Reflection on Iteration # 4

Context Project: Computer Games

Group: 8

User Story #	Task					
	Id	Assigned to	Estimated effort	Actual effort	Done	Notes
Implementing multiplayer	Task 1	Joost and Thijs	5 points	+/- 20 hours	Yes	During implementation concerns for a better network architecture emerged.
	Task 2	Thijs and Soheil	5 points	+/-10 hours	Yes	During the implementation of this task the need for a server client architecture became clear.
	Task 3	Rob	3 points	+/- 3 hours	No	We decided that it was best to implement a complete sever-client architecture before actually synchronizing scores.
Passenger stealing	Task 1	Rob	3 points	+/- 7 hours	Yes	Implementing this task went according to plan.
	Task 2	Aidan and Joost	4 points	+/- 7 hours	Yes	Implementing this task went according to plan.
	Task 3	Aidan and Soheil	4 points	+/- 4 hours	No	We decided that it was best to implement a complete sever-client architecture before actually integrate the passenger stealing with the multiplayer functionalities.
Testing	Task 1	Rob and Aidan	5 points	+/- 20 hours	Yes	Encountered some problems with configuring continuous integration on Jenkins, classes from external libraries could not be loaded (but this problem has been fixed). As well there was a problem with testing classes that need to import resources like sprites.

Main Problems Encountered

Problem 1

Description:

The current network architecture is not feasible. Currently, the network is a complete peer to peer system. Every message a client wants to share with other players is broadcasted to all other players. This results in inefficient network usage but that is not the main problem. The main problem of this architecture is the absence of a central authority, resulting in desynchronizing or conflicting game states.

Reaction:

This problem led to some proper research, that we probably should have done at the start of the implementation. This research gave us the client-server architecture as solution for our problem. We already started redesigning and reimplementing parts of the multiplayer system and will finish this next sprint.

Problem 2

Description:

In the last weeks we have had some problems with configuring continuous integration on Jenkins. The main problems were that Jenkins could not find the classes from external libraries needed for testing our software system. A second problem we encountered was that some classes that we wanted to test needed resources that were loaded elsewhere in the code at runtime.

Reaction:
We found a way to let Jenkins know where the missing libraries could be found and that it should compile them before using the classes for testing. We made some adjustments to the build.gradle file to achieve this.

Adjustments for the next Sprint Plan

For the next sprint we want to have clearer tasks (i.e. which are not dependent on other tasks and that it is obvious from the description what the exact purpose of the task is). As well we want to assign the tasks in such a way that the tasks that are related to each other are done by/assigned to the same team members to optimize workflow. This happened automatically during this sprint, because it is not feasible to for example let all of the team members work on the multiplayer related tasks.