COMP361D1

Requirements Elicitation - Use Cases

Group

- Hexanome-09
- Java/2D

Team Members

- Hanaozhe Hu
- Aaron Lohner
- Jules Niyonkuru
- Faisal Rabbani
- Annie Ren
- Insha Sohail
- Jonathan Zhang

Use Case Model

DO NOT SUBMIT -->

1. Play Colt Express (mention the lobby) - Insha
    1.1. Create new game - Faisal
    1.2. Join game - Faisal
        1.2.1. Join private game - Faisal
        1.2.2. Join public game - Faisal
    1.3. Load previous game - Faisal
    1.4. Round (this is a step, not a use case) - Aaron
    1.5. Take turn (schemin phase) - Jon
        1.5.1. Play action card (climb, rob, punch, shoot, move, etc.) - Jon
    1.6. Take turn stealin' phase - Annie
        1.6.1. Single-player action
            1.6.1.1. Move left/right- Annie
            1.6.1.2. Move up/down - Annie
            1.6.1.3. Rob - Annie
            1.6.1.4. Ride - Annie
        1.6.2. Multi player action - Insha

1.6.2.1. Punch - Insha
1.6.2.2. Shoot - Insha
1.6.2.3. Move marshall - Insha

1.6.3. End of round event/actions (this is a step) - Aaron

1.7. (As a step in Play Colt Express: repeat above 3 use cases until rounds are completed) - Aaron

- *Determine winner (step) - Aaron*
- Send Message in Chat(?) - Jules
- Save Game & Exit - Jules

Interacting correctly with the Lobby Service - Hu

1. Play Colt Express use case: register game Colt Express in Game Services of LS.

10:13

2. CreateNewGame use case: if we want to create a new game in Colt Express, we need to create a new session that is unlaunched,
and if we want to play with other players, we need to add players to one unlanched session. We can also remove players from that unlanched
session.

10:13

3. CreateNewGame use case: inside Extension part, Player informs System that he wishes to cancel game creation, then sessions are removed
accordingly.If the session was already launched a quit request is sent to the associated game-service.

10:14

4. Inside CreateNewGame, JoinExistingGame and LoadExitingGame use cases : before the game started, change the state of session from
unlanched to lanched.

10:14

5. SaveGame use case:
When we save game, we need to register a savegame at the LS.Also when we remove a savegame,we removes a previously registered savegame.

10:14

6. Play Colt Express use case: unregister game Colt Express in Game Services of LS and delete all sessions from game Colt Express.

7.Save Game & Exit use case: when a game ends, remove that session inside LS

TODO:

1. *Make sure every use case is referenced--directly or indirectly--from PlayColtExpress - Jules+Aaron - DONE*

2. **Finalize diagram - Insha**
3. *Add in Lobby Service steps where indicated - Jules - DONE*
4. *Add Jon's use cases (and make noted changes) - Jon - DONE*
5. **Add actions to stealin phase list and replace BasicMove use case - Annie**
6. **Add extensions where indicated - Insha**
7. **SUBMIT - Insha**

<-- DO NOT SUBMIT

SUBMIT THE USE CASES BELOW

Use Cases

PlayColtExpress
Use Case: PlayColtExpress
Scope: ColtExpress
Level: User Goal
Intention in Context: The intention of the *Player* is to play a game of Colt Express against other players.
Multiplicity: Multiple *Players* can play Colt Express concurrently. A given player is not allowed to play multiple games simultaneously.
Primary Actor: *Player*
Secondary Actors: *Player* (who play the roles of opponents), *Lobby Service*
Main Success Scenario:

1. *Player* logs on to *System*.
2. *System* requests possible actions from *Lobby Service.*
3. *Lobby Service* returns actions to *System.*
4. *System* presents to *Player* the list of available options, which include load an existing game, join an existing game, or create a new game.
   Step 5 is executed once enough players have joined the game.
5. Players choose which car to start in.
   Step 6 is repeated until all the rounds in the game have been played.
6. Players play round.
7. *System* informs all *Players* about who won the game.
   Step 8 takes place concurrently throughout steps 5-7
8. Players communicate using chat.

Extensions:
1a. *Player* is new and creates credentials to register with *System* before logging in. Use case continues at step 1.
2a. *Player* has invalid credentials to log in to *System*. Use case ends in failure.

5a. Player was not able to create, or join or load an existing game. Use case continues at step 4.
7a. *System* informs *Players* that are on the Train that they cannot win the game if the last round featured an Escape event. Use case continues at step 7.
7b. *System* informs *Players* that multiple bandits won the Gunslinger prize. The prize is divided evenly among the winners. Use case continues at step 7.
7c. *System* informs *Players* that multiple bandits are the richest. The player with the fewest Bullet cards wins.

7c.1. *System* informs *Players* that multiple players have the fewest Bullet cards.

These bandits end in a tie. Use case continues at step 7.

CreateNewGame
Use case: CreateNewGame
Scope: ColtExpress
Level: Subfunction
Multiplicity: Each *Player* can create a one game per session
Primary actor: *Player*
Secondary actor(s): *Lobby Service*
Main Success Scenario:

1. *Player* indicates to *System* that they would like to start a new game.
2. *System* directs *Lobby Service* to initiate a new session.
3. *Lobby Service* acknowledges request from *System* and initiates a new session.
   Step 4 is executed once enough players have indicated they would like to start a new game.
4. *System* instructs *Lobby Service* to launch the session
   Lobby Service launches session
5. *Lobby Service* sends a ready-to-play notification to *System.*
6. *System* receives ready-to-play notification from *Lobby Service*

Extensions:

3a. *Lobby Service* does not acknowledge request. Use case continues at step 2.

JoinGame
Use case: JoinGame
Scope: ColtExpress
Level: Subfunction
Multiplicity: Many *Players* can join an existing game.

Primary actor: *Player*
Secondary actor(s): *Lobby Service*
Main Success Scenario:

1. *Player* indicates to the *System* that they would like to join a game
2. *System* directs *Lobby Service* to load games that are accepting new players.
3. *Lobby Service* acknowledges the request and returns to *System* a list of all available games that are accepting new players.
4. *System* returns to *Player* a list of all available games for the *Player* to join
5. *Player* informs *System* of game they choose to join
6. *System* instructs the *Lobby Service* to add the *Player* to the game session
7. *Lobby Service* acknowledges the request and adds the *Player* to the existing session
   Player waits until session is launched by the Lobby Service
8. *Lobby Service* sends a ready-to-play notification to *System.*
9. *System* receives ready-to-play notification from *Lobby Service*

Extensions:

3a. *Lobby Service* does not acknowledge the request. Use case continues at step 2.
7a. *Lobby Service* does not acknowledge the request. Use case continues at step 6.

LoadPreviousGame
Use case: LoadPreviousGame
Scope: ColtExpress
Level: Subfunction
Multiplicity: Many *Players* can join a previously saved game.
Primary actor: *Player*
Secondary actor(s): *Lobby Service*
Main Success Scenario:

1. *Player* indicates to *System* that he would like to load a previously saved game.
2. *System* directs *Lobby Service* to load a previously saved game associated to the *Player* ‘s account.
3. *Lobby Service* acknowledges request and returns to *System* saved games associated to *Player* account.
4. *System* returns to *Player* saved games associated to his account.
5. *Player* informs *System* of the game to be loaded.
6. *System* informs *Lobby Service* of the game to be loaded.
7. *Lobby Service* acknowledges request and starts forked session
   Step 8 is executed once enough players have indicated they would like to continue the previously saved game.
8. *System* directs *Lobby Service* to launch the session.
9. *Lobby Service* launches the session and sends a ready-to-play notification to *System*
10. *System* receives ready-to-play notification from *Lobby Service*

Extensions:

3a. *Lobby Service* does not acknowledge the request. Use case continues at step 2.
7a. *Lobby Service* does not acknowledge the request. Use case continues to step 6.
10a. *System* does not receive ready-to-play notification. Use case continues at step 8

ChooseCar
Use Case: ChooseCar
Scope: ColtExpress
Level: Subfunction
Intention in Context: The intention of the *Player* is to select which train car he/she would like to start in
Multiplicity: All *Players* can choose their cars at the same time
Primary Actor: *Player*
Main Success Scenario:

1. *System* informs all Players of the current car that can now be selected
2. *Player* informs *System* if they want to start in the current car
3. *System* informs all Players of the Players that are starting in the current car
   Step 1, 2 are repeated until all train cars have been visited
4. *System* updates the game state and informs all Players of the new game state

---

PlayRound
Use Case: PlayRound
Scope: ColtExpress
Level: Subfunction
Intention in Context: The intention of the *Player* is to play their turn.
Primary Actor: *Player*
Secondary Actors: *Player (who play the roles of opponents)*
Main Success Scenario:

1. *System* informs *Players* of the turn types and of the end-of-round event (or train station event for the last round in the game) for this round. The turns and events which may take place can be from the following options:

   Turns:

   - Standard turn: Action cards are played face-up.
   - Tunnel: Action cards are played face-down.
   - Speeding-Up: Take 2 turns in a row (players can choose to either play or draw cards on each turn)

- Switching: Turns take place in reverse order, beginning with the first player
- Turmoil: Action cards are all revealed at the same time. The Action cards will be played out in the Stealin' phase in the normal order

Round events

- Angry Marshal: The Marshal shoots the bandits on the roof of his car. Each bandit on the roof of the Marshal's car receives 1 Neutral Bullet. Then the Marshal moves 1 car toward the caboose. If the Marshal is already in the caboose, he does not move.
- Swivel arm: Each bandit on a roof moves to the roof of the caboose
- Braking: Each bandit on a roof moves to the roof of the adjacent car closest to the locomotive. Bandits on the roof of the locomotive do not move.
- Take it all: A third strongbox is placed in the Marshal's space
- Passenger's rebellion: Each bandit inside a car receives 1 Neutral Bullet
- Panting horses: In a 5-6 player-game, the 2 Horses closest to the Caboose are removed from the game. In a 2-4 player-game, only the Horse closest to the Caboose is removed.
- A shot of Whiskey for the Marshall: If there is a Whiskey Flask on the Marshall's position, it is removed from the game. If it is a classic Whiskey Flask, the Marshall moves to the adjacent Car, toward the back of the train. If it is an Old Whiskey Flask, the Marshall moves on as many Cars as necessary to reach the Caboose. Each time the Marshall meets a Bandit during this movement, the latter must flee on the roof and receives a Neutral bullet.
- Higher speed: All Horses, the Stagecoach the Shotgun and all Bandits who are on the roof of the train move one Car towards the back of the train.
- The Shotgun's rage: All bandits on the Stagecoach or on its roof or in the Car in front of the Stagecoach or on its roof receive a Neutral Bullet.

Train station events:

- Marshal's revenge: Each bandit on the roof of the Marshal's car places their least valuable purse on the roof of their car. If a bandit has no purse, they lose nothing.
- Pickpocketing: Each bandit with no other bandits in their space may take 1 purse in their space and place it face-down on their character sheet.
- Hostage conductor: Each bandit inside or on the roof of the locomotive takes 1 $250 purse from outside the game.
- Sharing the loot: If several Bandits are on the same Car and at the same level, if any of them have any Strongboxes, the value of the Strongboxes is shared between all of the present Bandit (if needed, the result is rounded down).
- Escape: All the Bandits remaining in the Train (either inside a Car, the Locomotive or on the roof) have to leave the train. The game is over for those players but they are in the race for the victory.

- Mortal bullet: Each player loses $150 for each Bullet received during this Round.

2. *Players* take turns during schemin' phase.
3. *Players* take turns during stealin' phase.
4. *System* informs *Players* of new game state.

Extensions:
1a. *System* informs *Player* with the Lady's Poodle hostage that they receive a Neutral Bullet at the beginning of the round; Use case continues to step 1.
(1-4)a. *Player* chooses to save and exit game. Use case ends in failure.

ScheminPhase
Use Case: ScheminPhase
Scope: ColtExpress
Level: Subfunction
Intention in Context: The Player wants to plan out actions based on the round card and other Player's actions
Multiplicity: Several Players can play action cards or draw cards. One at a time, in set order.
Primary Actor: Player
Secondary Actors: Player Specific Display, General Display
Main Success Scenario:

Step 1 occurs concurrently for all Players

1. Systems takes 6 cards from each Player's deck, places them in their hand, and sends the graphical result to Player Specific Display

Repeat steps 2-4 for each player

2. System sends a message to General Display indicating the type of turn and who's turn it is to play
3. Player chooses to PlayActionCard (ref. PlayActionCard)
4. System sends a message to General Display indicating that the Player has concluded their turn

Step 5 occurs concurrently for all Players

5. System takes any unplayed cards in Players' hands, places them on top of their decks and sends graphical result to Player Specific Display

Extensions:
1a. Player's character is Doc and Player is not in possession of The Poker Player or The Minister hostage; System gives Player an additional card

Use case continues at step 2.

1b. Player kept cards from last round

1b.1. System instead gives Player (6-# of cards kept) cards and sends graphical result to Player Specific Display

Use case continues at step 2.

1c. Player has The Minister hostage; System gives Player one fewer cards

Use case continues at step 2.

(2-4)a. It is a switching turn:

(2-4)a.1. *System reverses the Player turn list*

Use case continues at step 5.

(2-4)b. It is a turmoil turn:

(2-4)b.1. *Steps 2-4 happen concurrently for all Players and cards go on the pile all at once after all Players have used PlayActionCard (cards go on the pile in normal order regardless of who chooses their action faster)*

Use case continues at step 5.

3a. Player informs System he/she wants to draw 3 cards instead

3a.1. System takes 3 cards from Player's deck, places them in their hand, and sends graphical result to General Display

Use case continues at step 4.

3b. It is a normal turn and Player informs System he/she would like to use an old whiskey flask

3b.1. Player invokes PlayActionCard twice (ref. PlayActionCard)

3b.2. System flips or discards whiskey based on previous state and displays the result on General Display

Use case continues at step 4.

3c. It is a normal turn and Player informs System he/she would like to use a normal whiskey flask

3c.1. System gives Player three cards and displays this operation on General Display

3c.2. Player invokes PlayActionCard (ref. PlayActionCard)

3c.3. System flips or discards whiskey based on previous state and displays the result on General Display

Use case continues at step 4.

3d. Player's character is Ghost and he/she is choosing to play their first action face-down and Player is not in possession of The Photographer or The Poker Player hostage

3d.1. Player invokes PlayFace-Down (ref. PlayFace-Down)

Use case continues at step 4.

3e. It is a speeding-up turn:

3e.1. Player informs the System on whether he would like to draw 3 cards or invokes PlayActionCard an additional time (ref. PlayActionCard)

Use case continues at step 4.

3f. It is a tunnel turn and Player does not have The Photographer hostage:

3f.1. Player invokes PlayFace-Down (ref. PlayFace-Down)

Use case continues at step 4.

5a. For each End of Round Bonus icon shown on the round card Players may select up to that many cards in their hands to keep for next round

Use case ends in success

PlayActionCard
Use Case: PlayActionCard
Scope: ColtExpress
Level: Subfunction
Intention in Context: Player wants to select and move a card from his/her hand onto the action pile
Multiplicity: Only one Player can be interacting with one instance of this function
Primary Actor: Player
Secondary Actors: General Display, Player Specific Display
Main Success Scenario:

1. Player informs System which action card he/she would like to play
2. *System moves selected card from Players hand to the action pile*
3. System displays the new addition to the action pile on General Display

Extensions:
2a. System does not move the selected card from Player's hand to action pile because Player selected a bullet card

2a.1. System sends error message to Player Specific Display prompting them to choose a valid card

Use case continues at step 1.

2b. System does not move the selected card from Player's hand to action pile because Player selected a ride card while in possession of The Zealot hostage

2b.1. System sends error message to Player Specific Display prompting them to choose a valid card

Use case continues at step 1.

2c. System does not move the selected card from Player's hand to action pile because Player selected a punch card while in possession of The Teacher hostage

2c.1. System sends error message to Player Specific Display prompting them to choose a valid card

Use case continues at step 1.

PlayFace-Down
Use Case: PlayFace-Down
Scope: ColtExpress
Level: Subfunction
Intention in Context: Player wants to play an action card and be able to see what he played while other Players can only see the back of the card that was played
Multiplicity: Only one Player can be interacting with one instance of this function
Primary Actor: Player
Secondary Actors: General Display, Player Specific Display
Main Success Scenario:

1. Player informs System which card in his/her hand he would like to play face-down
2. *System moves selected card from Player's hand area to the action pile*

Steps 3 and 4 happen concurrently

3. System displays a card back being added to the action pile on the General Display
4. System shows the image side of the action card on the action pile on the Player Specific Display of the Player that just played that card

Extensions:

2a. System does not move selected card from Player's hand area to the action pile because Player selected a bullet card

2a.1. System sends message to Player Specific Display prompting Player to choose a valid action card

Use case continues at step 1.

2b. System does not move selected card from Player's hand to action pile because Player selected a ride card while in possession of The Zealot hostage

2b.1. System sends message to Player Specific Display prompting Player to choose a valid action card

Use case continues at step 1.

2c. System does not move selected card from Player's hand to action pile because Player selected a punch card while in possession of The Teacher hostage

2c.1. System sends message to Player Specific Display prompting Player to choose a valid action card

Use case continues at step 1.

StealinPhase

Use Case: StealinPhase

Scope: ColtExpress

Level: Subfunction

Intention in Context: The intention of the *Player* is to have their action cards resolved

Multiplicity: Only one *Player* can have their one action card resolved at one time

Primary Actor: *Player*

Main Success Scenario:

1. *System* informs *Player* that it is his/her turn
2. *System* informs *Player* what action card is being resolved, which can be one of the following:
    - a Move, an action that involves one interaction with the system
    - a ChangeFloor, an action that involves no interaction with the system
    - a Rob, an action that involves two interactions with the *system*
    - a Ride, an action that involves one interaction with the *system*
    - a Punch, an action that involves multiple actors and interactions with the system

- a Shoot, an action that involves multiple actors and interactions with the system
- a MoveMarshall, an action that involves multiple actors and interactions with the system

3. *System* informs all *Players* about new game state

Extensions:

2a. *System* informs *Player* that they are no longer on the Train or Stagecoach. Use case continues at step 3.

3a. *Player* informs *System* that they play a Ride card in this round which includes the Escape train station event. Use case continues at step 3.

Move

Use Case: Move

Scope: ColtExpress

Level: Subfunction

Intention in Context: The intention of the *Player* is to make a move, i.e. an action that involves one input sent from the *Player* to the System

Multiplicity: Only one *Player* can make a Move at one time

Primary Actor: *Player*

Main Success Scenario:

1. *System* informs *Player* about the adjacent train cars that he/she can choose to move to and asks *Player* to choose the car to move to
2. *Player* informs System about the position he/she wants to move to (data: which car or roof to move to)
3. System updates the game state
4. System informs all Players of the new state

Extension:

1a. System informs *Player* that they are on the roof of a car and can thus move up to three cars in either direction; use case continues at Step 2.

2a. *Player* informs System of an invalid position that they can not move to

2a. 1. System informs *Player* that his/her selection is invalid and asks *Player to choose again*; use case continues at Step 2.

2b. System informs *Player* that Marshal is in the car that *he/she* chose to move to

2b. 1. System informs *Player* that they can not stay in the same car where Marshal is located and that they have received a Neutural Bullet card and have been sent to the roof of the car; use case continues at Step 3.

~~**ChangeFloor**~~

~~**Use Case:** ChangeFloor~~

~~**Scope:** ColtExpress~~
~~**Level:** Subfunction~~
~~**Intention in Context:** The intention of the *Player* is to change their floor, i.e. an action that involves no input sent from the *Player* to the System~~
~~**Multiplicity:** Only one *Player* can make one ChageFloor move at one time~~
~~**Primary Actor:** *Player*~~
~~**Main Success Scenario:**~~
~~1. *System* informs *Player* whether if they are currently in a car or on a roof~~
~~2. *Player* informs System about the position he/she wants to move to (data: which car or roof to move to)~~
~~5. System updates the game state and informs all Players of the new state~~
~~**Extension:**~~
~~1a. System informs *Player* that they are on the roof of a car and can thus move up to three cars in either direction; use case continues at Step 2.~~
~~2a. *Player* informs System of an invalid position that they can not move to~~
~~2a. 1. System informs *Player* that his/her selection is invalid and asks *Player to choose again*; use case continues at Step 2.~~

Rob
Use Case: Rob
Scope: ColtExpress
Level: Subfunction
Intention in Context: The intention of the *Player* is to take a loot token of their choice from the current car
Multiplicity: Only one *Player* can take up to one loot token at one time
Primary Actor: *Player*
Main Success Scenario:
1. *System* informs *Player* of the loot token(s) that he/she can choose to rob
2. *System* asks *Player* to choose one token
3. *Player* informs *System* about the loot token they choose to take by selecting one of the tokens (data: which loot token to choose)
4. *System* informs all *Players* about the new state.

Extensions:
1a. There are no loot tokens in the train car that *Player* is in.
1a. 1. System informs *Player* that he/she has nothing to rob; Use case continues at step 4.

Ride

Use Case: Ride
Scope: ColtExpress
Level: Subfunction
Intention in Context: The intention of the *Player* is to ride a horse of his/her choice
Multiplicity: Only one *Player* can ride one horse at one time
Primary Actor: *Player*
Main Success Scenario:

1. *System* informs *Player* of the horse(s) that he/she can ride
2. *Player* informs *System* of the horse he/she choose to ride via selecting the desired horse (data: which horse to ride)
3. *System* informs *Player* of the possible train cars that he/she can move to
4. *Player* informs *System* of the destination train car (data: which car to move to)
5. *System* informs all *Players* of the new game state

Extensions:

1a. There are no horses adjacent to the *Player*'s car

1a. 1. System informs *Player* that he/she has no horses to ride; use case continues at step 5.

2a. *Player* selects a horse that is not adjacent to his/her car

2a. 1. System informs *Player* that his/her selection is invalid; use case continues at step 1.

3a. *Player* is next to the stagecoach and chooses to ride to stagecoach:

3a. 1. System informs *Player* of the available hostages the *Player* can choose from

3a. 2. *Player* selects a hostage *and* informs System of the hostage they chose; use case continues at step 5.

3a. 1a. Player has no hostages to choose from

3a. 1a. 1. System informs *Player* that they have no hostages to choose from; use case continues at step 5.

3a. 2a. *Player* already has a hostage:

3a. 2a. 1. System informs *Player* that they can not take the hostage; use case continues at step 5.

Punch
Use Case: Punch
Scope: ColtExpress
Level: Subfunction
Intention in Context: The intention of the *Player* is to punch a player of their choice from the current car
Multiplicity: Only one *Player* can punch another player at one time
Primary Actor: *Player*
Secondary Actor: *Opponent Player*
Main Success Scenario:

1. *System* informs *Player* of the *Opponent Player* that he/she can choose to punch
2. *System* asks *Player* to choose one *Opponent Player*
3. *System* informs *Player* the cars to which the *Opponent Player* can be punched
4. *System* asks *Player* to choose the car
5. *Player* informs the *System* which car they would like to place the *Opponent Player* after punching them
6. *System* informs *Player* of the loot tokens that he/she can choose to have dropped in the player space
7. *System* asks *Player* to choose one loot token
8. *Player* informs *System* about the loot token they choose to drop
9. *System* informs all *Players* about the new state

Extensions:

1a. *System* inform *Player* that there are no *Opponent Players* in the train car that the *Player* is in.

1a. 1. System informs *Player* that he/she has no *Opponent Player* available to punch; Use case continues at step 7.

1b. *System* informs *Player* that they cannot punch because they obtained The Teacher hostage this round. Use case continues at step 9

2a. *Player* is in the same car Belle and at least another *Opponent Player*

2a.1 *System* informs *Player* that Belle cannot be punched. Use case continues at step 2.

2a.2 *System* informs *Player* that Belle can be punched because Belle is in possession of the Poker Player hostage. Use case continues at step 2.

5a. *System* informs *Player* that the Marshal is in the car chosen and that the *Opponent Player* has been sent to the roof of that car. Use case continues at 6.

6a. *System* inform *Player* that there are no loot tokens with the *Opponent Player* that is being punched.

6a. 1. System informs *Player* that there are no available tokens to drop; use case continues at step 7.

Shoot

Use Case: Shoot
Scope: ColtExpress
Level: Subfunction
Intention in Context: The intention of the *Player* is to shoot a player of their choice
Multiplicity: Only one *Player* can shoot another player at one time
Primary Actor: *Player*
Secondary Actor: *Opponent Player (represents the one who is shot)*
Main Success Scenario:

1. *System* informs *Player* of the players that he/she can choose to shoot

2. *System* asks *Player* to choose one player
3. *Player* informs *System* about the player they choose to shoot
   System deals one *Player* Bullet card to the *Opponent Player* that is shot
4. *System* informs all *Players* about the new state.

Extensions:

1a. *System* informs *Player* that there are no *Opponent Players* for the *Player* to shoot; (none in the adjacent cars if Player is on the ground or none in the line of sight if *Player* is on a roof).

1a. 1. System informs *Player* that he/she has no player available to shoot; *Player* retains the Bullet card; the use case continues at step 5.

1b. System lets the *Player* know that since the *Player's character is Django* that their shot will move the *Opponent Player* to the end of the train.

TUCO - SHOOTS THROUGH ROOFS
DJANGO - HIS SHOT SHOOTS YOU TO THE END OF THE TRAIN
BELLE - CANNOT BE SHOT IF YOU HAVE ANOTHER OPTION (BUT **CAN** BE SHOT IF SHE HAS POKER PLAYER)
PLAYER HAS NO BULLETS REMAINING

MoveMarshall
Use Case: MoveMarshall
Scope: ColtExpress
Level: Subfunction
Intention in Context: The intention of the *Player* is to move the Marshall
Multiplicity: Only one *Player* can move the Marshall at a time
Primary Actor: *Player*
Main Success Scenario:

1. *System* informs *Player* of the cars that he/she can move the Marshall to
2. *Player* informs *System* about the car to which they are moving the Marshall
   System moves the Marshall into the car
3. System informs all Players about the new state.

Extensions:

4a. There are *Players* in the car the Marshall is in
   4a. 1. System deals one Neutral Bullet card to the *Players* in the car
      4a.1.i. *Players* are moved to the top of the car; use case continues at step 3.

SendMessage
Use Case: SendMessage
Scope: ColtExpress

Level: Subfunction
Intention in Context: The intention of the *Player* is to send a message to other players
Multiplicity: Multiple players can send messages.
Primary Actor: *Player*
Main Success Scenario:

1. *Player* informs *System* that he wishes to send a message in chat to other players.
2. *System* presents a chat box to *Player*.
   Player writes message in chat box
3. *Player* informs *System* that the message is ready to be sent to other players.
4. *System* updates the chat box state and informs all Players about the new state.

SaveExit
Use Case: SaveExit
Scope: ColtExpress
Level: Subfunction
Intention in Context: The intention of the *Player* is to save and exit the game and be able to resume the game later
Multiplicity: Multiple players can save the same game.
Primary Actor: *Player*
Secondary Actor: Lobby Service
Main Success Scenario:

1. *Player* informs *System* that he wishes to save and exit the game.
2. *System* informs *Players* that *Player* has decided to save and exit the game and that the game will not continue until the same number of initial players are available.
   System stops the game.
3. *System* informs *Lobby Service* to save a forked copy of the existing session on *Player's* account.
4. *Lobby Service* acknowledges request from *System* and saves the forked session.
5. *Lobby Service* informs *System* that the game was successfully saved.
6. *System* informs *Player* that his game was successfully saved.
7. *System* instructs *Lobby Service* to put this game on the list of possible games to join
8. *Lobby Service* informs *System* that this game was put on the list of possible games to join.

Extensions:
4.a. *Lobby Service* does not acknowledge the request. Use case continues to step 3.