Pokémon Battle Simulator

Human Computer Interaction

Luigi Ariano
luigi.ariano@stud.unifi.it

Edoardo Bonanni
edoardo.bonanni@stud.unifi.it

Master's Degree in Computer Engineering

Università degli Studi di Firenze

2021-2022

Abstract

Pokémon Battle Simulator *is a game that aims to simulate the real Pokémon battle games. In the application are implemented all features and functionalities that allow the player to choose his team and play a match in* ***singleplayer*** *or* ***multiplayer*** *mode.*

In this report, we will illustrate the entire ***development process*** *of the application itself, starting from the motivations, passing through the implementation, models, wireframes and mockups, up to show the complete game and the several usability tests.*

1 Introduction

The Pokémon is one of the most popular anime of our generation and alongside the games has formed a huge community around this world. In Pokémon, humans, known as **Pokémon trainers**, catch and train Pokémon to battle other creatures for sport. Our game basically aims to reconstruct the old Pokemon games created around this world, starting with an analysis of the characteristics of the inserted Pokémon, choosing your personal team with whom you can play and improve with other trainers or with the CPU and refining the strategies that are behind a battle of this type.

We tried basically to make the application as realistic as possible compared to a real Pokemon game, trying to involve even inexperienced gamers by entering all the Pokemon statistics necessary to start a game. However, a little prior knowledge is needed on the effectiveness of the types of the various moves and just like the real games, experience is the only weapon that allows you to improve and win many matches.

1.1 Motivation

Pokémon games have always been owned by one of the most important and powerful game companies, namely Nintendo. Because of that, games have always been developed for certain types of consoles, all owned by Nintendo itself, limiting the possible players who can enjoy the various games.

Essentially, our idea was to develop a Pokémon game in **python** that could run on any type of PC. On the web there are several open-source Pokémon battle simulators but they have many bugs in them or a lack of graphical interface that doesn't allow the user to simulate a real battle as if he purchased the original game. All obviously motivated by our passion for this world.

Beyond that, we decided to develop the game in both singleplayer and multiplayer modes, in particular we paid attention to the latter one as it was not cared for the classic Pokémon games, indeed it was difficult to find other gamers with whom to play to a real challenge between Pokémon trainers.

We had to do some **brainstorming** sessions to define the key points of the project, starting from the dynamics of the game and trying to have a clear interface that would allow users to understand the various phases and be able to play with **extreme ease**.

This has led to the creation of a very sim-

ple interface for choosing Pokémon in the team, where the user can modify his team in complete freedom and analyze the possible strategies to use. Subsequently, another interface allows the choice of the game mode, to then enter in the real battlefield; in this case we wanted to graphically simulate a real Pokémon game in order to be understandable for a beginner but absolutely familiar for an experienced trainer.

2 Implementation

Before getting to the full game, we defined all the dynamics around the game and then we concentrated on the graphic interface of the various windows, starting from the wireframes drawn on **Photoshop**, then defining mockups developed using **QtDesigner** and a basic version of the **pygame** graphics. Finally, the final graphical interfaces have been made in such a way that they can be pretty neat and similar to the classic Pokémon games, but also understandable and accessible to all those who had never tried this type of games. Obviously, these interfaces allow to interact with Pokémon and their moves by the rules of the game, while also integrating animations and sounds for greater player involvement. Now let's see some key points that allowed us to define the entire development of the application.

2.1 Models

In the initial phase of the implementation, we defined models that allowed us to manage and keep track of all the features inherent to the main instances of the game that is the actors of the application.

Model is the base class that contains the other models of the project, in particular it has two instances of the Player class, that represents the user and the enemy and the Pokédex which contains the whole list of Pokémon with their moves.

Player class represents a Pokémon trainer, who is uniquely identified by an ID and as attributes it has its name and the team that it has chosen for the fight.

Pokédex class contains a list of all Pokémon in the game, a list of all usable moves and a dictionary containing type advantages among Pokémon. This class is required in order to replicate the dynamics of the original game.

Pokémon is the class that represents each individual Pokémon, it has several attributes including its name, its types, its stats and the moves it can use.

Move class represents a Pokémon move and has several attributes including a unique ID, move name, description, nature, power, accuracy, number of uses of the move (**PP**) total and remaining and finally a characteristic that identifies the type of move (i.e. physical, special or state).

The informations relating to Pokémon, moves and type advantages are read by specific pre-filled **csv** that provide us with all the data necessary to simulate the game behavior.

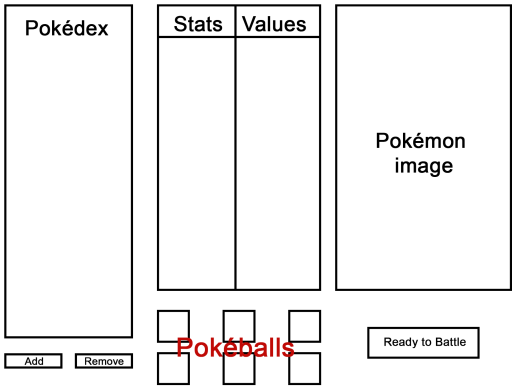
Starting from these models, all the project dynamics have been defined including the course of the turn, attacks and their types,

changes to the stats and HP of the Pokémon. We have also implemented the behavior of some particular moves, which modify the classic course of the turn and the display of Pokémon on the battlefield with the relative game strategies (i.e. fly, dig, solarbeam, hyperbeam and so on that could be a problem if the trainer doesn't know their strange behaviours).

2.2 Wireframes, Mockups and Final Game

The interface was one of the key nodes of the entire project development, as we had to think a lot to obtain a **simple** but **impactful** GUI that could be clear at first use and exploit the potential included in the elaborate.

We started by creating wireframes of the main game windows through drawings on Photoshop to schematize their structure in order to have a clear scheme about **what** we want to insert in the application, **where** and **why** it's necessary to have that features by eliminating all the superficial objects in the windows.



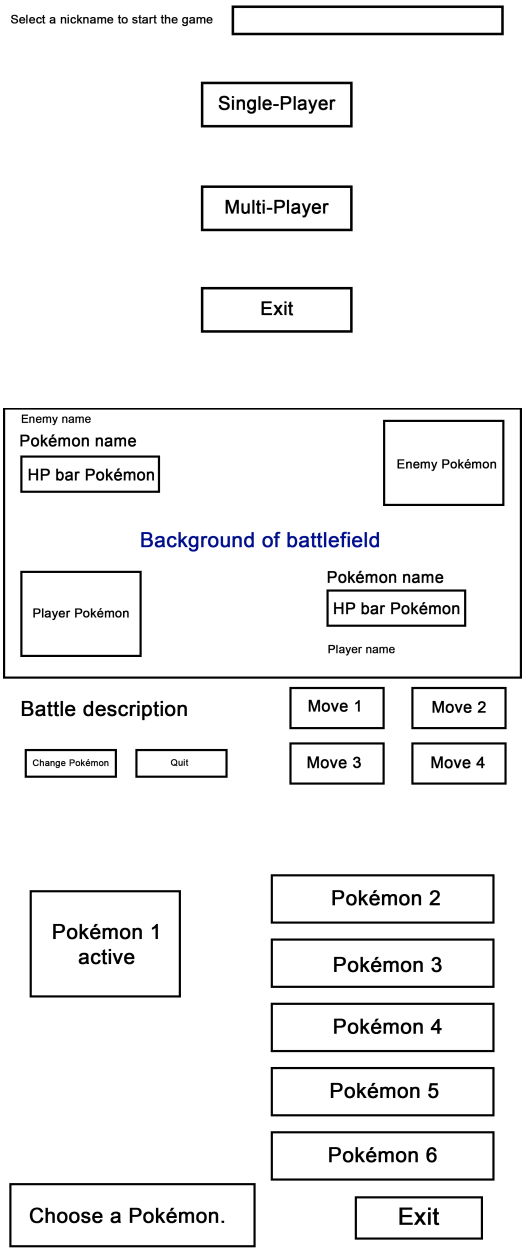
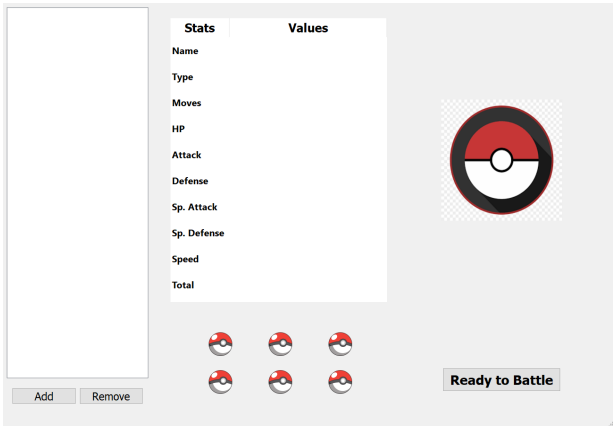
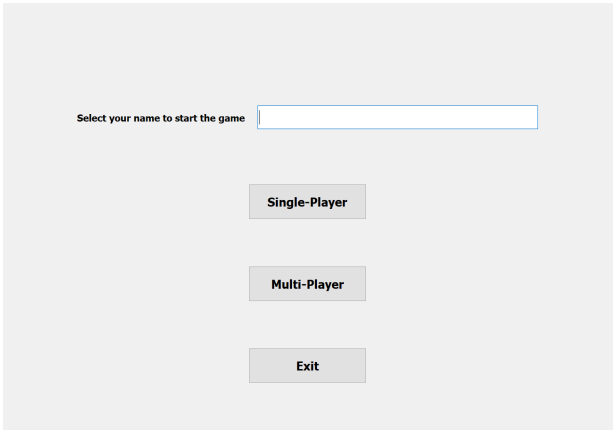
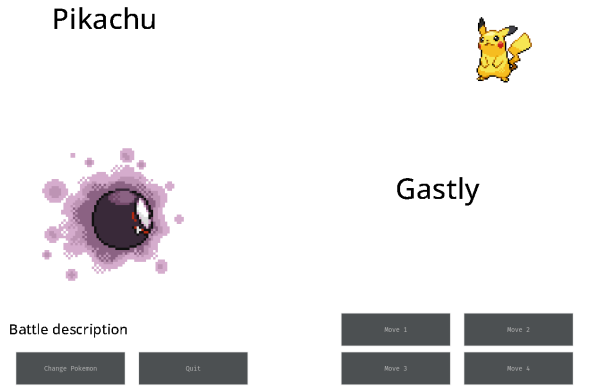


Figure 1: The images show the wireframes of PokemonChoice, GameModeSelection, BattleField and TeamChoiceMenu windows.

The second step was to create the mockups of all windows by using QtDesigner for PokemonChoice and GameModeSelection in order to obtain a preliminary version of our GUI. The same modus operandi is repeated for the BattleField and TeamChoiceMenu windows but being pygame windows we had to design basic graphics using the functions of this package, therefore it was more complex to implement the basic structures necessary for the functioning of the project.







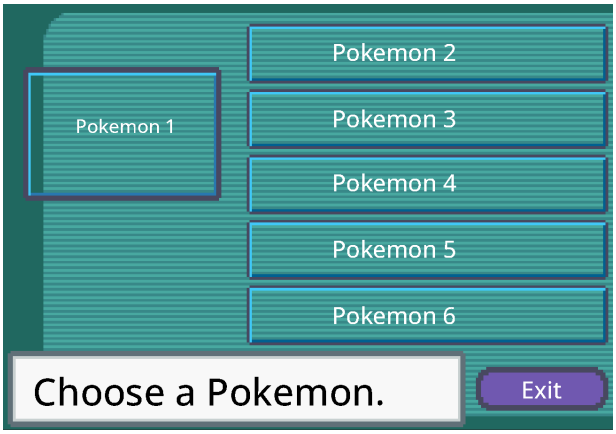


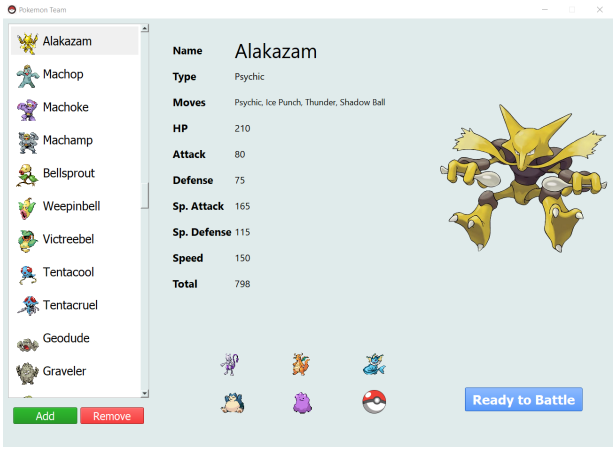
Figure 2: The images show the mockups of PokemonChoice, GameModeSelection, BattleField and TeamChoiceMenu windows.

In the end, we created the graphical interfaces in their entirety by adding colors, a style, gifs and images in order to make everything extremely understandable, easy to use and pleasing to the eye.

The interaction with the GUI is simple and clear, the various buttons are explanatory with respect to the event they generate, allowing for example to add/remove Pokémon

in the team, choose the game mode, the moves that the Pokémon have to do or change the Pokémon in battle with another in the team. These interactions are related to transitions/animations that the game makes so that the user can adapt to the events that happen and not feel confused or lost.

Obviously, the user can return to the Pokémon choice from the next window or leave the game for any reason, in order to leave him free to browse the various pages.



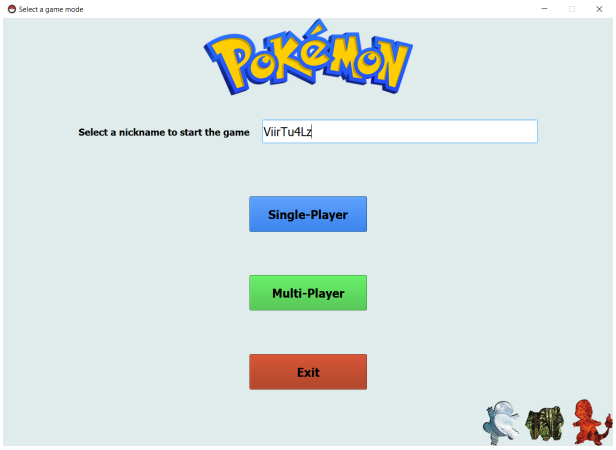


Figure 3: The images show the real game screenshots of PokemonChoice, GameModeSelection, BattleField and TeamChoiceMenu windows.

2.3 Game modes

The singleplayer mode consists of a battle between the trainer and the CPU, in fact this allows an user to train itself and learn all the strategies and techniques present around these Pokémon battles. The CPU is not ex-

tremely smart, it is assigned a random team for the match and its Pokémon makes moves always in a random way in order to not be too difficult for a beginner. Therefore, the idea is that an inexperienced trainer tries this mode, improves and can be ready for real battles with experienced trainers in the multiplayer mode.

The multiplayer mode consists of a fight between two different trainers, thus making the mode extremely compelling and exciting for any type of Pokémon trainer. To do this, it is necessary to set up a **client-server** communication, where each player can do his own moves, make choices for his team or also leave the game. The two players that will fight will be considered as clients, therefore, it is necessary to start a **server** that allows to manage the communication between these two entities.

In particular, in python there is the library **socket** that allows to define a communication of this type. When the server is started, it defines its **IP** (local or global depending on its use), its **port** and the maximum number of clients that can connect to it at the same time. Then, the server waits for possible client requests to initiate communication and when it receives a request, it will execute a thread that will allow to manage the communication between that particular client and the server itself.

As previously mentioned, in our case the clients connected to the server must necessarily be two, so the server as soon as it receives a message from one of them will forward it to the other.

A client to be able to connect with the server must use a custom **Network** instance that requires the IP and port of the server to initiate a communication.

The client, after having sent the first packet indicating to the server that it wants to establish a **TCP** connection, waits for packets from the latter; once the connection between these entities has been set, both exchange empty packets continuously in order not to lose the communication established due to the set timeout. However, when a trainer wants to interact with his move or choice towards the other player, the packets exchanged between the server and the clients will no longer be empty but will contain all the information necessary for the game.

Therefore, to respect the game turns it is necessary that both trainers send their moves or choices, waiting for the opponent's move if he has not already made them. The informations that the players exchange during the turns to execute the correct performance of the game conceptually concern the updating of the models, so they are:

- the enemy trainer's name;
- the Pokémon team that the enemy chose for the battle;
- the kind of choice the enemy made (i.e. the Pokémon move in battle, its possible exchange with another one in the team or quit the game);
- the choice of the opposing Pokémon to replace the exhausted one in game.

The custom **Message** class allows clients to send their moves/choices to the other

player, so when a client has to send these informations it inserts the content of the message in the data attribute of the class relating to the corresponding player and sets the variable of sending. Once both clients have completed these operations, they will check the data attribute relating to the enemy player and discover his choice.

Obviously, for both modes it is necessary to insert an imaginative **nickname** (just for fun) and be careful about the words chosen, since no checks on the input words have been carried out.

2.4 Animations and sounds

We have decided to include **animations** and **sounds** within the game to make the game more realistic. The animations concern various things within the battle, i.e. the visualization of the trainers and the Pokéballs from which the Pokémon come out at the beginning of the game, the explosions generated by the attack of one Pokémon towards the other, the HP bar decrease slowly, the Pokémon change in battle with another one and the animations about the special moves.

This also allows us to have a minimum of **dynamism** within the project because, for example, also the PokemonChoice window has within gifs for the entire Pokédex making everything more likely to the original versions of the game.

Even sound generates a sense of involvement, indeed, we set a soundtrack, a sound for explosions due to attacks and also another one for the victory/defeat of the trainers in order to point out the end of the battle.

3 Usability Tests

The final step of this entire process is the testing phase. We developers have run a lot of tests on the game in order to check for the possible presence of some bugs. Obviously, some types of moves (i.e. special moves) can cause a greater problem to the development of the game precisely due to their nature that modify the turns themselves. In practice we have analyzed and tested all the possible events that our application manages (and not), so starting from the choice of Pokémon for the team to the conclusion of the battle all the possible behaviors are evaluated.

The test is structured around a series of questions or statements that the participants had to answer. After users have tried the game, they would have to answer a **SEQs**, i.e. a single ease questions about how difficult or simple they found certain features implemented. The single ease questions are questions to which the respondent can reply with an integer from 1 to 5 (both included) that are used to measure the user experience. They are a simple metric both for the respondent to understand easily and for developers to analyze the results and gives us the same quality of results as other more complex metrics.

Our idea was to have 10 people test our game, taking 5 **beginners** who have never played a real Pokémon game and 5 people who are **experts** instead of these games. We made all these people try the game **twice** in order to give us evaluations related to the first and second use of the application and trying to diversify these evaluations between

expert and beginner trainers. This allowed us to have an opinion as heterogeneous as possible with questions/statements to be evaluated according to everyone's opinion and others to be answered as true or false, trying to sample users with different backgrounds.

What we expect to see is that experts players are already confident from the first test with the interface and interaction of the game, the behavior of the battle and all the events that concern it, i.e. animations, special moves that modify the turn and Pokémon choices. Probably the multiplayer mode will turn out to be more engaging and stimulating for any type of player than the singleplayer mode which is more suitable for a beginner.

Inexperienced players, on the other hand, will feel less confused in using the game during the second test, having had a prior approach to the application unlike experts trainers whose gaming experience shouldn't change much between the two tests.

The total number of questions to which the expert trainers had to answer are **15**, instead, for the others, the question (10) relating to the comparison of the game experience of our simulator with a real Pokémon game was not asked.

In the table 1, we can see that the beginners appreciated the mechanics of the battle, the game was exciting and engaging, in particular, the singleplayer mode was on average more difficult and complex to play for this type of people.

The least appreciated or understood things were the stats and properties of the Pokémon, the animations and the moves that modify the turn which ask for a previous knowledge of the game.

The second test showed a greater understanding of the game in all its features, a higher appreciation for the animations and the ease with which inexperienced trainers duel against the CPU in the singleplayer mode, highlighting a superior understanding by the player regarding the battle.

The expert trainers quickly understood the goal of the game with all its features, an immediately positive and intuitive feeling in the interaction with the application itself.

The game turns out to be much more competitive in the multiplayer mode as it is a challenge between experienced people with a game strategy. In this case, the animations were not appreciated since compared to the original games they were numerically inferior and so repetitive; the speed of the simulator was evaluated slightly slow in subsequent tests probably because an expert trainer prefers a higher dynamism of the application.

3.1 User advices

A final request made to all users who have tried the game was to write the strengths of the application, what they liked the most and all the possible problems or suggestions they have seen or thought.

The **comments** were very explanatory and made us reflect on the problems or not of the game: the interface with which users approached in its simplicity and clarity was generally appreciated but we were suggested, for example, to insert a bar search to find the Pokémon in the initial list, an option

that allows to choose the difficulty of the singleplayer mode, specify other stats and properties of the Pokémon, increase the animations and develop the double Pokémon mode for each trainer. Some game mechanics are not clear to inexperienced trainers such as the effectiveness of certain moves on some Pokémon or the fact that some of them change the dynamics of the game.

In particular, we appreciated a **comment** made by a user that we want to quote for his **honesty** on what he thinks of our simulator: "The game is well designed, I enjoyed playing it and reminded me my childhood time. In the singleplayer mode, I would suggest to rebalance the Pokémon selected by the computer in order to make the game more competitive (otherwise it may happen that a high level Pokémon has to battle against a much lower level one). Morevore, in the starting page where you have to select your team, I would suggest to let appear the button *Ready to Battle* only once you have selected all your team members."

4 Conclusions

In this report, we described our Pokémon Battle Simulator that is a game that simulates Pokémon battles, trying to get as close as possible to real games with animations, sounds and menus designed specifically for this but also introducing new features inside. In fact, before the battle it is possible to view the entire Pokédex with the related Pokémon and all the statistics concerning them, the two singleplayer and multiplayer modes that are present allow any type of expert trainer or not to compare itself with its skills and its limits. We tried to get a game that was dynamic and understandable for any type of user, getting good feedback from those who tried it.

In conclusion, thanks to the usability tests we were able to understand the critical issues of the application such as insert a bar search to find the Pokémon in the initial list, an option that allows to choose the difficulty of the singleplayer mode, specify other stats and properties of the Pokémon, also some game mechanics are not clear to inexperienced trainers such as the effectiveness of certain moves on some Pokémon.

Despite some problems, the general **feedback** is very positive, the game was understood, appreciated and enjoyable for different types of users, making us satisfied with the application developed.

Surely, in future versions, we intend to improve all the critical issues that are exposed to us, add new features such as double battle, game difficulties or the full screen mode that gives a greater involvement for Pokémon trainers.

Questions

1) Is the goal of the game clear?
2) Is the choice of the team's Pokémon simple and understandable?
3) Evaluate the difficulty of adding, removing and replacing the Pokémon of team.
4) Are the stats of the Pokémon clear?
5) Is access to the game modes understandable?
6) The choice menu of the Pokémon to replace (if for example the active Pokémon is exhausted) is clear and understandable.
7) The animations of the game are nice.
8) Is the course of the turn clear in the case of using moves that modify the normal execution (i.e. fly, dig, solarbeam, quick attack, etc.)?
9) Evaluate the game's speed.
10) Is the gaming experience similar of classic Pokémon games?
11) The game is easy to use after a short time.
12) Is singleplayer game mode challenging compared to multiplayer?
13) Does the game work correctly or does it have unexpected behaviours (i.e. it run different moves than the one indicated, it doesn't change the Pokémon chosen to insert in battle or the turn does not work properly compared to the choices made)?
14) Is the game exciting and challeging in its game modes?
15) Would you play the game again and recommend it to others?

	First Test	**Beginners**	**First Test**	**Experts**	**Second Test**	**Beginners**	**Second Test**	**Experts**
Questions	μ	σ	μ	σ	μ	σ	μ	σ
1	4	0.63	4.8	0.4	4.4	0.8	4.8	0.4
2 (T/F)	1	0	0.8	0.4	1	0	1	0
3	3	0.63	3.2	1.6	3.8	0.4	3.8	0.75
4	2.4	0.8	4.2	0.4	3.2	0.75	4.2	0.4
5 (T/F)	1	0	1	0	1	0	1	0
6	3.8	0.75	4.8	0.4	4	0.63	4.8	0.4
7	3.6	0.49	2.8	0.75	3,4	0.8	3.2	1.17
8	2	0	4.4	0.8	3.4	0.49	4.4	0.8
9	3.4	0.49	3.6	1.02	3	0.63	2.6	0.9
10*	-	-	4.6	0.49	-	-	4.8	0.4
11	3.4	0.49	5	0	4.2	0.4	5	0
12	4.2	0.4	3.2	0.4	3.2	0.4	3.6	0.49
13	3.8	0.98	3.8	1.6	4.2	0.98	4	0.63
14	3.8	0.4	4.2	0.75	3.6	0.49	4.2	0.4
15	4.4	0.49	3.8	0.75	4.4	0.49	3	1.17

Table 1: Table shows the results of first and second test for beginners and experts players. For each question/statement we show the mean rating (μ) and the standard deviation (σ). The results are preceded by the questions/statements made in the usability tests to the players. *Question asked only for experts trainers.