CS345 Milestone 2

Deadwood Use Cases

Riley Hendrickson

Use Cases (step by step responsibilities given in parentheses in MSS):

Case 1: Player wants to move into a new area and take a role

Actor: Player

Precondition: It's Player's turn and they aren't currently working on a role.

Postcondition: Player is in a new area and is working on a role.

Trigger: Player has either chosen to not upgrade their rank or has already done so and opts to move areas.

MSS:

1. System prompts the user with their options (view)
2. Player selects the move option (controller)
3. System prompts player to select the area they would like to move to (view)
4. Player selects desired area (controller)
5. System validates the area is a neighboring area (SystemValidator)
6. System moves the player to the new area (Player)
7. System checks if the scene card in that area is flipped over and flips the card if not (GameData)
8. System prompts player to select the role they would like to work on (View)
9. Player selects a role (Controller)
10. System validates the player has the rank required for that role (SystemValidator)
11. System assigns that role to the player (GameData)

Termination Outcome: Player has moved into a new area and taken on a role.

Extensions:

5a. Selected area is not a neighboring area

.1: System prints error message and takes area input again

10a. Selected role requires a higher rank than that of Player's

.1: System prints error message and takes role input again (view and controller)

Case 2: Player wants to act on a role they're currently working on

Actor: Player

Precondition: It's Player's turn and they are working on a role already

Postcondition: Player has either attempted to perform their role and failed, or they successfully performed their role

Trigger: Player opted to act on their role

MSS:

1. System prompts player if they would like to rehearse or act (View)
2. Player selects act (Controller)
3. System simulates a dice roll for Player (DieRoller)
4. System validates the result of the roll is greater than or equal to the budget of the scene (SystemValidator)
5. System removes a shot counter from the scene (GameData)
6. System checks that Player's role is on the card (Player)
7. System gives Player two credits (Player)

Extensions:

4a. Player's dice roll result is less than the budget of the scene, Player's role is on the card

.1: Player gets nothing, turn ends

4b. Player's dice roll result is less than the budget of the scene, Player's role is not on the card

.1: System gives Player a dollar (Player)

.2: Turn ends

Termination Outcome:

(on success) Player has successfully acted on their role and has been properly awarded

(on failure) Player has attempted and failed to act on their role and has either been paid, or not

Case 3: Player wants to rehearse on the role they're working on

Actor: Player

Precondition:

Player wants to rehearse and doesn't have enough practice chips to guarantee success on an acting attempt

Postcondition: Player has acquired a new practice chip

Trigger: Player opts to rehearse rather than act

MSS:

1. Player opts to rehearse their role (Controller)
2. System validates the current count of practice chips +1 does not guarantee success on Player's next acting attempt (SystemValidator)
3. System gives Player a practice chip (Player)

Extensions:

2a. Player's updated count of practice chips plus their rank is greater than or equal to the rank of the player's role, i.e. the player has a guaranteed chance of successfully acting on their role

.1: System does not allow player to rehearse and prints an error message

.2: System switches over to act action

Termination Outcome:

Player has finished their turn and has an updated amount of practice chips

Case 4: A scene has wrapped

Actor: System, Active Player (player whose turn it is)

Precondition: A scene no longer has any shot counters

Postcondition: Bonus money has been distributed to the appropriate players

Trigger: the final shot counter from the scene is removed

MSS:

1. System checks if at least one player is on the scene card (GameData)
2. System simulates a number of die roll equal to the budget of the movie for Active Player (BonusMoneyHandler)
3. System distributes the respective die results to each role from the largest result to the smallest (BonusMoneyHandler)
4. System gives each player on a role on the card a number of dollars equal to the sum of their respective die roll results (BonusMoneyHandler)

5. System gives players on a role off of the card a number of dollars equal to the rank of their role (BonusMoneyHandler)
6. System removes the scene card (GameData)

Extensions/Alternate Flow:

1a. No players are acting on roles on the scene card

.1: System removes the scene card and pays no bonus money (GameData)

Termination Outcome:

Scene card is removed, bonus money is distributed if there is at least one player on the card

Case 5: A day has ended

Actor: System

Precondition: only one scene remains

Postcondition: players have been returned to the trailers, ten new scene cards have been distributed to the board, all shot counters reset if there are remaining days to be played, otherwise the system proceeds to scoring

Trigger: the second to last scene card just wrapped

MSS:

1. System verifies the current day is not the final day (GameState)
2. System returns players to the trailers removes final scene cards, and distributes ten new scene cards face down and resets shot counters (Player, GameData)
3. System increments day counter (GameState)

Extensions:

1a. The day that just ended is the final day

.1: System proceeds to scoring (ScoreHandler)

Termination Outcome: players are at the trailers, board is updated (if days remain)

Case 6: System needs to initialize the board and start game

Actor: System

Precondition: Program has just began running

Postcondition: Board information has been stored into proper objects

Trigger: User runs the program

MSS:

1. System reads card's xml file and stores card information (GameDataParser, CardParser)
2. System reads and stores set information (GameDataParser, SetParser)

Extensions:

1a. Exception raised during card reading method calls

.1: System prints error message about XML file, exits with error value 1 (GameDataParser)

2a. Exception raised during set reading method calls

.1: System prints error message about XML file, exits with error value 1 (GameDataParser)

Termination Outcome: board information has been stored and initialized

Case 7: System needs to simulate a die roll

Actor: System

Precondition: n/a

Postcondition: result of the die roll has been sent to the appropriate class(es)

Trigger: Current active player is attempting to act/System needs to determine order of turns/Bonus money for a scene wrapping needs to be determined

MSS:

1. System runs the dice roll simulator (DieRoller)
2. System prints the value of the result (View)
3. System sends the value of the result to the caller (Controller, DieRoller)

Extensions:

N/A

Termination Outcome: System has simulated the die roll and the result has been stored properly

Case 8: Player wants to upgrade at the casting office

Actor: Player

Precondition: Player has the necessary funds to make at least one upgrade, is at the casting office, and it is their turn

Postcondition: Player has an updated rank

Trigger: Player has selected to upgrade their rank at the start/end of their turn

MSS:

1. Player selects upgrade rank (Controller)
2. System prompts Player to select the upgrade they desire (View)
3. Player selects desired upgrade (Controller)
4. System validates Player has the required funds for that rank (SystemValidator)
5. System updated Player's rank (Player)

Extensions:

4a. Player does not have the required funds for selected rank

.1: System prints error message (View)

.2: System takes rank input again (Controller)

Termination Outcome:

Player has upgraded their rank and funds have been updated

Case 9: System needs to calculate scoring and determine winner

Actor: System

Precondition: All days have been completed

Postcondition: All players have received their score and a winner has been determined and displayed

Trigger: The final day just completed

MSS:

1. System iteratively for each player calculates the credit point total (ScoreHandler)
2. System then calculates the dollar point total (ScoreHandler)
3. System lastly calculates the rank point total (ScoreHandler)

4. System sums the above three totals (ScoreHandler)
5. System assigns the total for each player's respective score (Player)
6. System determines the highest total and assigns that player as the winner (GameData)
7. System prints the name of the winner (View)
8. Game ends (Controller)

Extensions:

6a. More than one player have the highest total score i.e. there is a tie

.1: System prints there was a tie and prints the names of the players in the tie as the winners (Controller, View)

Termination Outcome: Winner displayed, every player knows their respective score

Case 10: System needs to display the board information to users

Actor: System

Precondition: System has initialized and read/stored the board information/ board state has changed

Postcondition: Board has been displayed to the users

Trigger: board state just changed, or the game just started and board information just got read and initialized

MSS:

1. System reads area information iteratively (GameData)
2. System stores area information in the board (Controller)
3. System calls appropriate UI display methods (Controller)

Extensions:

N/A

Termination Outcome: Board information has been displayed to the users