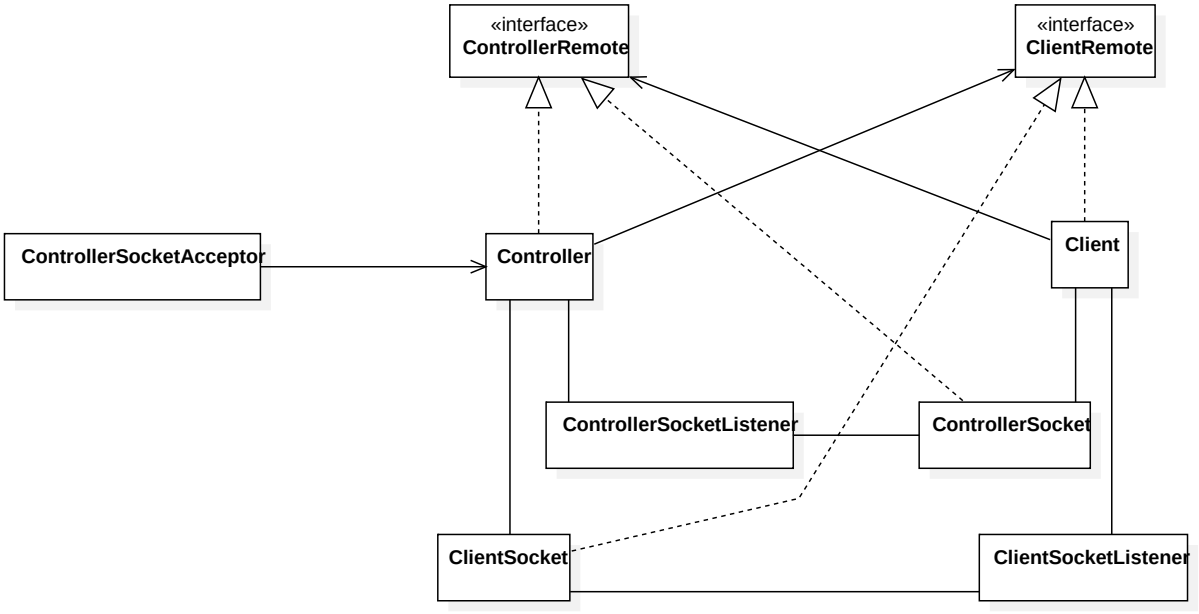
INTRODUCTION

The network in our project supports both RMI and Socket technologies. Socket is build to let the client and the server invoke the same remote methods used in RMI, so, once descripted the protocol used in socket communication, the protocol used in the RMI communication is the same for both technologies. Here is the reference scheme of the network



SOCKET PROTOCOL

We used socket programming to write a network implementation based on the same paradigm of RMI. More specifically, client and server communicate using messages structured as follows:

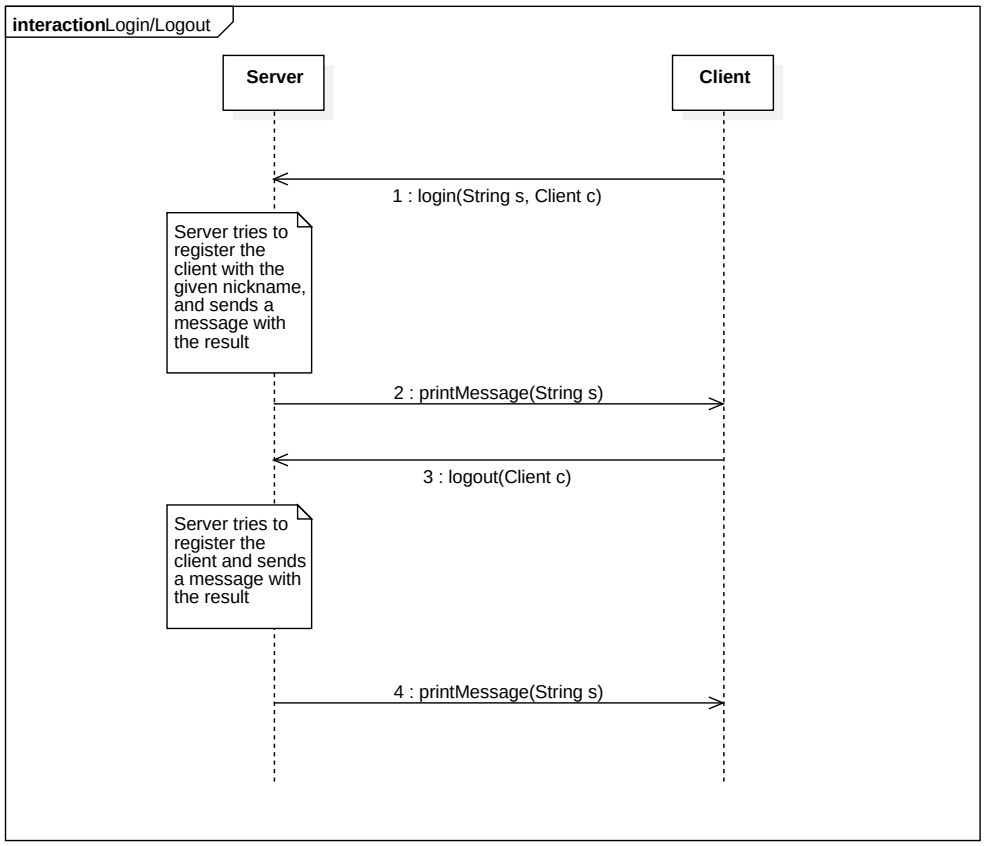
methodName ; *serializedParameters*

When a message like this is received, it is processed by a parser which identifies the requested method and invokes it using the correct, de-serialized parameters.

EXAMPLE: login;Bob
The parser invokes the methods associated to the keyword “login” passing a string containing “Bob” as parameter.

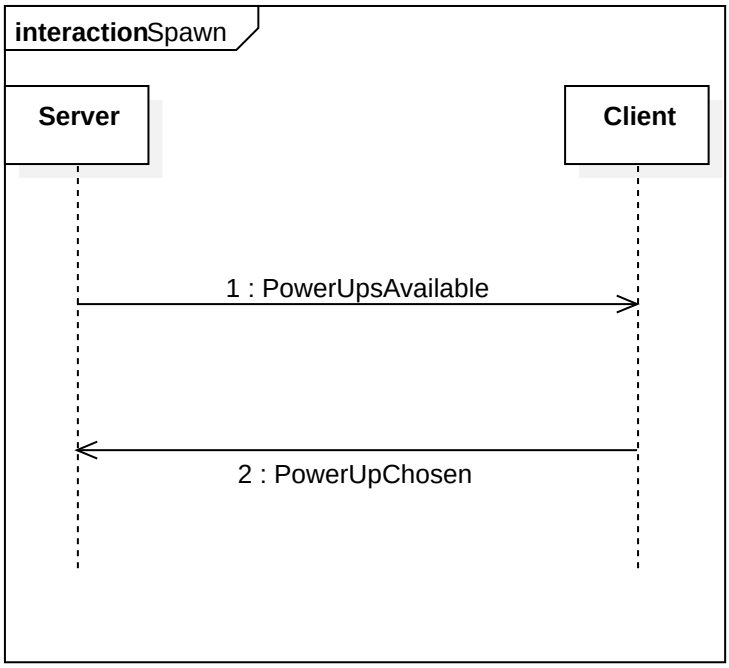
COMMUNICATION PROTOCOL

INITIAL PHASE



SPAWN

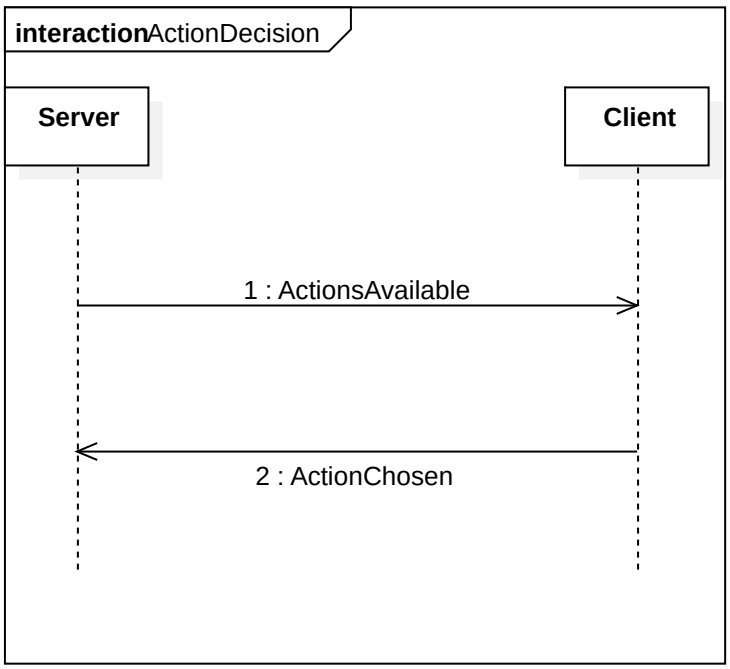
This interaction happens both during the first turn and when a player is killed and needs to respawn.



PowerUpsAvailable: ArrayList<PowerUp>. The list of powerups that the player has.
PowerUpChosen: PowerUp. The powerup chosen by the player to respawn.

ACTION DECISION

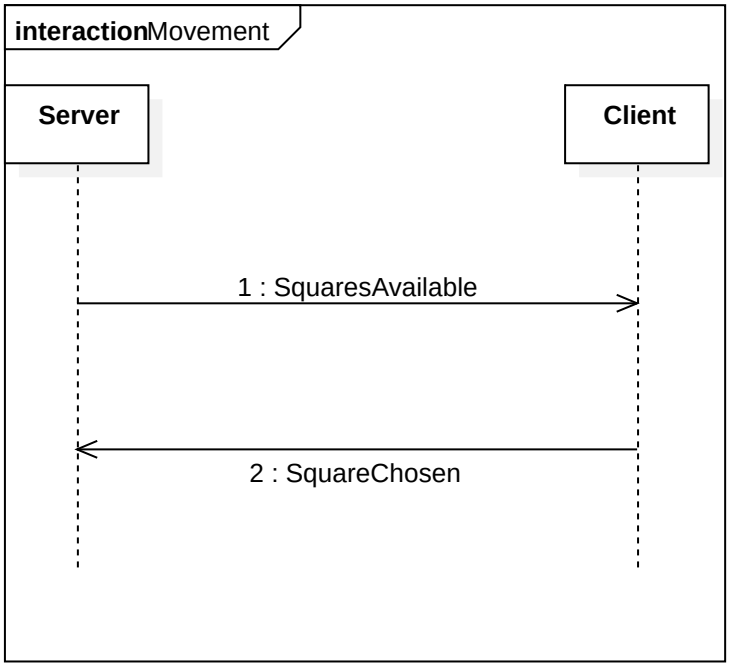
This interaction happens when the player must decide what to do with his action.



ActionsAvailable: ArrayList<String>. The list of actions that the player can do.
ActionChosen: String. The action chosen by the player.

MOVEMENT

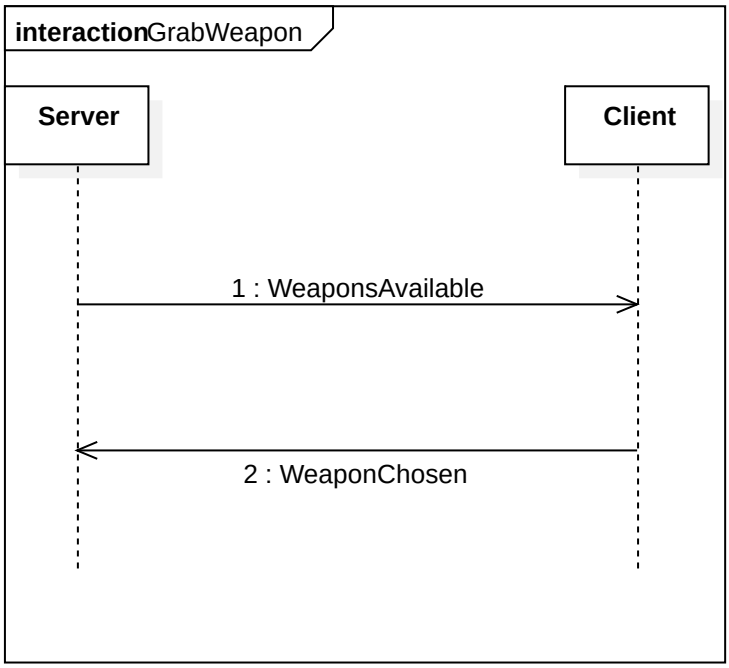
This interaction happens when the player decides to move himself or another player. It can be caused by an action, a weapon or a powerup.



SquaresAvailable: ArrayList<Square>. The list of squares where the player can move himself or another player.
SquareChosen: Square. The destination chosen by the player.

GRAB WEAPON (< 3 WEAPONS)

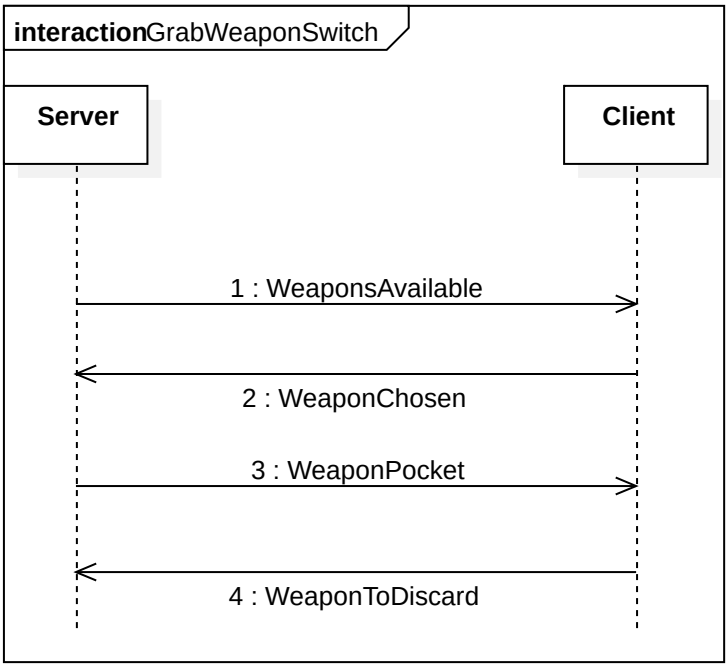
This interaction happens when a player grabs in a SpawnSquare and has less than three weapons. There is no need for an interaction when the player grabs in a TileSquare, because the all the PowerUps and Tiles gets added and the player doesn't have to choose anything.



WeaponsAvailable: ArrayList<Weapon>. The list of weapons that the player can buy in that SpawnSquare.
WeaponChosen: Weapon. The weapon that the player wants to buy.

GRAB WEAPON (3 WEAPONS)

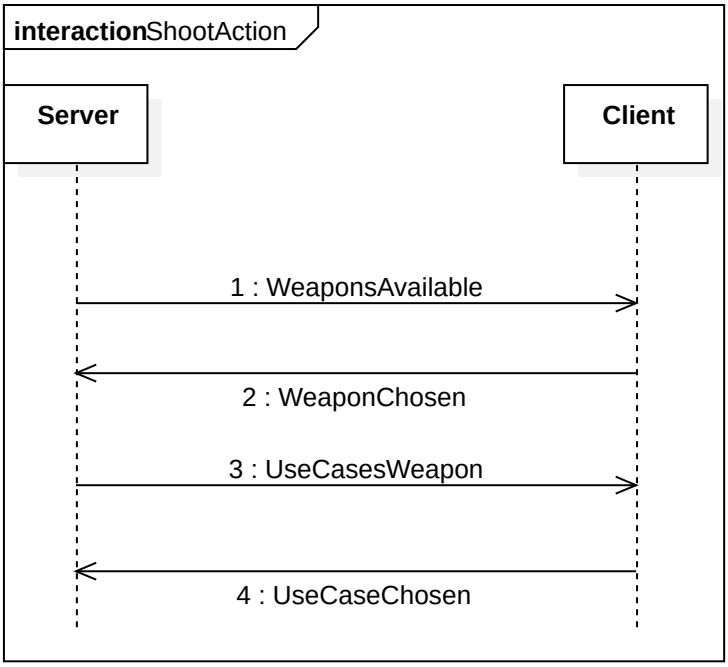
This interaction happens when a player grabs in a SpawnSquare and has already three weapons.



WeaponsAvailable: ArrayList<Weapon>. The list of weapons that the player can buy in that SpawnSquare.
WeaponChosen: Weapon. The weapon that the player wants to buy.
WeaponPocket: ArrayList<Weapon>. The list of the weapons of the player.
WeaponToDiscard: Weapon. The weapon that the player wants to discard.

SHOOT ACTION

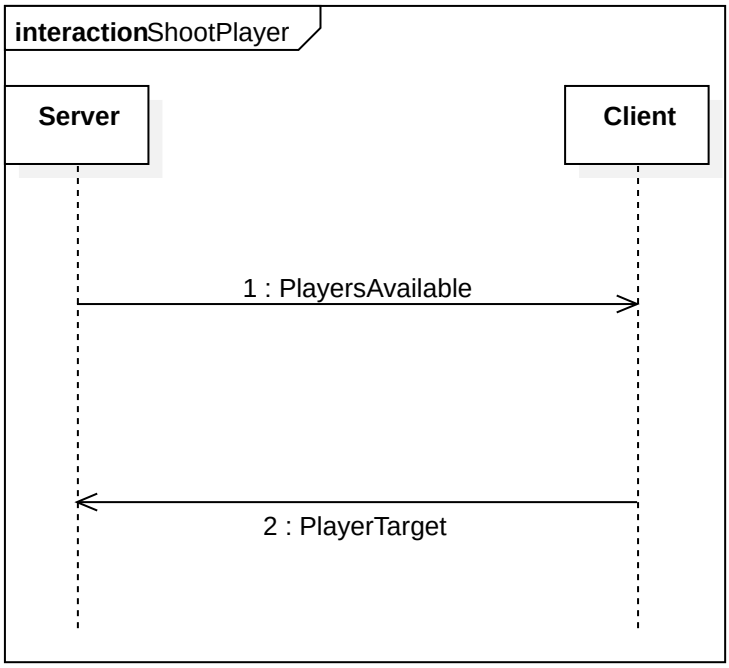
This interaction happens when a player decides to shoot.



WeaponsAvailable: ArrayList<Weapon>. The list of weapons that the player can use.
WeaponChosen: Weapon. The weapon that the player wants to use.
UseCasesWeapon: ArrayList<String>. The different methods of use of the chosen weapon.
UseCaseChosen: String. The method of use chosen by the player.

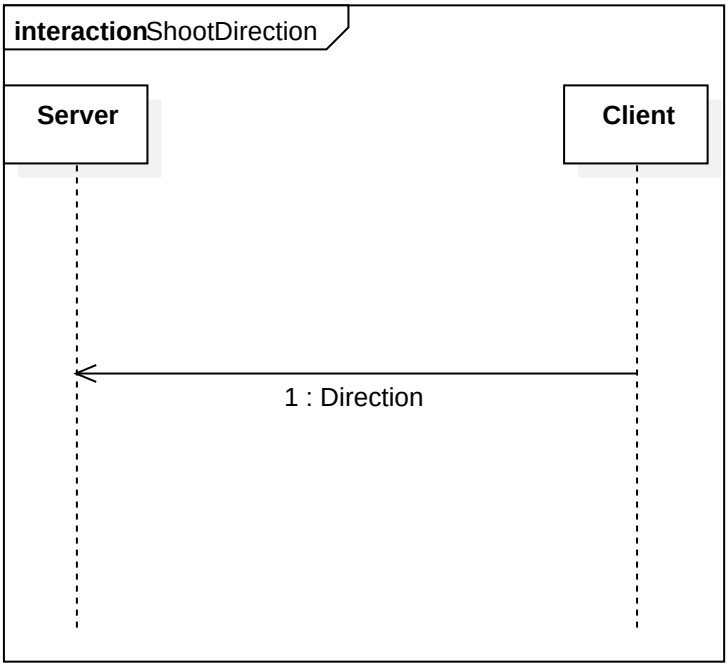
After the choosing of the method of use, the player will have to choose different type of targets (sometimes he won't have to choose).
There are 4 different interactions that can happen when using a weapon:

1. Choose another player



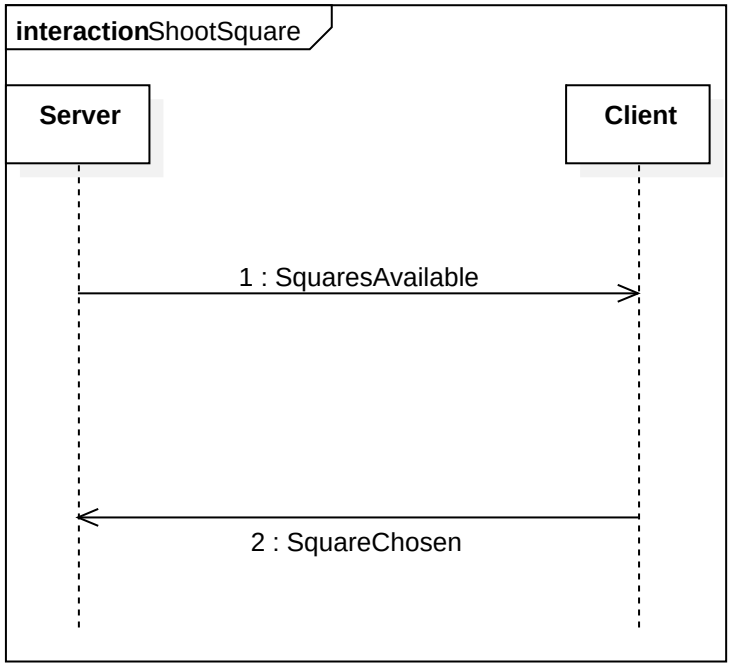
PlayersAvailable: ArrayList<Player>. The player that can be hitten with the weapon.
PlayerTarget: Player. The player that has to be hit from the shot.

2. Choose a direction



Direction: String.
The Server doesn't send an object because there are always the 4 possible directions where to shoot.

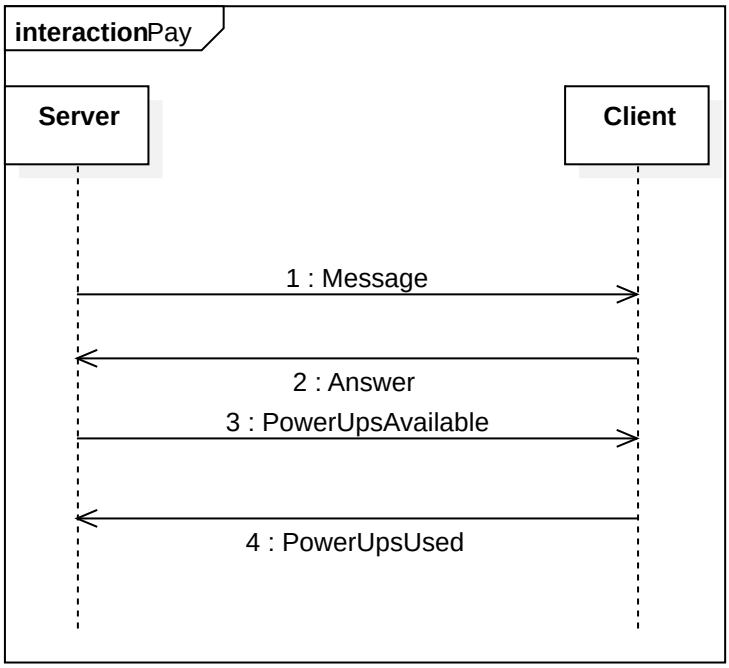
3. Choose where to move (already discussed before)
4. Choose a square where to shoot



SquaresAvailable: ArrayList<Square>. The list of the squares where the player can shoot.
SquareChosen: Square. The square where the player decided to shoot.

PAY

This inteaction happens when a player has to pay.

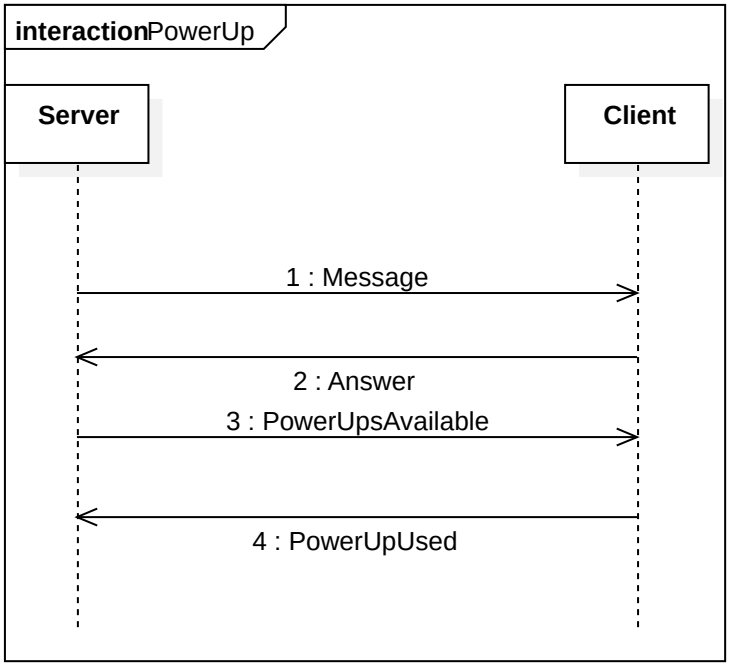


Message: String. Represents the question: “Do you want to use some powerups to pay?”
Answer: Boolean. The answer to the previous question.
PowerUpsAvailable: ArrayList<PowerUp>. The powerups that the player has got.
PowerUpsUsed: ArrayList<PowerUp>. The powerups that the player wants to use to pay.

The messages 3 and 4 are sent only if the answer is “true”.

POWERUP

This interaction happens when something happens and a player (the player of the turn or another player) can use a powerup.

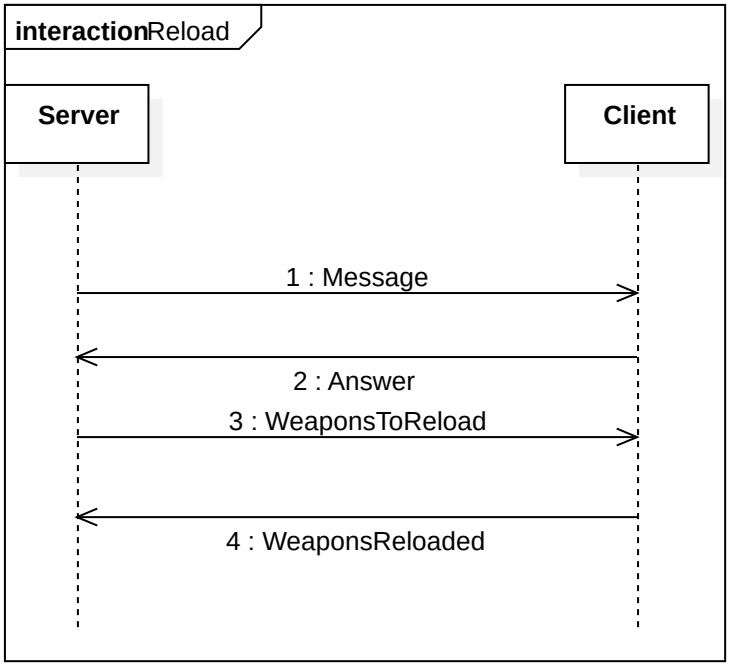


Message: String. Represents the question: “Do you want to use a powerup?”
Answer: Boolean. The answer to the previous question.
PowerUpsAvailable: ArrayList<PowerUp>. The powerups that the player has got.
PowerUpsUsed: ArrayList<PowerUp>. The powerups that the player wants to use.

The messages 3 and 4 are sent only if the answer is “true”.

RELOAD

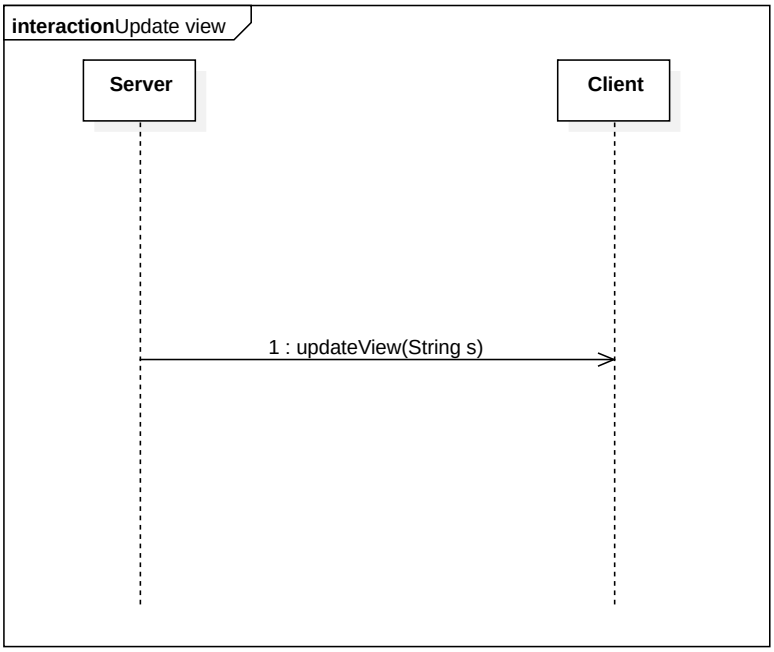
This interaction happens when the player can reload.



Message: String. Represents the question: “Do you want to reload?”
Answer: Boolean. The answer to the previous question.
WeaponsToReload: ArrayList<Weapon>. The weapons that the player has got and that aren’t loaded.
WeaponsReloaded: ArrayList<Weapon>. The weapons that the player wants to reload.

UPDATE THE VIEW

At the end of the turn, if the player hasn’t done errors, the effects of that turn are applied to the model. In that moment, the server will send an update to the views of the active players.



END GAME

When the game is ended, the server will calculate all the points and will send the final classification of the players to every player.