Sprint Retrospective, Iteration #2

Context Project: Carried Away

Group: MIGI1

User Story	Task	Member responsible for the task	Task Assigned To	Estimated Effort per Task *(in hours)*	Actual Effort per Task *(in hours)*t	Done *(yes / no)*	Priority (A—E) *(A is highest)*	Notes
As a developer of this game, I would like to know about how JMonkey works in relation to our project	Research about JMonkey with the Oculus Rift	Nils	Nils	4.0	2.0	Yes	A	Oculus sdk must be running.
	Connect Oculus Rift to an environment	Nils	Nils	12.0	10.0	No	B	The code doesn't work due to some problems with Maven and the general project setup.
As the lead artist, I would like to create a prototype environment in order for us to work and test in.	Create placeholder environment	Damian	Damian	4.0	10.0	yes	A	Actually took way longer due to the fact that our project isn't a JMonkey project, asset integration doesn't work that well in that case. It does work now though. See problems section below.
	Create platform for the commander to stand on	Damian	Damian	1.0	1.0	yes	B	Trivial once the asset integration was set up.
	Research JMonkey animations	Damian	Damian	4.0	4.0	yes	C	
	Make the platform move at a set speed with a placeholder for the commander on top	Damian	Damian	3.0	2.0	yes	C	Easier than expected mainly because of the same reasons as for creating the platform
As a developer, I would like to be able to access the game through a menu	Create the server main menu	Remi	Remi	3.0	8.0	No	C	The menu exists, but is not functional. A lot of technical difficulties have kept me from being able to work on it and the tutorial I used to build the menu seems to be incomplete.
	Create an Android app main menu	Remi	Remi	4.0	0.0	No	D	
As a developer, I want to create a basic Android app, so that a basic app exist upon which the final Android app can be built	Integrate Android app in project	Marcel	Marcel	2.0	10.0	Yes	A	This was way less trivial than expected.
	Create basic interface	Marcel	Marcel	4.0	0.0	No	A	
	Implement basic gyroscope functionality	Marcel	Marcel	4.0	0.0	No	B	
	Integrate networking in the app	Marcel	Marcel	4.0	0.0	No	B	This task overlapped with the "Basic Networking Implementation" task.
As a developer, I want my deliverables to be done in time	Create final version of the product vision	Remi	All	2.0 pp	Remi - 6.0 Others - 4.0pp	Yes	A	The four hours for everyone was the length of the meeting on wednesday. Remi spent 2 extra hours on the vision in the weekend.
	Create final version of the product planning.	Nils	All	2.0 pp	Luka&Marcel - 3.0pp Others - 2.0 pp	Yes	A	

	Create final version of the game design	Damian	All	2.0 pp	Marcel - 4.0 Damian - ?? Others - 1.0 pp	Yes	A	
As a gamer, I'd like to start a lobby for this game on my computer, so that Android users from my local network can join it.	Research JME3 Networking	Luka	Luka	4.0	4.0	Yes	A	-
	Basic Networking Implementation	Luka	Luka	8.0	6.0	Yes	A	This task was very vaguely defined. The JME3 Networking library contains everything that is needed so much additional implementation wasn't required (creating a ServerWrapper and ClientWrapper class cost around 1 hour). What is included in the actual effort in this task is the automatic detection of servers running on the same LAN, which is required for the two tasks below.
	A Lobby in the server UI	Luka	Luka, Remi	4.0 total	0.0	No	D	-
	Joining a Lobby from the Android UI	Luka	Luka, Remi	4.0 total	0.0	No	E	-

Main Problems Encountered

Problem 1

Description: Most encountered problems when trying to write code, was due to technical issues. For example, there are two different IDEs being used within the group: The Eclipse IDE and the JMonkey SDK. There are some features that one IDE can handle, but not the other. For example, the JMonkey SDK does not react too well with Maven (some team members keep getting errors when trying to reload the Pom), while Eclipse has all sorts of features that handle Maven. Also, there are many JMonkey features that Eclipse does not have, requiring many of our members to work in the JMonkey SDK, all while it complains about Maven. So summarized: Many of the project's dependencies do not work well together.

Reaction: This is not a problem that can be solved easily. Searching on Google for the many encountered problems does not work as they are very specific to our project setup. So the reaction was simple: Asking other groups for help, because maybe they've encountered similar problems and if they don't have an answer, then it's just keep trying out different possible solutions until one works. There isn't really a smarter way of handling situation like these.

Problem 2

Description: Vacation is a problem. As of wednesday (04/05/2016), some members announced that they were going on vacation on that day, which they didn't say at the time of creating this sprint backlog. This resulted in some problems regarding time, and together with **problem 1** it caused a lot of tasks not being finished in time.

Reaction: The reaction was "we need to be prepared for this next time". Some unfinished tasks will have to be moved to next sprint.

Problem 3

Description: Untestable code. One of the problems with the automatic server detection feature on the LAN is that it's nearly impossible to make unit tests about it, let alone an end-to-end test because that would require simulating a LAN.

Reaction: Through the use of the PowerMockito framework, many methods of the ServerFinder and ClientFinder class could be tested. However, because every member of the group agreed that these tests were incredibly complicated, they were not left in the pull request. As an alternative, the code is heavily documented (JavaDoc) and two main methods were added that allow for manual testing.

Problem 4

Description: Asset integration. Because the project wasn't created in JMonkey, the JMonkey SDK doesn't recognize the asset folder. The way JMonkey handles models is by converting for example a blender object to a JMonkey object. The blender object is then split into materials, textures and a mesh. When a JMonkey object is created, the object automatically refers to its material and texture files. However, this only works if the asset structure is recognized by the SDK.

Reaction: A standard way that the wiki described was by using the importing external assets option. This didn't see to do anything though. So a custom file has been created inside the nbproject folder, this is where a JMonkey project would keep its meta-files. Even though our project isn't a JMonkey project, it still uses that information to determine that our assets folder is indeed something JMoney can make use of. Besides that, some code has been added in Environment.java that tells the assetmanager what the root is of the asset folder.

Adjustments for the next Sprint

- Solving technical issues (as described in **problem 1**) as early as possible with multiple team members working on it next to each other. Since this isn't really a task, it won't be mentioned in the Backlog for next sprint.
- Describing tasks more detailed so that confusion like with the "Basic Networking Task" are avoided (not knowning when a task is done).
- More more more coding. This was already mentioned in the retrospective of the previous sprint, however, complications (problem 1) and wrongly estimating the time for documents caused the group to not make good on its promise. Hopefully, since there are no document deadlines next sprint and the project setup problems are solved as soon as possible, next week there will be enough code to be able to present a demo.
- Because next week will be 'code week', the commit messages will need to be more detailed than they have been.