Communication framework

Client-server communication and error handling

The way communication works in our implementation, is by sending location updates with a fixed time interval. Each client sends its location to the server every second, the server then adds a timestamp to it and sends that location to all active clients. They keep track of a small history of all the active players. If a player hasn’t sent his/her location in the past five seconds, he/she is considered ‘inactive’ or ‘offline’ and is displayed as such (in a faded colour).

If the server has not received a ping after 30 seconds, the player is removed from the locations list and is regarded to be no longer participating in the game. The player can join the game again when he reconnects and is then added back to the ‘active locations’ list.

# Dropping connections

When the client does not send an acknowledgement within those 5 seconds (due to a communication error or bad signal), the player is still regarded as participating in the game. Other players can initiate a battle, tap on the player, etc. (but the other player is still put on hold until a two way acknowledgement is set-up for battle). If the lack in communication persists for 30 seconds or more, the player is regarded as no longer participating in the game. If other players are waiting for a battle to instantiate, they are notified that the player is offline, and any communication between the online and offline player is terminated.

If the browser is put in the background or the phone is locked, the client-server communication is dropped.

# New implementations

In the second part of the project, the team is asked to still keep track of ‘older locations’ of players. Removing an inactive player from the locations list is not feasible in this scenario, as the information of the last location is lost. But this sort of location can be kept separately from other active locations that do need to be updated every few seconds, as these remain constant anyway. This will reduce server stress.

# Example

Consider the following example. Player X and player Y are (amongst others) playing a game. X and Y are near each other and plan to gather and battle. A moment before the meetup, player X is called. The game in X’s browser is thus put in the background.

Now, player Y wants to initiate battle and sends a signal to X. Since X is calling, he/she won’t acknowledge Y and the battle will not be instantiated. Y will soon realise that X is currently not responding, first by the visual clue, later by the pop-up notification and disappearance of X on the game map. Player Y will now choose to whether to wait and see if X comes back online, or simply to search another player to battle.

When player X is done calling and reopens his/her browser, he/she is added to the game again.