**Play Flash Point**

**Use Case:** Play Flash Point

**Scope:** Flash Point

**Level:** User Goal

**Intention in Context:** The intention of the *Player* is to play a game of flash point with his/her team mates.

**Multiplicity:** Multiple players can play flash point concurrently. A given player is not allowed to play multiple games simultaneously.

**Primary Actor:** *Player*

**Secondary Actors:** *Player (who play the roles of team mates).*

**Main Success Scenario:**

1. Player enters game menu.
2. Player chooses to either Join/Load/Create a game.

*Step 3 is executed after room owner starts the game.*

1. Player chooses initial position for firefighters.
2. Take turns.

*Step 4 is repeated until players win/lose.*

1. System informs player if they won/lost the game.

**Extension:**

2a. If the game is a saved game, step 4 is executed.

2b. If the player is not able to join, create, or load the game, repeat step 2.

3a. If the player is playing in experienced mode, player also choose together initial position to place fire engine and ambulance.

4a. If enough players want to continue to get a perfect win, the game continues.

**Join Existing Game**

**Use Case:** Join Existing Game

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** The intension of the Player is to join a server that someone else created and is not started yet.

**Primary Actor:** Player

**Secondary Actor:** Other Players, Server Owner

**Main Success Scenario:**

1. Player informs System that he wish to join in a specific server.

2. System presents game lobby to Player.

3. Player chooses a character informs System that he is ready to begin the game.

4. Player waits all other players to ready and the server owner start the game.

**Extensions:**

2a. Player informs System that he wishes to exit this lobby. Use case ends in failure.

4a. If server owner exits the lobby, player exits as well. Use case end in failure.

**Load Game**

**Use Case:** Load game

**Scope:** Flash Point

**Level:** subfunction

**Intention in Context:** load the game that early saved

**Multiplicity:** Multiple players can load the same game concurrently. A single player can load different game.

**Primary Actor:** Player

**Main Success Scenario:**

1. System presents the Player a list of saved games.

2. Player chooses the desired game to load.

3. System shows player the game lobby.

4. System detects if the game is ready to be started.

5. Player informs the system to start the game.

**Extensions:**

2a. The saved game file is damaged; use case ends in failure.

(3-5)a. Player changes his mind and doesn’t want to play anymore, he can exit; use case ends in failure.

**Create New Game**

**Use Case:** Create New Game

**Scope:** Flash Point

**Level:** Subfunction

**Intention in context:** The intention of the player is to start a new game from scratch.

**Primary Actor:** Player

**Main Success Scenario:**

1. Player informs the system to create a new game for x number of players.

2. System presents details of games for players to choose. (Map, difficulties, fire etc.).

3. Player informs System the detail of game settings.

4. System presents the player the created game lobby.

5. System detects if the game is ready to be started.

6. Player informs the system to start the game.

**Extension:**

2a: Time is limited for player to choose types. if player does not finish, system will automatically choose one mode.

3a: When ready bottom is not pressed within limited time, system will set it ready. Use case continues at step 2

(2-6)a : Player informs Systems that he/she wishes to cancel game creation. Use case ends in failure.

**Place Starting Positions**

**Use Case:** Place Starting Positions

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** The intention of the Players is to Place their starting firefighter and vehicle positions.

**Primary Actor:** Player

**Secondary Actors:** Other players (teammates).

**Main Success Scenario:**

1. System places initial game objects (fire, hotspot, etc).
2. System informs Player to place their starting position on any of the board spaces outside of the building.
3. Player informs System the position he wishes to place.
4. System informs players to vote for vehicle initial positions.
5. System informs Player of new game states.

**Extension:**

3a. If the position Player chooses is not valid *(Occupied, Wrong Scope, etc.)*

3a.1 System informs Player of invalid placement.

3a.2 System informs Player to place a firefighter on a required position.

**Communication**

**Use case:** Communication

**Scope:** Flash point

**Level:** subfunction

**Intention in context:** Player to communicate with his teammates

**Primary actor:** Player

**Secondary actors:** Other players (Teammates)

**Multiplicity:** several users can communicate simultaneously

**Main success scenario:**

1. Current player performs one of the following:

- A text communication,

- A voice communication.

2. System publishes the message to all players

**Text Communication**

**Use case:** Text Communication

**Scope:** Flash point

**Level:** subfunction

**Intention in context:** Player to communicate with his teammates in text message

**Primary actor:** Player

**Main success scenario:**

1. Player enters the text message

2. System receives the message

**Extensions:**

1a. Player enters empty string, use case ends.

1b. Player enters mature text, the text is replaced by “\*” (extra feature)

**Voice Communication**

**Use case:** Voice Communication

**Scope:** Flash point

**Level:** subfunction

**Intention in context:** Player to communicate with his teammates in voice.

**Primary actor:** Player

**Main success scenario:**

1. Player speaks to the microphone.

2. System receives the voice

**Extensions:**

1a. If no recording device is detected, use case ends

**Take-Turn**

**Use Case:** Take-Turn

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** The intention of the player is to play his turn.

**Primary Actor:** *Player*

**Secondary Actor:** Other player (player’s team mates).

**Main Success Scenario:**

1. System informs the *Player* that it is his turn.
2. The player takes action.

*Step 2 can be repeated as many times as the player wants.*

1. System informs player the updated game state after the player’s action.
2. Current *Player* informs the system to ends his/her turn.
3. System checks the action point. System changes the Player’s action point to 4 if it exceeded 4.
4. System informs the end of the player’s turn and *Game Manager* starts to Advance Fire and Replenish POI

**Extension:**

2a. If the turn time is up, then step 5 is executed regardless of the player’s remaining action points.

2b. If the player chooses to skip (saving his/her action point), step 5 is executed.

2c. If after taking action the game is over, the system pause/terminates the game and informs players if they won/lose.

6a. If there is no more POI to be replenished, skip this step.

6b. If there is no more POI to be replenished and no POI are on the map, the system informs players if they won/lost.

**Pass a Turn**

**Use Case:** Pass a Turn

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** Player wants to pass a turn

**Primary Actor:** Player

**Main Success Scenario:**

1. Player informs System that he/she wants to pass a turn.
2. System saves Player’s unused AP.
3. System informs Player of new game states.

**Extension:**

2a. System detects if Player’s unused AP exceeds the limit, System sets Player’s AP to the limit.

**Take Action**

**Use Case:** Take Action

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** The intention of the player is to perform action to cooperate with his/her team mates.

**Primary Actor:** *Player*

**Secondary Actor:** Other player (player’s team mates).

**Main Success Scenario:**

1. *Player* chooses one of the following actions to perform.

* The player chooses to move to an adjacent position (data: the position to move to).
* The player chops a block of wall (data: which block of wall the player wants to chop).
* The player opens/closes a door (data: which door the player wants to open/close).
* The player extinguishes smoke/fire on a position (data: which fire the player wants to extinguish).
* The player carries/drops a victim/hazmat (data: which item the player wants to carry up).
* The player uses his/her special ability (data: the target to use the special ability on if any). (This only exists in experienced mode).
* The player uses the fire engine extinguish fire.
* The player drives the ambulance/fire engine (data: player to travel with and destination location).

1. System informs players the updated *Player* status.
2. System informs players the updated game state.

**Extension:**

1a. *Player* does not have enough action point to perform the desired action.

1a.1. System informs the *Player* about insufficient action point.

1a.2. The action is omitted; the use case ends in failure.

1b. *Player* tries to perform an action on an invalid target.

1b.1. System informs *Player* about invalid target of action.

1b.2. The action is omitted; the use case ends in failure.

1c. *Player* moves to a place where POI is placed.

1c.1. System reveals the POI.

1c.2. System updates the overview list of POI.

1c.3. System removes the POI if it is a false alarm; use case continues at step 2.

1d. *Player* carries a victim to safe position (ambulance in experienced mode).

1d.1. System removes the victim from the *Player’s* inventory.

1d.2. System adds the victim to the list of rescued victims.

1d.3. System updates the overview list of POI, use case continues at step 3.

(1-3)a. If the time runs up during any steps, the taken action is performed before turn goes to next player; use case continues until all steps are done.

**Move**

**Use Case:** Move

**Scope:** FlashPoint

**Level:** Subfunction

**Intention in Context:** The intension of the Player is to move to a new position

**Primary Actor:** Player

**Secondary Actor:** *POI Manager*

**Main Success Scenario:**

1. Player informs System that he wishes to move to a new position.

2. System highlights all reachable positions/grids.

3. Player choose a position.

4. System determines the cheapest path to the destination.

5. System updates the process of walking.

6. *POI manager* determines if there is a POI need to be revealed.

**Extensions:**

3a. Player chooses an unreachable position, System warns the player. Use case ends in failure.

3b. Player informs System that he determines not to move. Use case ends in failure.

**Extinguish fire**

**User case:** Extinguish fire

**Scope:** System

**Level:** Sub-function

**Intention in Context:** Player want to extinguish fire or smoke in a specific place.

**Primary Actor:** Player

**Main Success Scenario:**

1. The player chooses to extinguish a fire.

2. change fire to smoke.

3. extinguish smoke

4. The player successfully finishes the action.

**Extension:**

(1-3)a. The player doesn’t have enough action point (2AP to extinguish fire; 1AP to change fire to smoke; 1AP to extinguish smoke). Fail use case

(1-2)b. The area where player stand doesn’t adjacent with fire. Fail use case.

3b The area where player stand doesn’t adjacent with smoke. Fail use case.

(1-3)d. If the player is carrying a victim, then all the AP must double when he or she is doing some actions.

**Carry Victim**

**Use Case:** Carry Victim

**Scope:** Flash Point

**Level:** Subfunction

**Intention in context:** The intention of the player is to carry the victim to safe place and rescue the victim successfully.

**Primary Actor:** firefighter

**Main Success Scenario:**

1. Move firefighter towards a POI.

2. Reveal a POI in the place where you are

repeat this step until you find a victim.

3. Carry the victim into an empty space or a space with smoke.

4. Victim is rescued when they are removed to a valid space.

- when in family mode, victims are rescued successfully when they are out of the house.

- when in other mode, victims are rescued successfully when they get to the ambulance.

5. System updates the information

5.1. removes the victim from player’s inventory

5.2. adds the victim to the list of rescued victims

5.3. updated the overview list of POI

6. use case continues at Take Action.

**Extension:**

1a. If the POI is none, then the use case ends in failure.

2a. Carrying a victim through fire will result in failure.

2b. Player doesn’t have enough AP to perform movements and carry a victim, system informs player, and use case ends in failure.

3a. If not in family mode, there is no ambulance when victim is out of the house, use case ends in failure.

**Carry Hazmat**

**Use Case:** Carry Hazmat

**Scope:** Flash Point

**Level:** Subfunction

**Intention in context:** The intention of the player is to carry the hazmat to safe place before it explodes.

**Primary Actor:** firefighter

**Main Success Scenario:**

1. Place the hazmat randomly in the house.

2. Move firefighter towards hazmat.

3. Carry the hazmat into an empty space or a space with smoke.

4. Hazmat is removed when it is carried to a safe space.

5. use case continues at Take Action.

**Extension:**

3a. Carrying a Hazmat through fire will result in explosion, use case ends in failure.

3b. Player doesn’t have enough AP to perform movements and carry a hazmat, system informs player, and use case ends in failure.

**Drive ambulance**

**Use Case:** Drive ambulance

**Scope:** Flash Point

**Level:** Subfunction

**Intention in context:** The intention of the player is to drive the ambulance to specific destination and help firefighter to rescue victim successfully.

Primary Actor: driver

**Main Success Scenario:**

1. choose the initial position to place the ambulance.

2. Drive the ambulance to places where you want to go to rescue victim.

radio the ambulance to move (no need to be in the vehicle) or ride along with the ambulance with 2AP.

3. System informs player if the victim is rescued successfully.

4. Use case continues at Take Action.

**Extension:**

1a: If ambulance is placed in wrong place, system will inform player to choose other places.

2a: If firefighter and ambulance are on the same spot, he can either send the ambulance to next spot or drive the ambulance. If there are two firefighters, two can drive the ambulance, no AP needed for one firefighter.

2b: When the victim is out of the house, if there is no ambulance, use case ends in failure.

**Drive Fire Engine**

Use Case: Drive Fire Engine

Scope: Flash Point

Level: Subfunction

Intention in context: The intention of the player is to drive the fire engine and extinguish fire.

Primary Actor: driver

Main Success Scenario:

1. Choose the initial position to place the fire engine.

2. Drive the fire engines to places where you want to go to extinguish fire.

3. Firefighter fire the deck gun to extinguish fire in random way in the adjacent quadrant.

4. Fire or smoke in the target space, or adjacent four spaces as long as they are not separated by walls are extinguished.

5. Use case continues at Take Action.

1a: If the fire engine is placed in wrong place, system will inform player to choose other places.

2a: Move the fire engine to one parking spot to another spot adjacent to it, otherwise more action points required.

3a: If the firefighter is not in the vehicle, use cases ends in failure.

3b: The firefighter can only fire the deck gun at the adjacent quadrant, otherwise use case ends in failure.

3c: If there is firefighter in the target quadrant, use case ends in failure.

**Use Fire Engine**

**Use Case:** Place Starting Positions

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** Player uses engine to extinguish fire.

**Primary Actor:** Player

**Main Success Scenario:**

1. Player interacts with the fire engine.

2. System detects if Player has enough AP. If not, use case ends in failure.

3. Player informs System the coordinate.

4. System detects if the coordinate is drivable/useable.

5. System informs Player of new game states.

**Extension:**

1a. Drive the Vehicle and Use Desk Gun

3a. For driving the engine, Player selects the coordinate.

For using the Desk, System randomly selects the coordinate.

4a. The eligible coordinate cannot be occupied by other objects. If is ineligible, go back to Step 3.

**Switch Role**

**Use Case:** Switch Role

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** The intention of the player is to switch to another role as firefighter to have different special abilities (this use case is only valid for advanced mode of the game).

**Primary Actor:** *Player*

**Secondary Actor:** Other player (player’s team mates).

**Main Success Scenario:**

1. Player clicks the fire engine and switch role option.
2. System presents a list of available roles to switch.
3. Player informs the system which role to switch to.
4. System changes the player’s status and special ability.
5. System changes the list of available roles for another switch.

**Extension:**

1a. The player has not enough AP to switch his/her role, the system will not show the switch role option.

(1-2)a. The player can cancel role switching any time before informing the system which role to switch to, in this case, no AP will be consumed.

4a. When *Player* changes to a role with extra AP or something like “Movement AP” the *Player’s* AP bar is also changed with the respect of the role’s descriptions.

**Interact Door**

**Use case:** Interact Door

**Scope:** Flash point

**Level:** subfunction

**Intention in context:** Player to interact with door objects.

**Primary actor:** Player

**Main success scenario:**

1. Player chooses to interact with the door object.

2. System checks Player’s AP.

3. System flips the open/closed status of the door object.

**Extensions:**

1a. If the door object is not adjacent to the player, use case ends in failure.

2a. If player’s AP is less than 1, use case ends in failure.

**Destroy Wall**

**Use case:** Destroy Wall

**Scope:** Flash point

**Level:** subfunction

**Intention in context:** Player to deal damage to a wall object.

**Primary actor:** Player

**Main success scenario:**

1. Player chooses to deal damage to a wall object.

2. System checks the remaining AP of the player.

3. System add 1 to the wall object’s damage count.

**Extensions:**

1a. If the wall object is not adjacent to the player, use case ends in failure.

2a. If player’s AP is less than 2, use case ends in failure.

3a. If the damage count of the wall is 2 after adding, System removes the wall object, and reduce 1 from the building’s health.

**Use Special Abilities**

**Use Case:** Use Special Abilities

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** The intention of the player is perform the special ability of the role the player is currently playing. (This use case only exists in experienced mode.)

**Primary Actor:** *Player*

**Secondary Actor:** Other player (player’s team mates).

**Main Success Scenario:**

1. The *Player* informs the system to perform special ability of the Player’s current role.
2. System updates the game status after performing the special ability.
3. System updates the player’s status after performing the special ability.

**Extension:**

1a. If the role’s special ability is a “passive” ability (like Generalist’s extra AP, Rescue Specialist’s extra Movement AP, less Chopping AP consumption and extra AP consumption to extinguish fire, etc.) that are cannot be used but act as the Role’s properties.

1a.1. When the role’s special ability involves special AP bar (i.e. Movement AP bar), AP consumption for related actions are first consumed from the special AP bar.

1a.2. When the role’s special ability will cost different AP then regular role to perform an action, the AP consumption for the specific action of the role is changed to the desired value.

1b. If the role’s ability involves rerolling a die (i.e. Driver/Operator).

1b.1. System informs the *Player* to have a choice of either reroll the dice or not.

1b.2. Player informs system whether to reroll or not reroll the dice.

1c. If the role’s ability involves other players (i.e. Fire Captain).

1c.1. System informs the other player involved in this special ability action about what the *Player’s* action on them is.

1c.2. The other player informs the system of their choice on performing the action.

1d. If the role’s ability is performed on other game objects, the game object changes with respect to the ability’s description.

3a. If the *Player* does not use up the special AP bar, the special AP bar will not be saved for the *Player* to use in next turn.

**Advance Fire**

**Use Case:** Advance Fire

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** The system will proceed the game by adding more fire.

**Primary Actor:** *Fire Manager*

**Main Success Scenario:**

1. The *Fire Manager* randomly picks a position on map to advance fire. The following are the possible cases of advancing fire.

* Advance fire on an empty tile adds a smoke to the space.
* Advance fire on a smoke causes the smoke turn to fire.
* Smoke adjacent to fire will be lighted up to fire.
* An explosion occurs.

1. The *Fire Manager* detects if any *Player* is affected by the advanced fire
2. The *Fire Manager* detects if any POI is affected by the advanced fire and remove the POI.
3. The *Fire Manager* detects the damage level of the house.

**Extension:**

1a. If the position of advanced fire has a hotspot, repeat step 1.

1b. If the position of advanced fire contains hazmat and fire, a new hotspot is placed; use case continues at step 2.

1c. When fire advanced on multiple hazmat.

1c.1. The *Fire Manager* informs *Player* to choose order of hazmat explosion.

1c.2. The *Player* informs the system of their choice of order.

1c.3. *Fire Manager* performs explosions in *Player* desired order; use case continues at step 2.

1c.4. *Fire Manager* performs explosions in random order if the player does not pick order of explosion on time; use case continues at step2.

1d. If the explosion happens near walls/doors.

1d.1. The wall/door is destroyed by the system.

1d.2. The *Fire Manager* increases damage level of house; use case continues at step 2.

1e. If the advance fire causes a shock wave.

1e.1. The *Fire Manager* follows the shock wave to an empty space/smoke/door/wall.

1e.2. The *Fire Manager either adds a fire to the empty space/smoke or adds a damage to the wall/door.*

1f. If the advance fire causes the house to collapse, the *Fire Manager* informs players if they won/lose.

2a. Player(s) is on a location with fire on.

2a.1. The *Fire Manager* asks the player to choose a valid location to respawn.

2a.2. The *Fire Manager* respawns the player on the location they choose/ on a random valid location if the player does not choose location to respawn in a given time; use case continues at step 3.

3a. Removing POI will result in failure to rescue victims.

3a.1. *Fire Manager* detects a victim involved in the fire and removing the victim will fail the rescue operation.

3a.2. *Fire Manager* informs players they lost the game; use case ends in success (game over).

4a. The house is destroyed due to too much damage taken.

4a.1. *Fire Manager* detects the level of damage is above maximum.

4a.2. *Fire Manager* informs players if they won/lost the game; use case ends in success (game over).

**Replenish POI**

**Use Case:** Replenish POI

**Scope:** Flash Point

**Level:** Subfunction

**Intention in Context:** The *POI Manager* will proceed the game by replenishing POI.

**Primary Actor:** *POI Manager*

**Main Success Scenario:**

1. *POI Manager* randomly generates a position for placing the POI.
2. *POI Manager* places the POI to the location.
3. System informs all players the updated map and POI status (list of POI not yet put into the map).

**Extension:**

1a. If the randomly chosen position has fire, the POI is placed to a new location without fire following the trace on the map.

2a. The POI is placed on the same location of a *Player*.

2a.1. The POI is immediately revealed to players by the system.

2a.2. *POI Manager* discards the POI immediately if the POI is a false alarm and repeat step 1.

2a.3. *POI Manager* updates the list of POI.

2b. There is no more POI to be placed.

2a.1. *POI Manager* checks if there are POI that are not yet placed onto the map.

2a.2. When no remaining POI detected, *POI Manager* checks for existing POI on map.

2a.3. When no POI on map, *POI Manager* informs system to end game.

2a.4. System presents all players if they win/lose.