player): void
BackSword
modifyPlayerDamage(player: Player): void
Sword
modifyPlayerDamage(player: Player): void

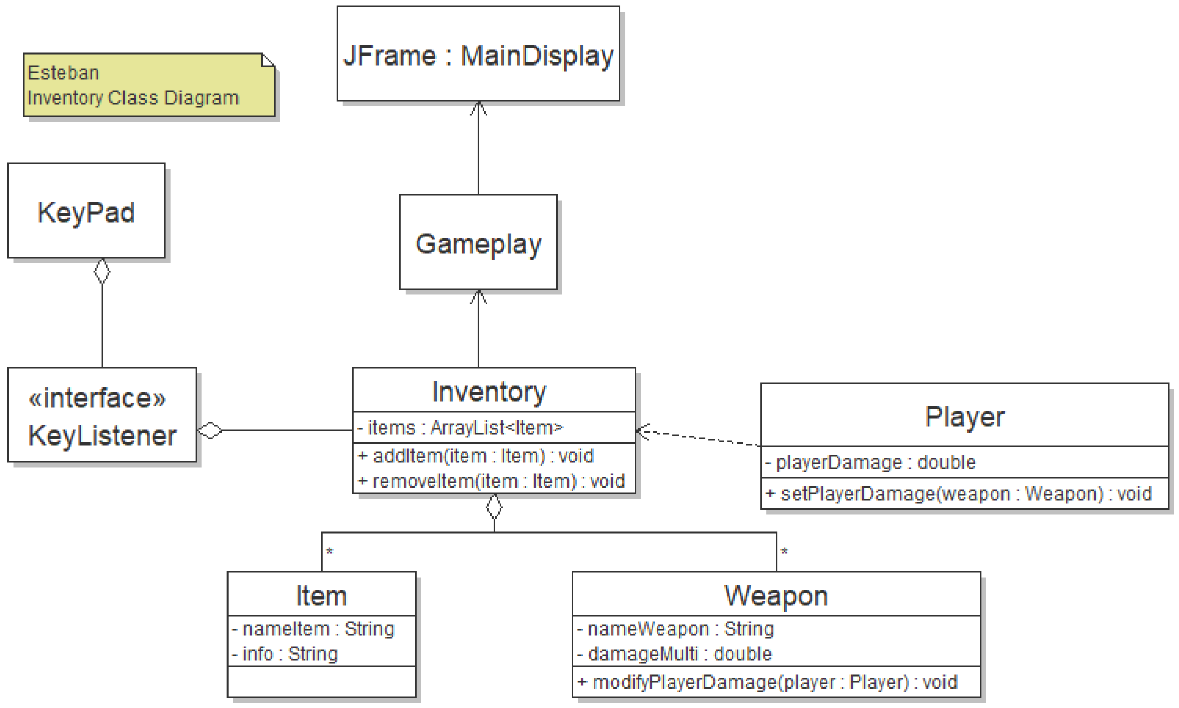
Strategy Pattern

Context	Player
Strategy	WeaponStrategy
Concrete Strategy	ShortSword, MidSword, LongSword
doWork()	modifyPlayerDamage(player: Player) void

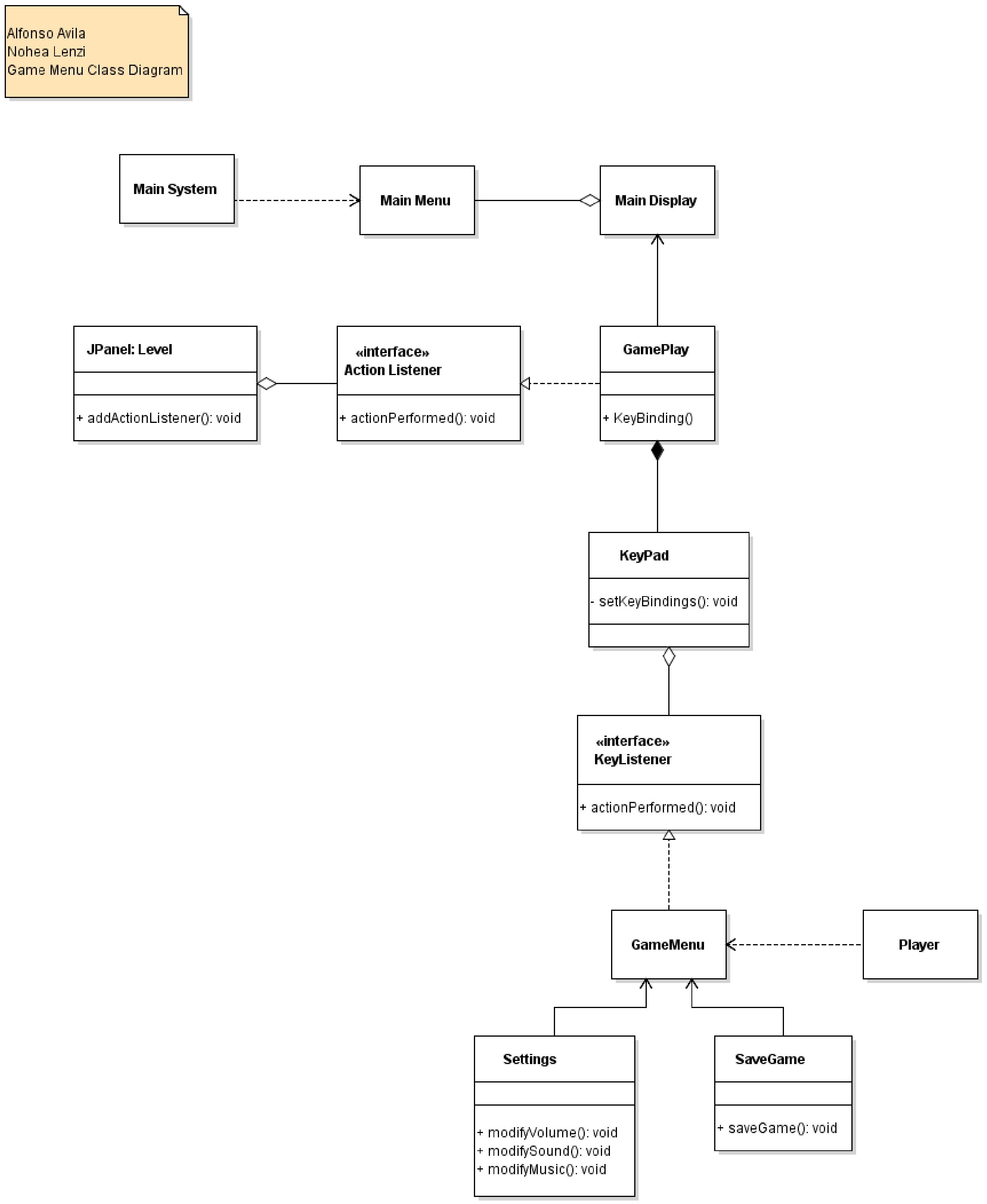
NPC Interaction Class Diagram -



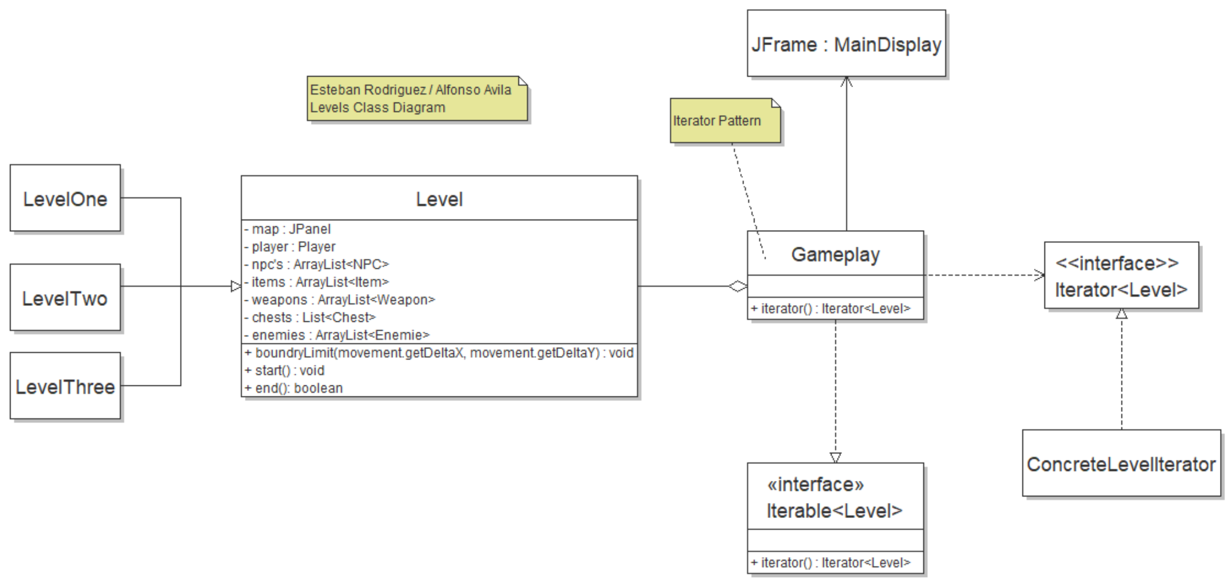
Inventory Class Diagram -



GameMenu Class Diagram -



Levels Class Diagram -



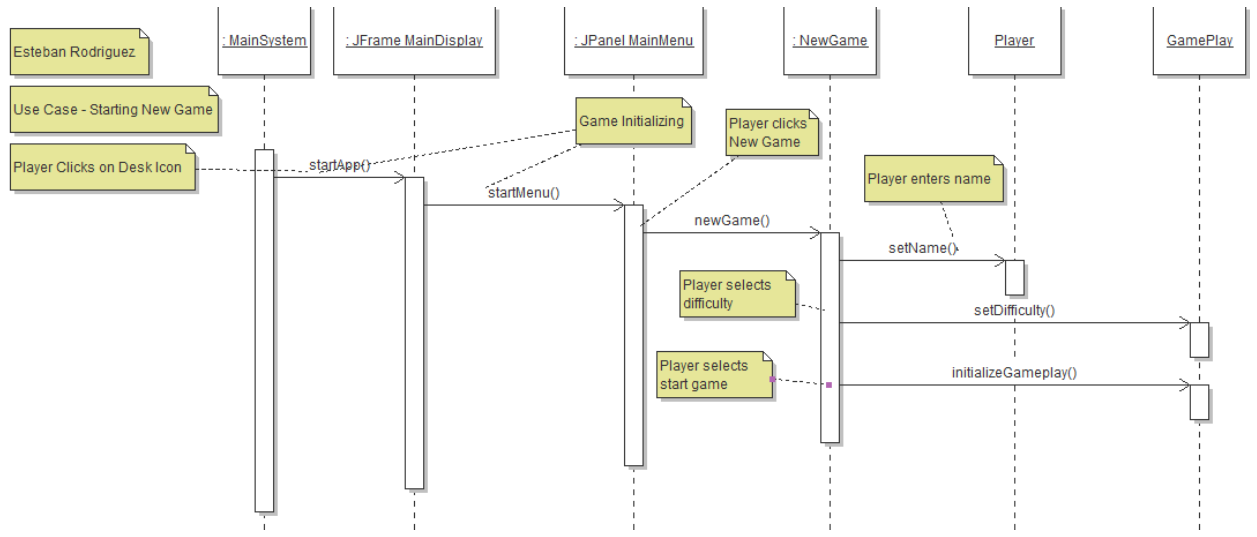
Iterator Pattern

Aggregate	iterable<Level>
Concrete Aggregate	Gameplay
Iterator	Iterator<Level>
Concrete Iterator	ConcreteLevelIterator
createIterator()	listIterator()
next()	next()
isDone()	Opposite hasNext()
currentItem()	Return hasNext()

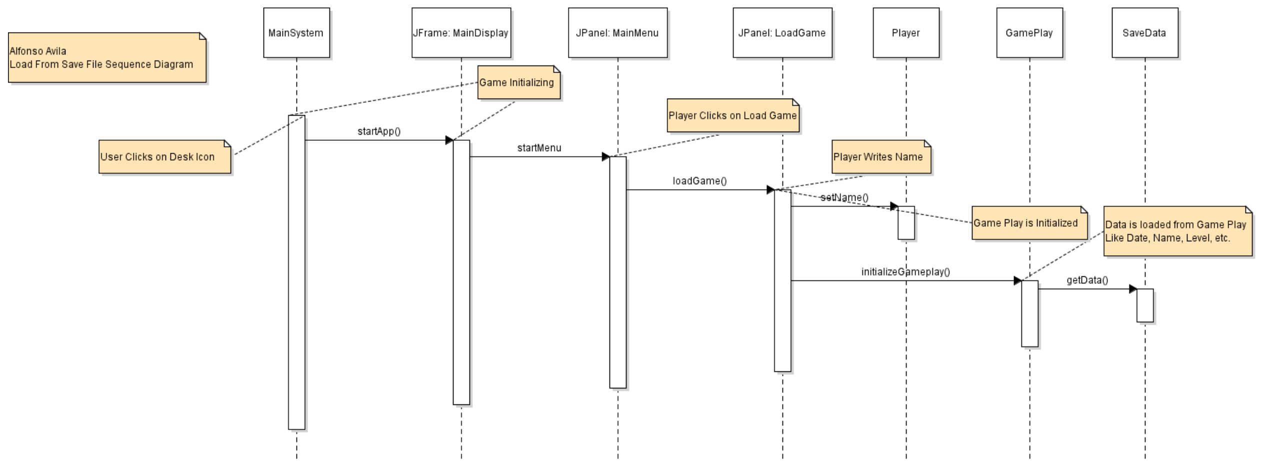
Sequence Diagrams:

Menu Sequence Diagrams:

Use Case - Starting New Game

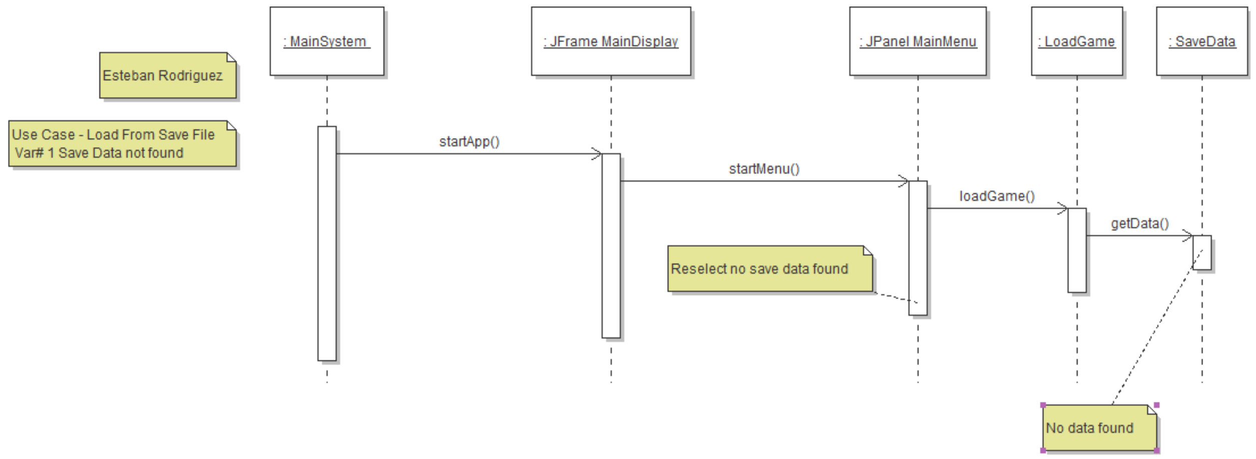


Use Case - Load From Save File

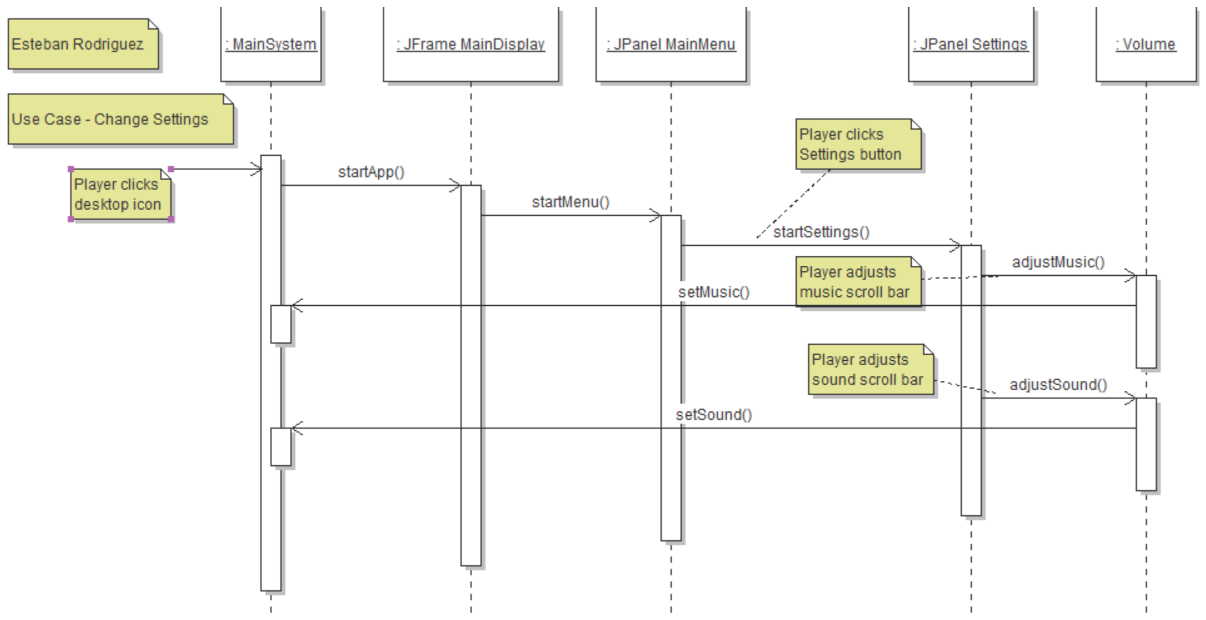


Wouldn’t player name already be saved in the saved game data

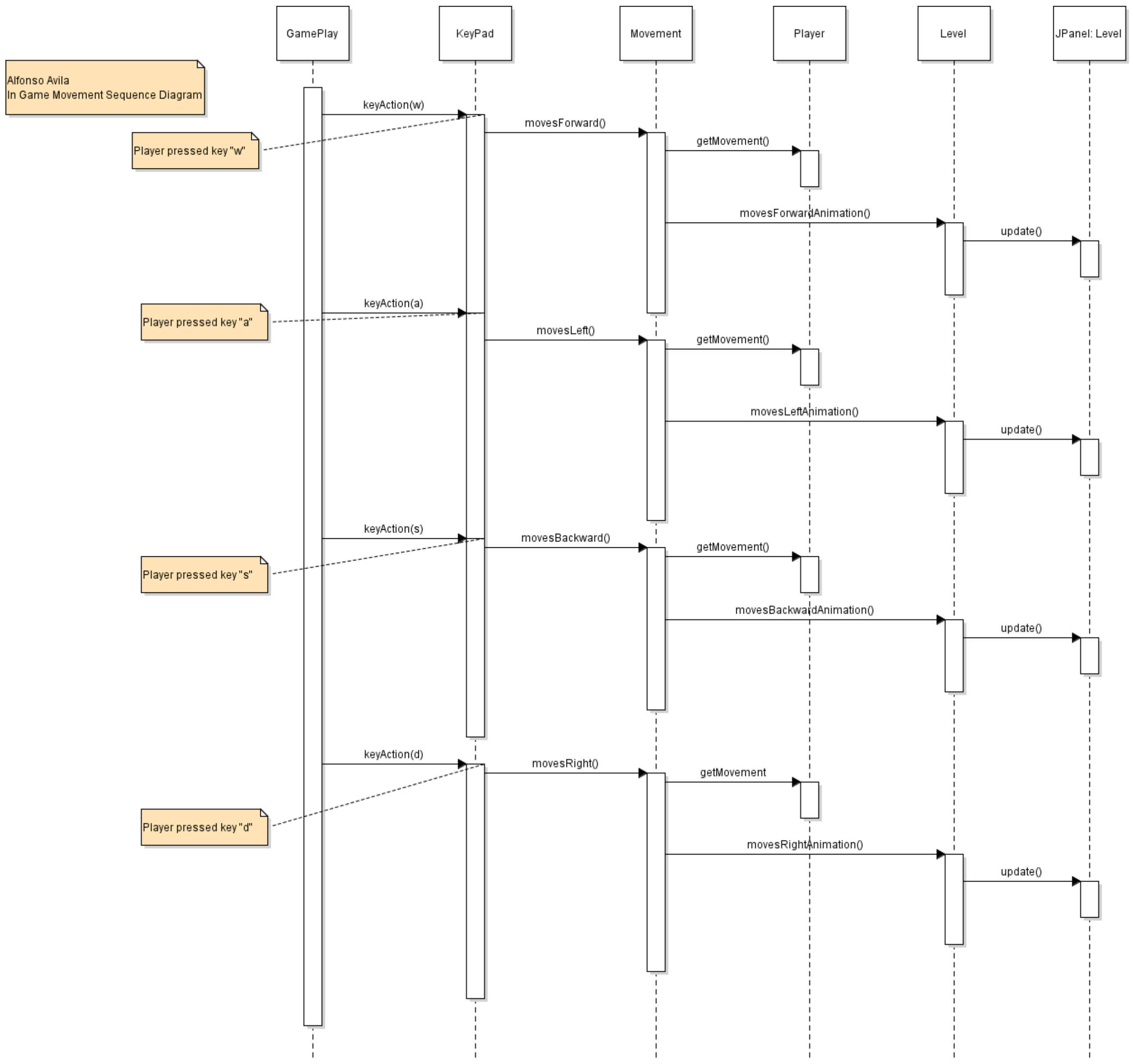
Use Case - Load From Save File : Variation #1 Save Data Not Found



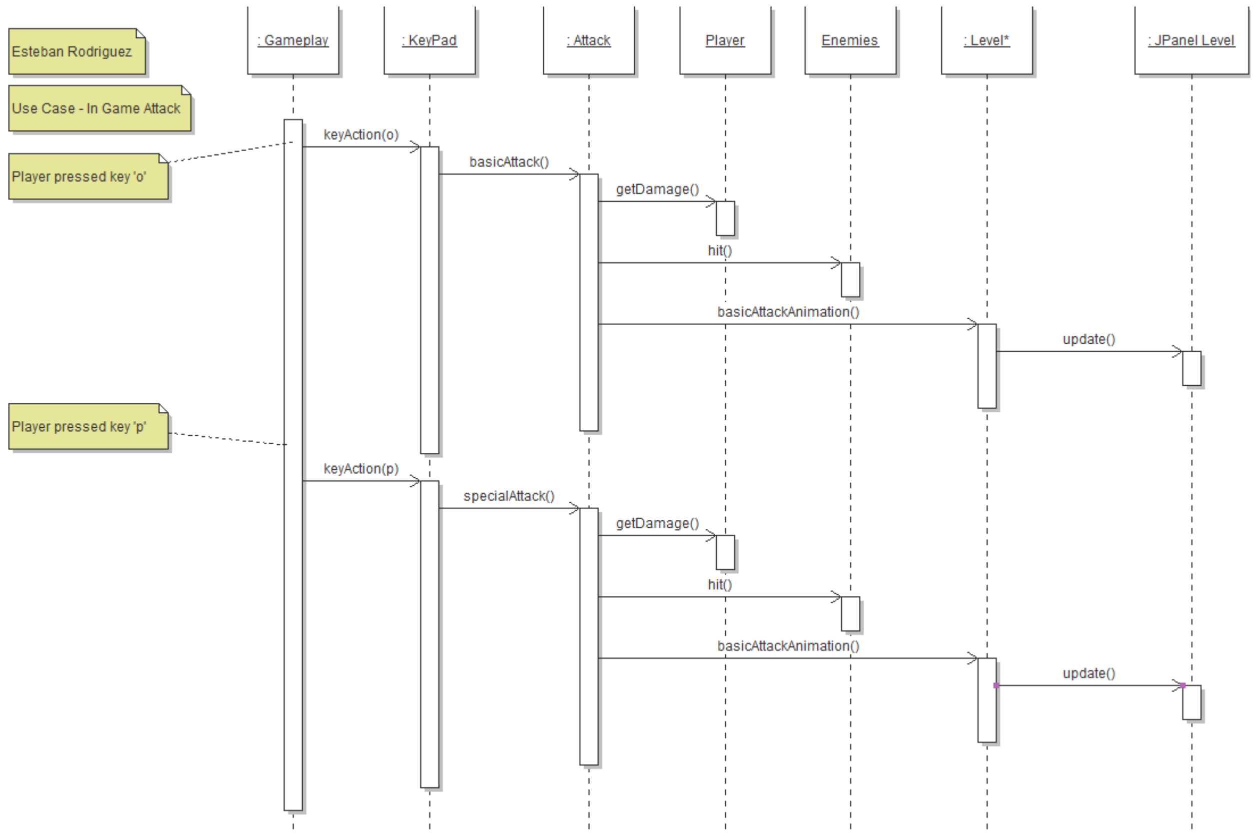
Use Case - Change Settings



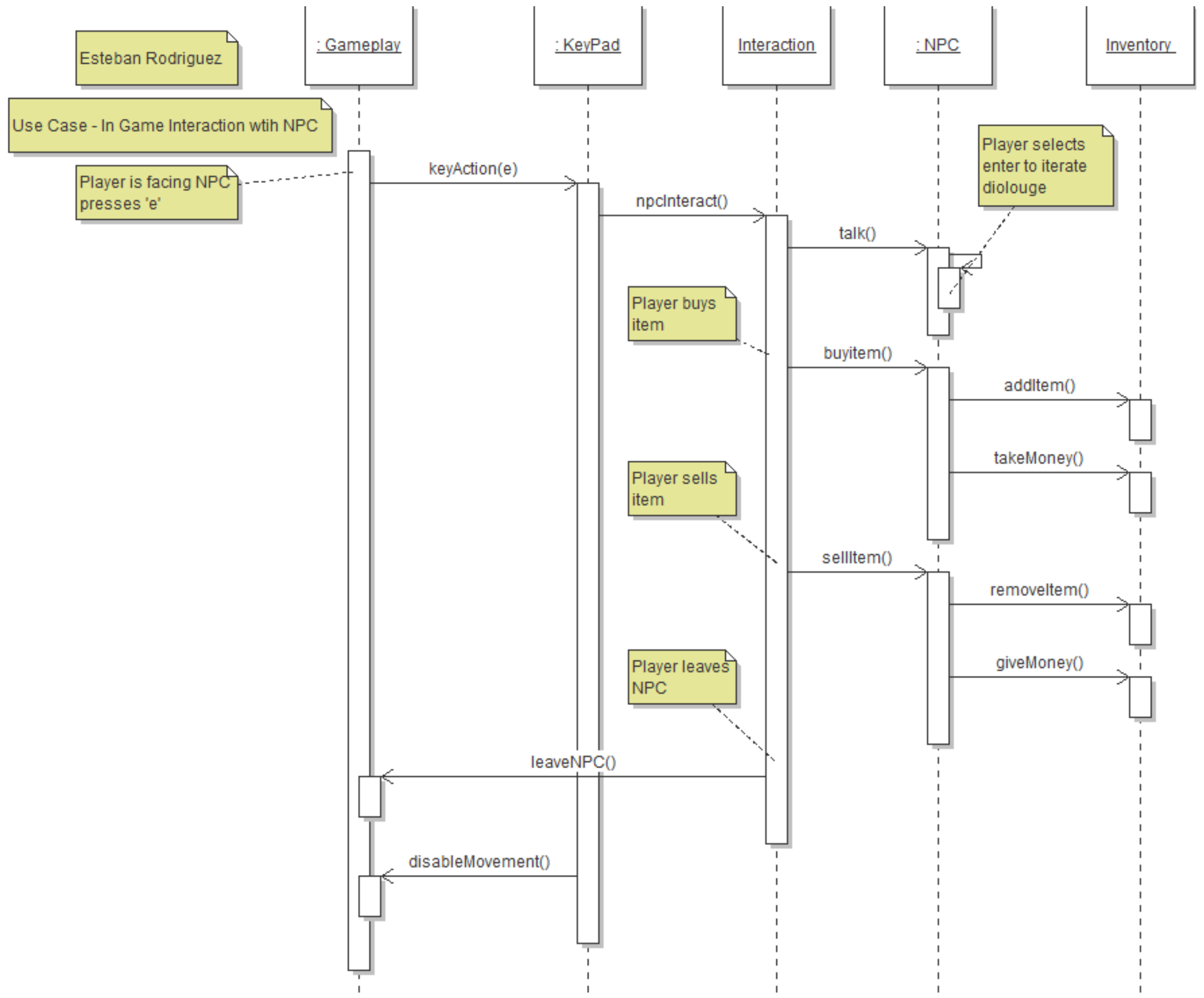
Use Case - In Game Movement



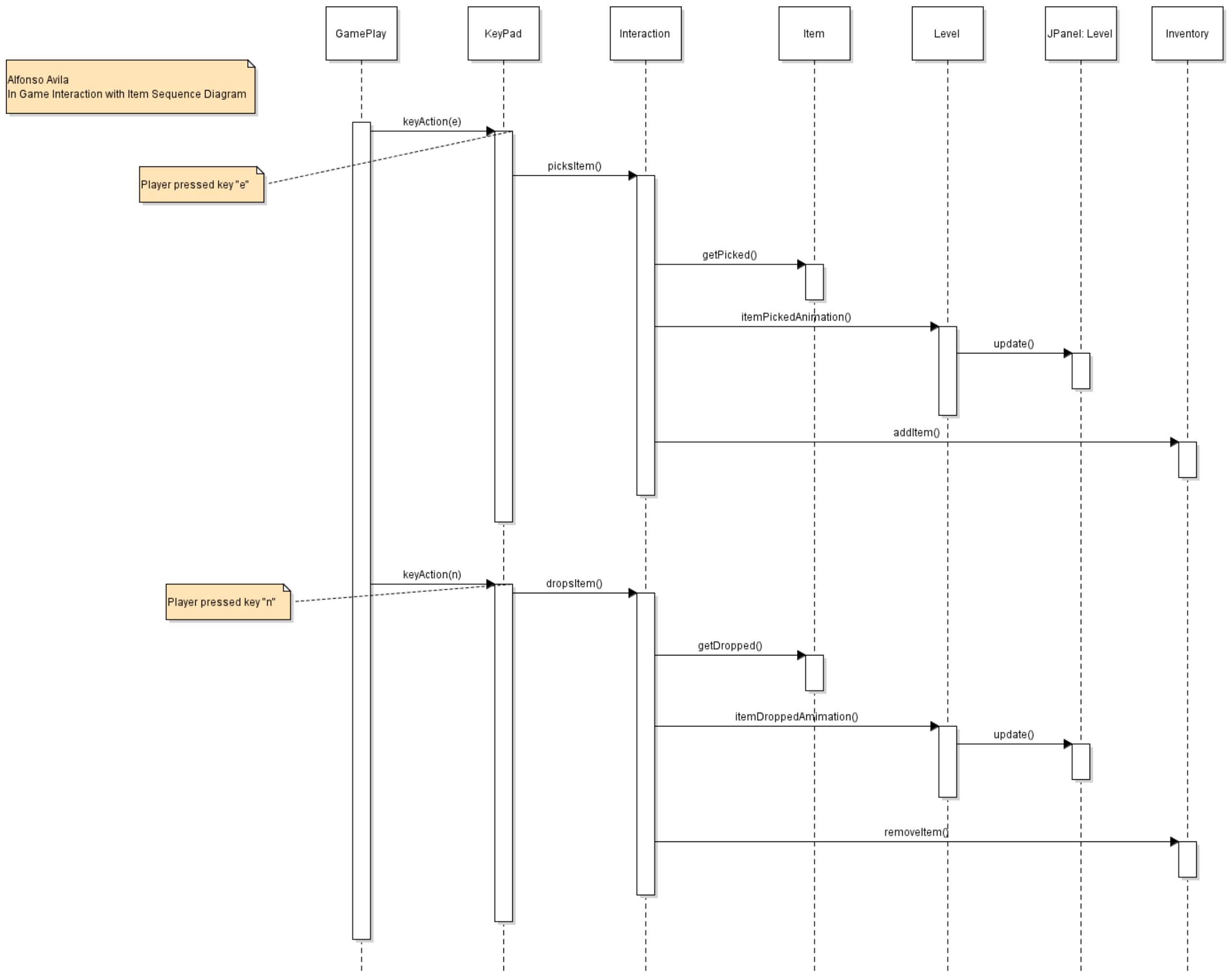
Use Case - In Game Attack



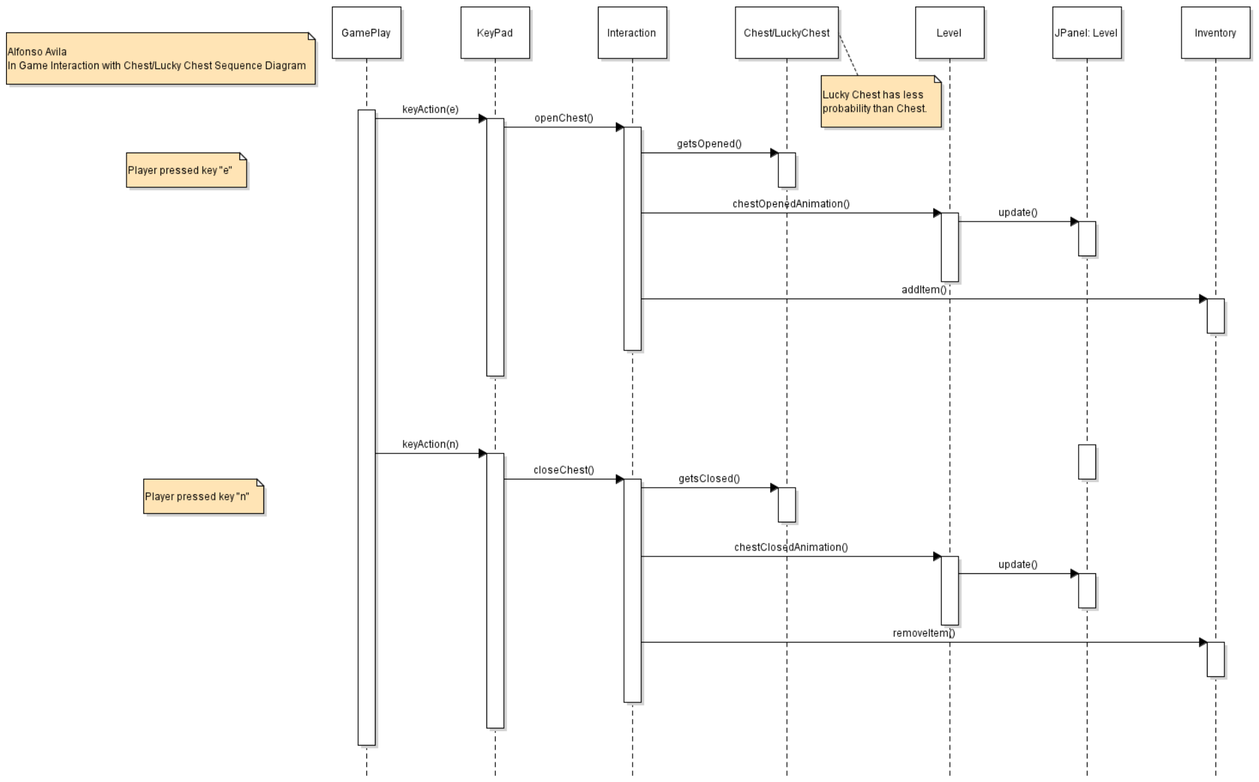
Use Case - In Game Interaction with NPC



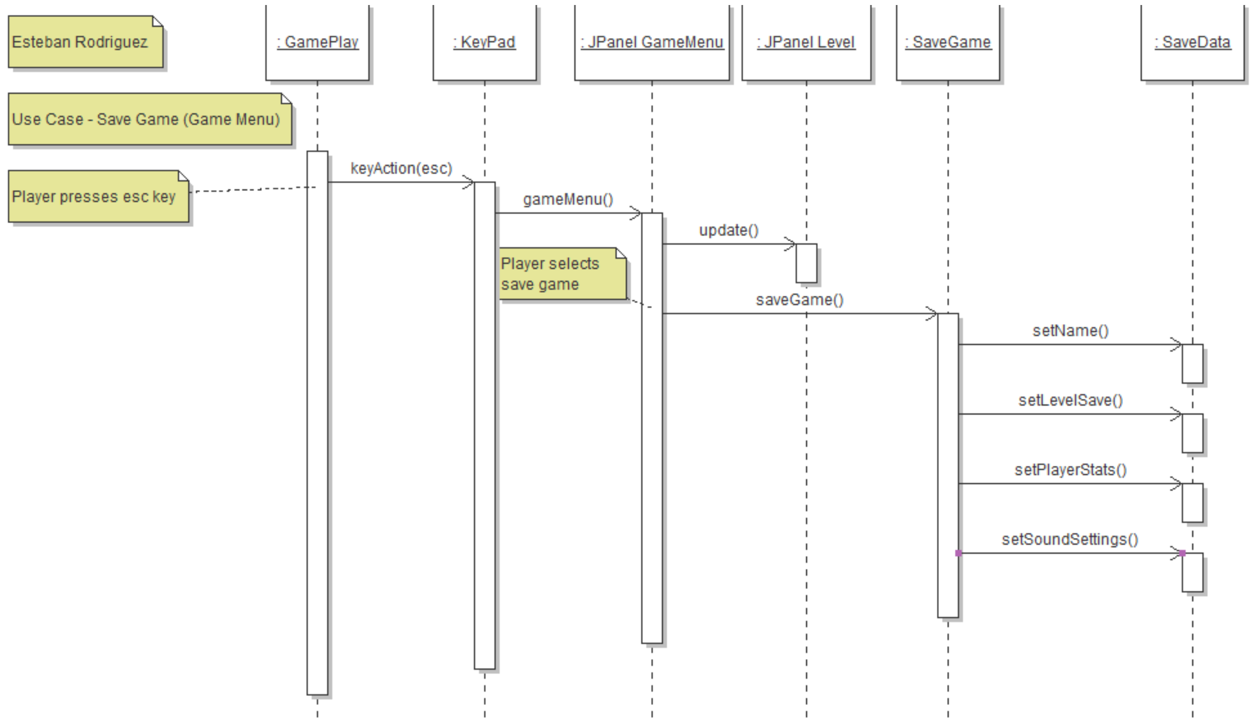
Use Case - In Game Interaction with Item



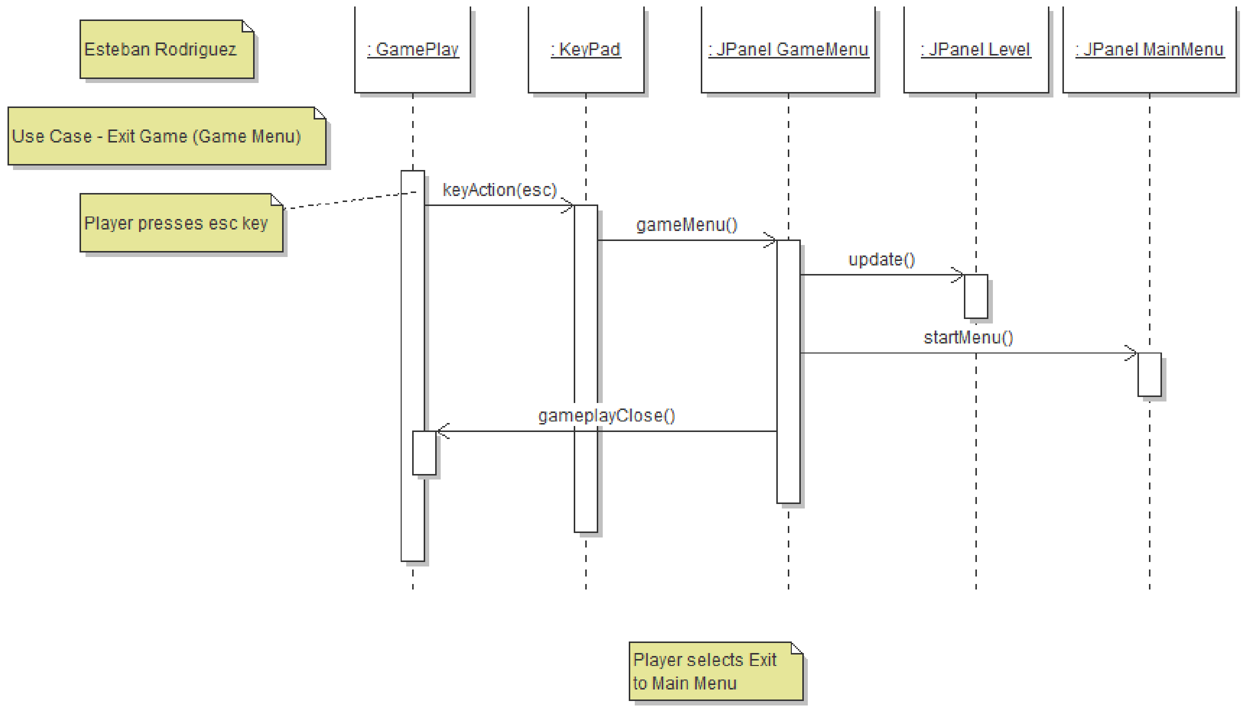
Use Case - In Game Interaction with Chest



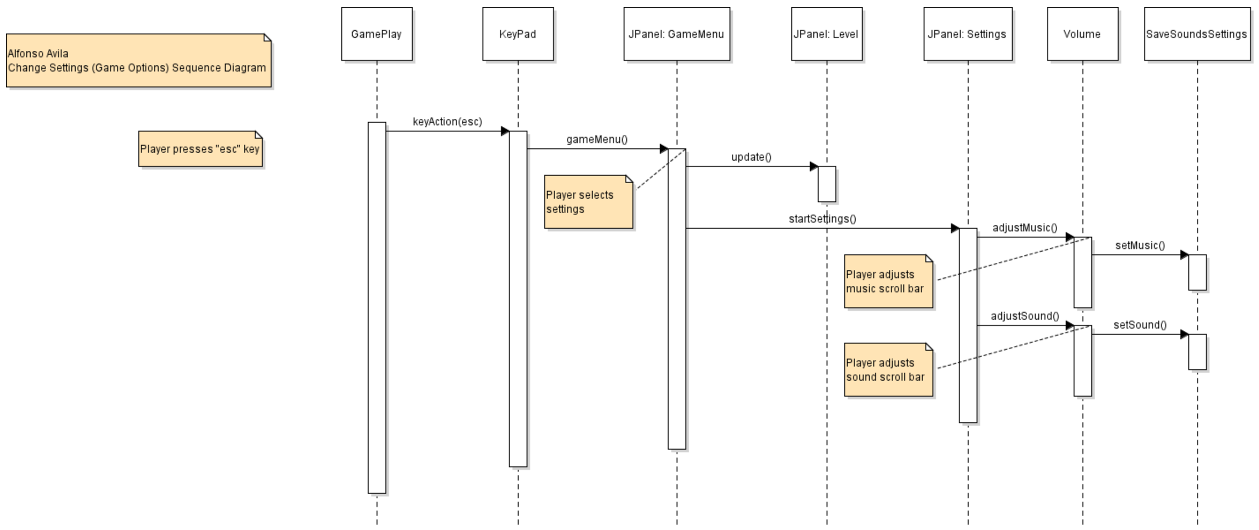
Use Case - Save Game (Game Menu)



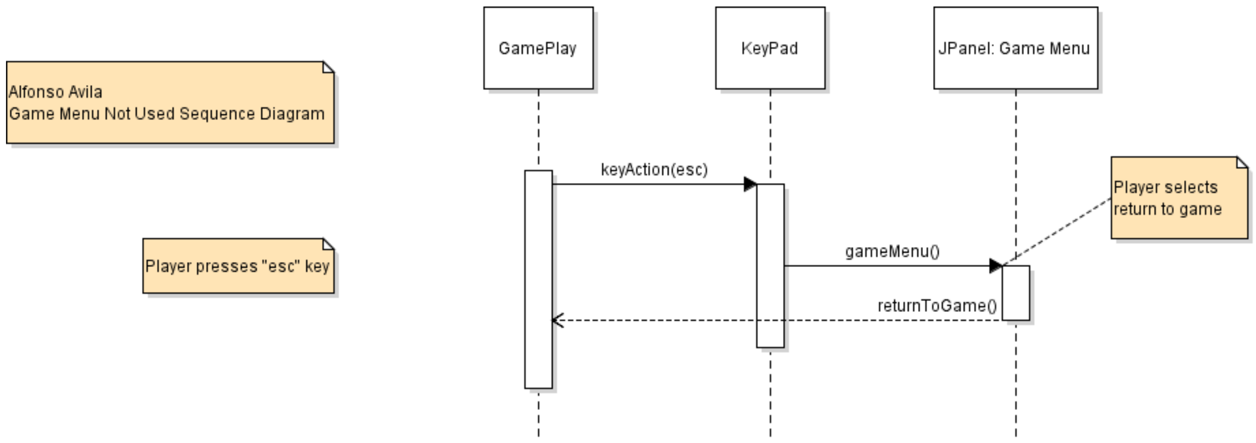
Use Case - Exit Game (Game Menu)



Use Case - Change Settings (Game Options)

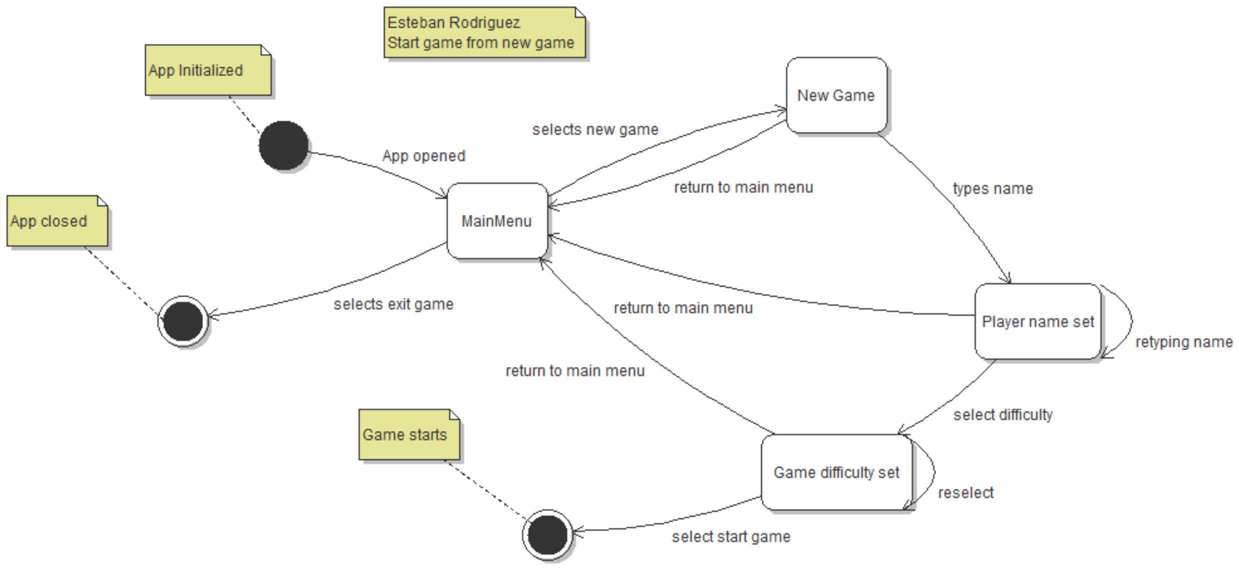


Use Case - Game Menu Not Used

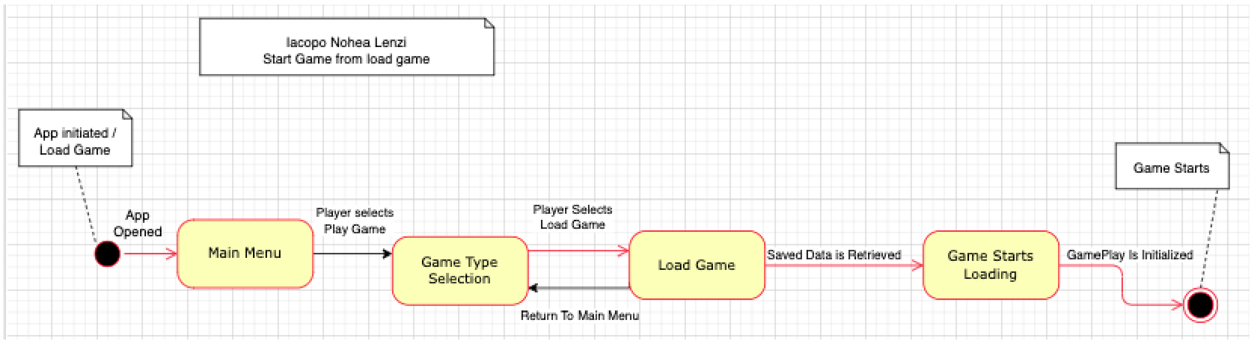


State Diagrams:

Starting Game From New Game -

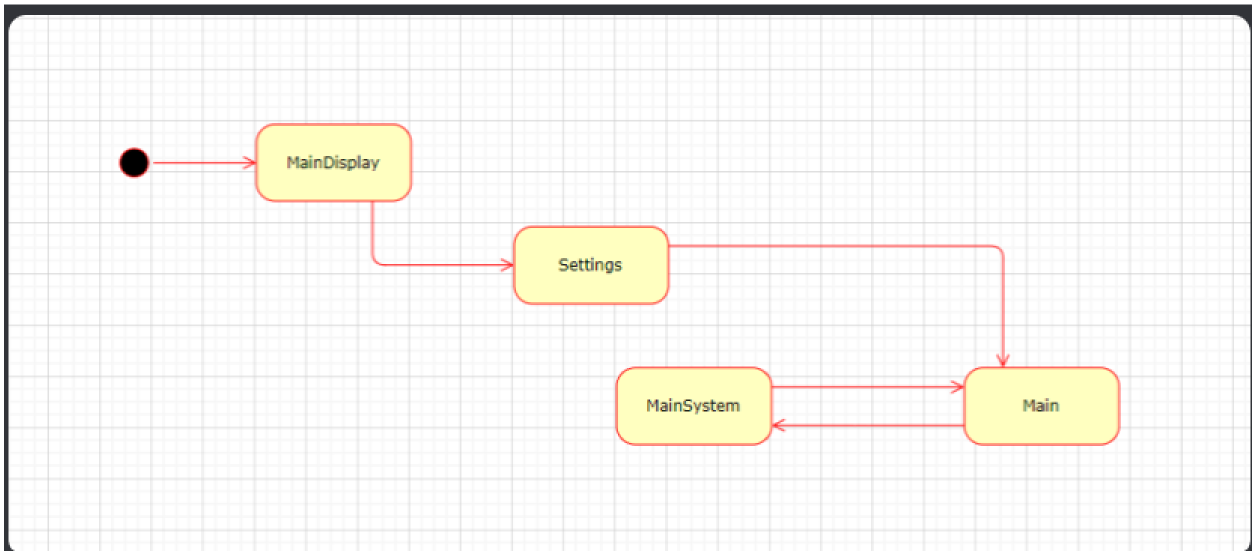


Starting Game From Load Game -

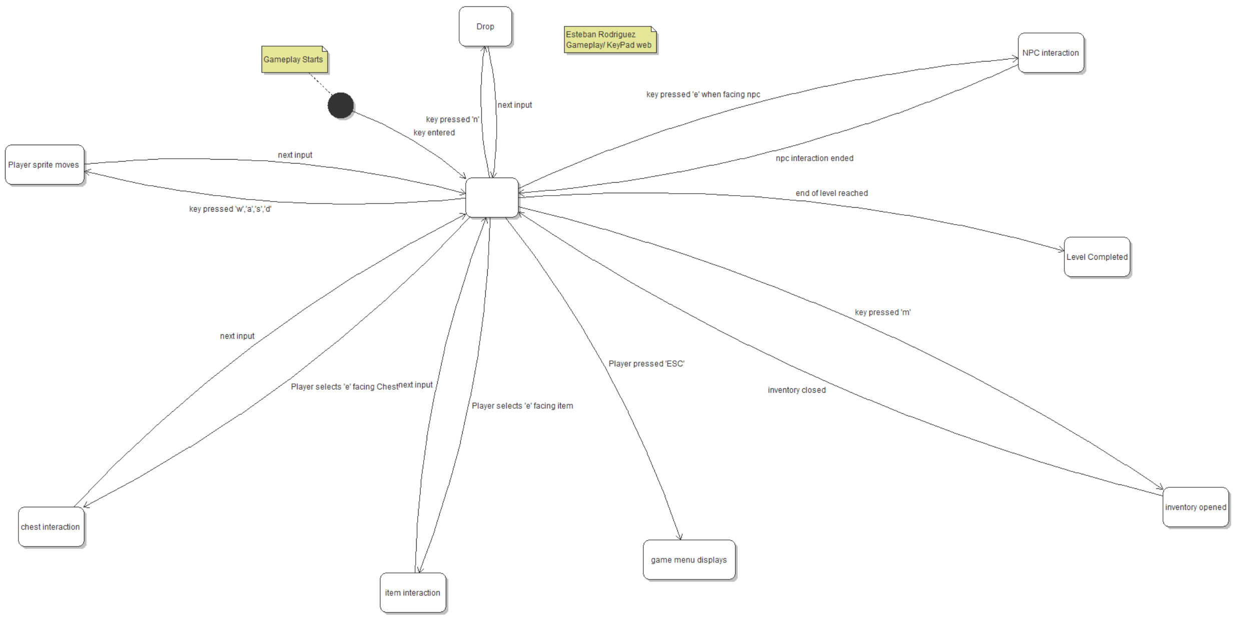


Changing Settings -

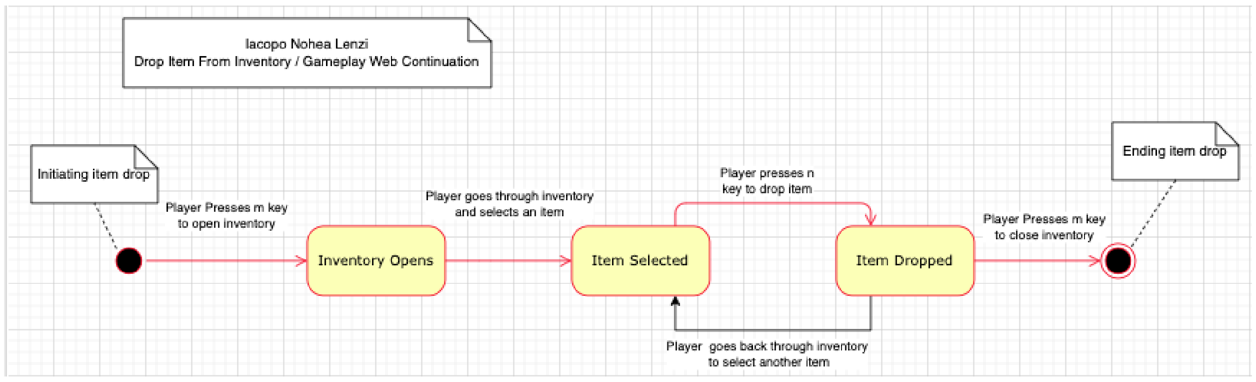
Vladimir -



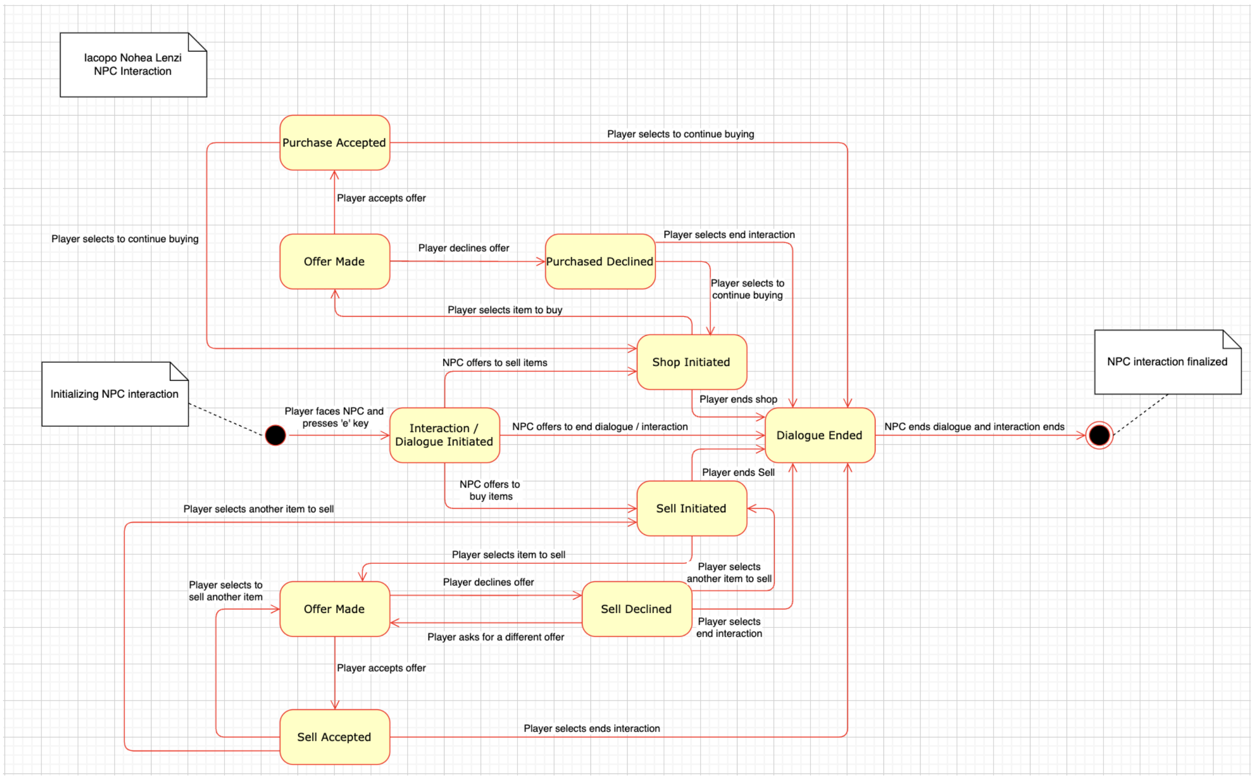
Gameplay/Keypad Web -



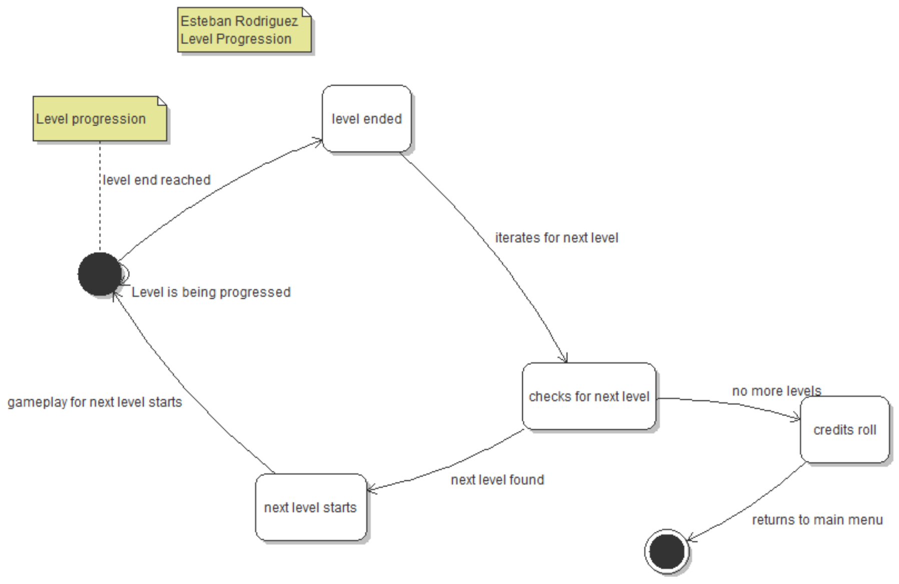
Drop Item From Inventory - Gameplay/Keypad Web Continuation -


Iacopo Nohea Lenzi
Drop Item From Inventory / Gameplay Web Continuation
Initiating item drop
Player Presses m key to open inventory
Inventory Opens
Player goes through inventory and selects an item
Item Selected
Player presses n key to drop item
Item Dropped
Player goes back through inventory to select another item
Player Presses m key to close inventory
Ending item drop

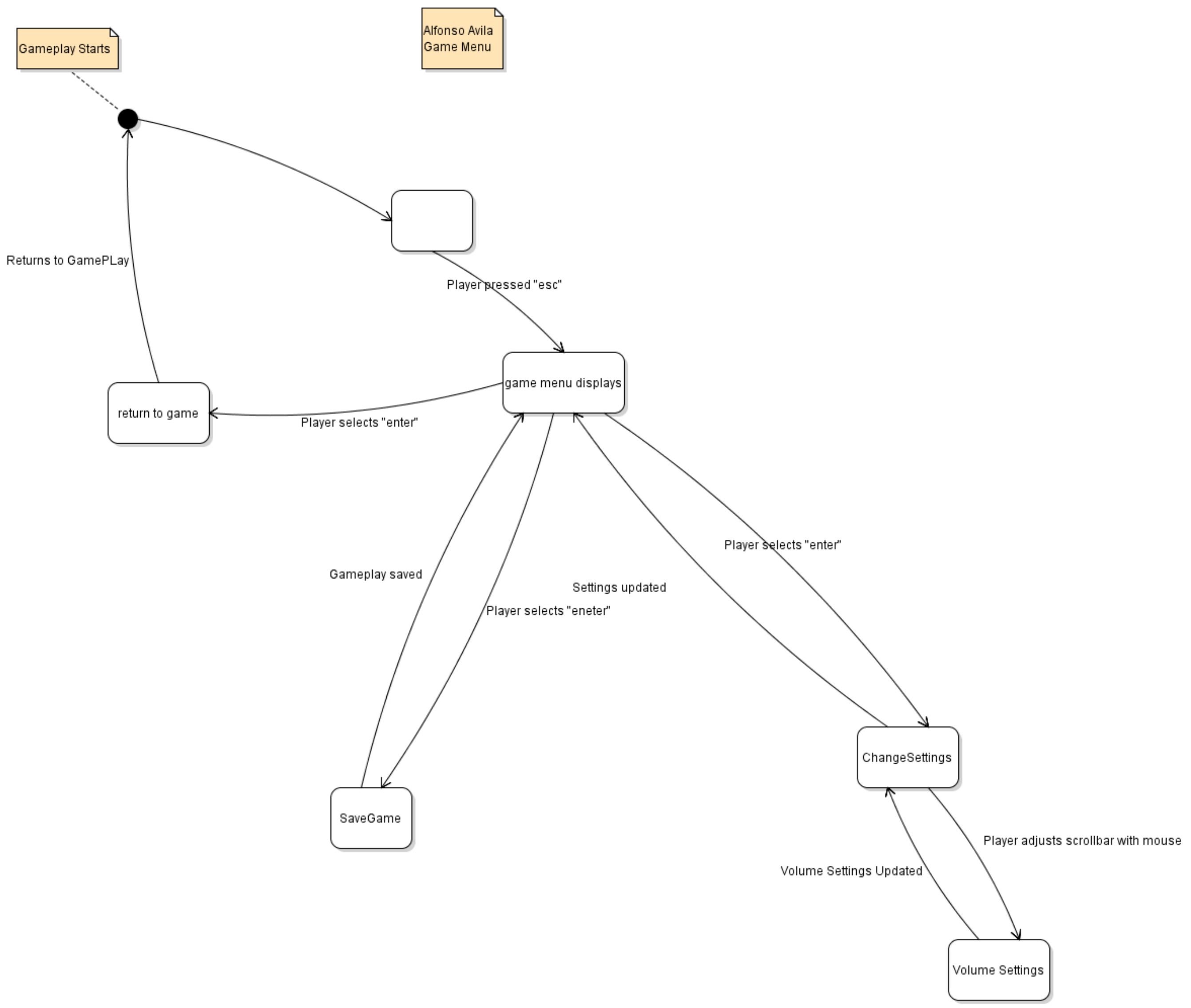
NPC Interaction - Gameplay/Keypad Web Continuation -


Iacopo Nohea Lenzi
NPC Interaction
Purchase Accepted
Player selects to continue buying
Player accepts offer
Player selects to continue buying
Offer Made
Player declines offer
Purchased Declined
Player selects end interaction
Player selects to continue buying
Player selects item to buy
Shop Initiated
NPC offers to sell items
NPC interaction finalized
Initializing NPC interaction
Player ends shop
Player faces NPC and presses 'e' key
Interaction / Dialogue Initiated
NPC offers to end dialogue / interaction
Dialogue Ended
NPC ends dialogue and interaction ends
Player ends Sell
NPC offers to buy items
Sell Initiated
Player selects another item to sell
Player selects item to sell
Player selects another item to sell
Player selects to sell another item
Offer Made
Player declines offer
Sell Declined
Player asks for a different offer
Player selects end interaction
Player accepts offer
Sell Accepted
Player selects ends interaction

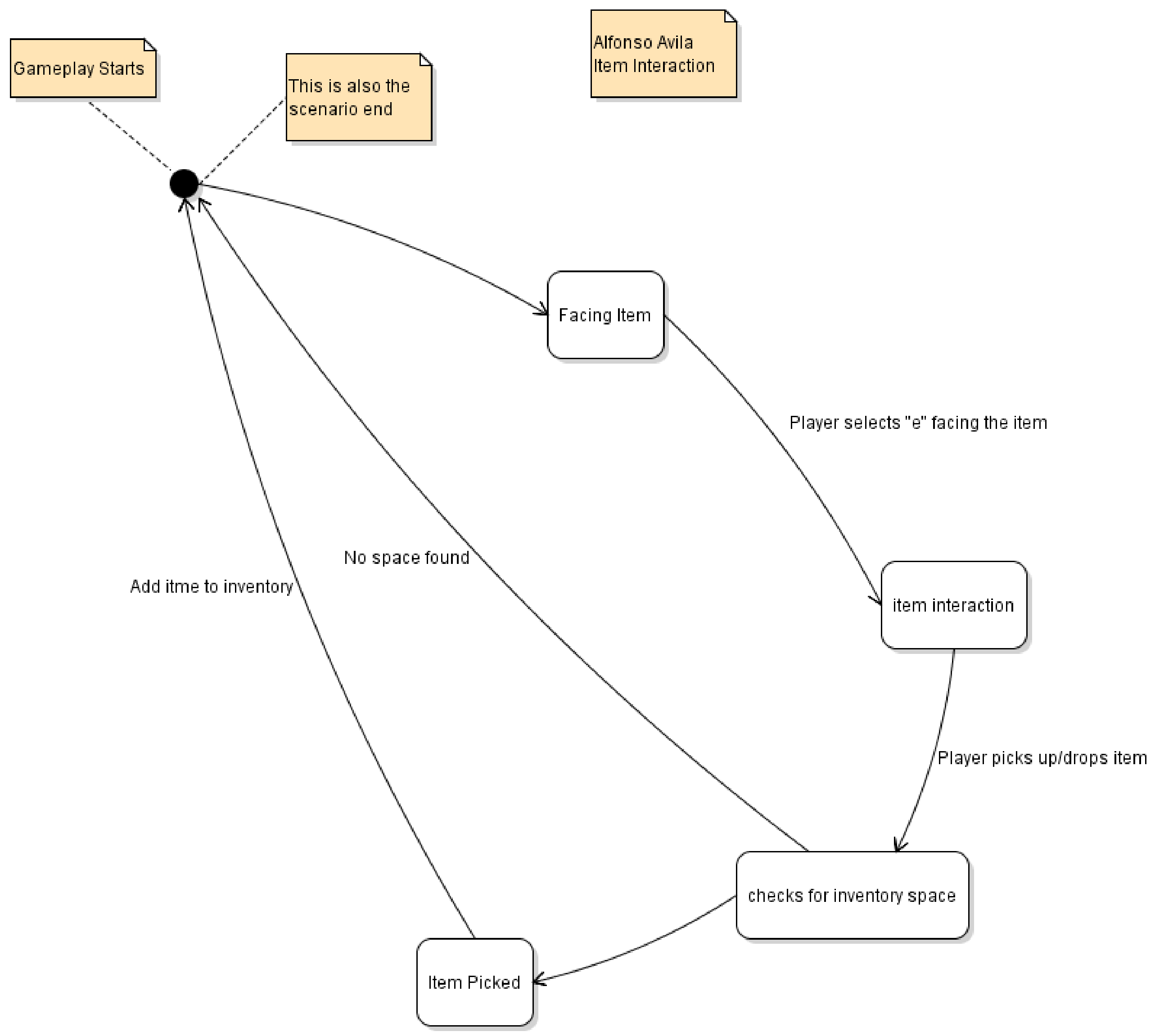
Level Progression -



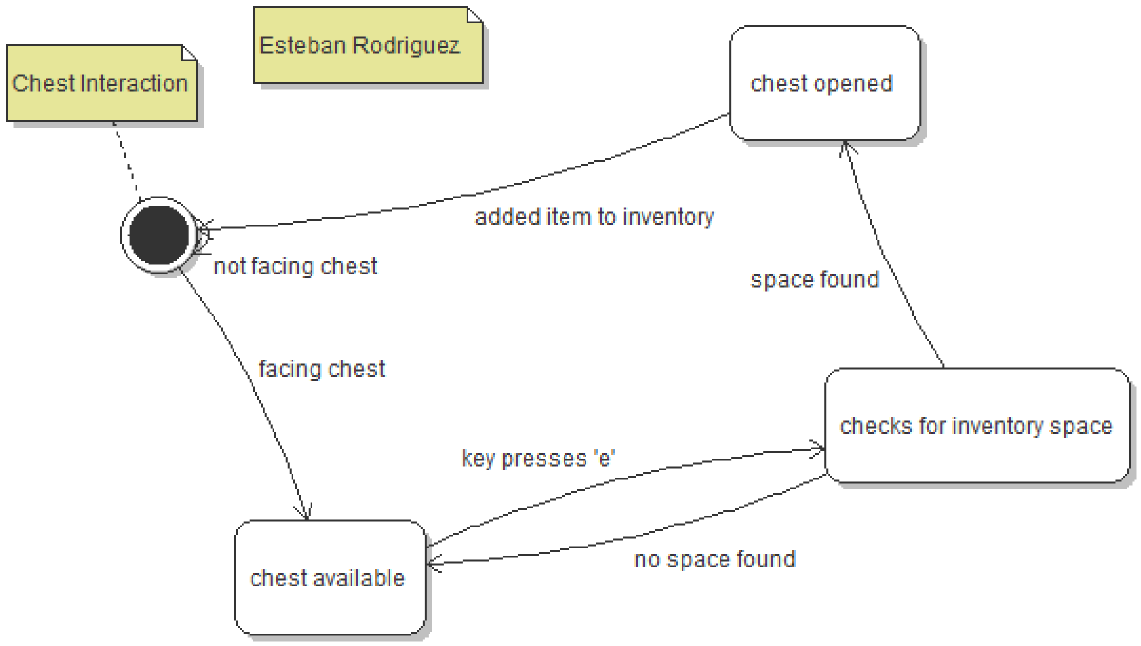
Game Menu -


Gameplay Starts
Alfonso Avila
Game Menu
Player pressed "esc"
Returns to GamePLay
game menu displays
return to game
Player selects "enter"
Player selects "enter"
Gameplay saved
Settings updated
Player selects "eneter"
ChangeSettings
SaveGame
Player adjusts scrollbar with mouse
Volume Settings Updated
Volume Settings

Item Interaction -



Chest Interaction -



Glossary -