HÁSKÓLINN Í REYKJAVÍK
REYKJAVIK UNIVERSITY

MMOs for Science

Harnessing Massively Multiplayer Gameplay to Speed Scientific Research.

Authors:
Gunnar Þór Stefánsson
Þór Adam Rúnarsson

Supervisor:
Elín Elísabet Torfadóttir

May 15, 2015

Contents

1 Introduction

This report presents details on the game design and prototype implementation of the project *MMOs for Science*, which seeks to boost scientific research by integrating research tasks into EVE Online as a seamless part of the gaming experience. The game design is twofold: On one hand we have a flexible design for a stand-alone mini-game in the theme of EVE Online where you perform research tasks for points, and on the other hand we have various design ideas for integrating the research tasks more deeply into the core gameplay of EVE Online. The report also covers the implementation of a browser-based prototype and the beginning of the real implementation of the mini-game inside the EVE Online client.

2 About the project

Many endeavours in scientific research require substantial amounts of human processing on enormous sets of data. Examples of such processing include annotating images to indicate where anomalies are present (e.g., finding stars in an image of the night sky) or classifying images into one of several categories (e.g., to identify which proteins are present in an image of a cell). Performing such tasks computationally is very difficult, but non-experts can learn to do so proficiently with only minimal, example-based training. This project aims to leverage the human computing power of players in EVE Online by first identifying how scientific data processing tasks can be embedded into the operation of the game, and then implementing a stand-alone prototype that can communicate with servers at MMOS and CCP.

2.1 CCP

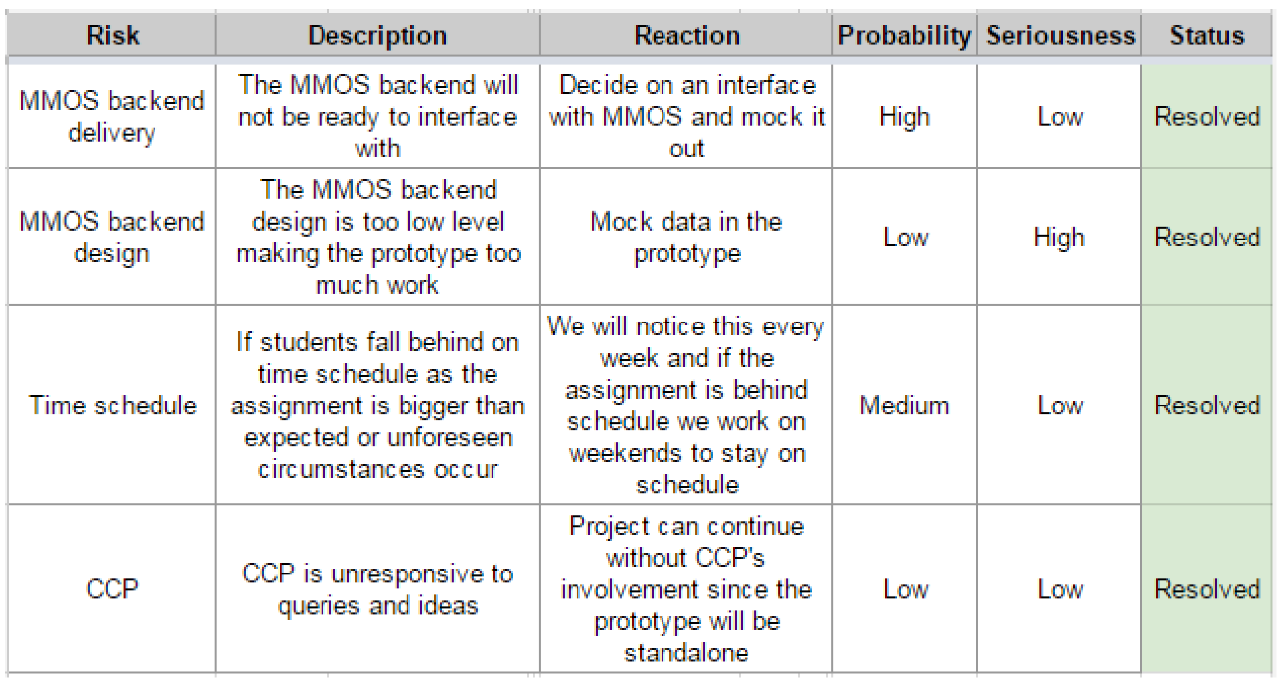
CCP is a leading independent developer of massively multiplayer games, and has been praised for its artistry, game design and unique player-driven, infinitely scalable storytelling narratives. In addition to EVE Online, CCP also develops DUST 514, a groundbreaking, free-to-play, massively multiplayer online first-person shooter for the PlayStation 3, and EVE: Valkyrie, a multiplayer spaceship dogfighting shooter, both set in the EVE Universe. Founded and headquartered in Reykjavik, Iceland, in 1997, CCP is privately held, with additional offices in Atlanta, Newcastle, and Shanghai. For more information, visit `www.ccpgames.com`.

2.2 MMOS

MMOS is a privately held start-up company from Switzerland specializing in the field of citizen science. The company was founded in 2014 by Attila Szantner and Bernard Revaz. Mr. Szantner has a long history in the field of IT, amongst other projects being one of the creators of iwiw.hu (the Hungarian Facebook which at it's height reached more than 4 million users). Mr. Revaz has 15 years of research history in physics (University of Geneva, University of California, EPLF). For more information visit `www.mmos.ch/`.

3 Risk analysis

Risk factors for the project were minimal as the idea could be implemented completely on its own without any involvement from the other stakeholders of the project. The primary risk factor was the responsibility of MMOS to provide an API that the prototype could interface with. This factor could have easily been mocked out as the interface needed was quite simple, mostly consisting of requesting a task and submitting a result. The involvement of CCP was not an important risk factor as their primary role was game design guidance. None of these risk factors were put to the test as the project was enthusiastically supported by both MMOS and CCP.



Risk	Description	Reaction	Probability	Seriousness	Status
MMOS backend delivery	The MMOS backend will not be ready to interface with	Decide on an interface with MMOS and mock it out	High	Low	Resolved
MMOS backend design	The MMOS backend design is too low level making the prototype too much work	Mock data in the prototype	Low	High	Resolved
Time schedule	If students fall behind on time schedule as the assignment is bigger than expected or unforeseen circumstances occur	We will notice this every week and if the assignment is behind schedule we work on weekends to stay on schedule	Medium	Low	Resolved
CCP	CCP is unresponsive to queries and ideas	Project can continue without CCP's involvement since the prototype will be standalone	Low	Low	Resolved

4 Work plan

The team working on the project consisted of two students which both had very similar schedules. Mondays, Wednesdays and Fridays were the primary working days and the weekends were used as needed. All work on the project was done side by side by the students throughout the whole project.

A backlog was created with stories that represented features and tasks that needed to be done and the team estimated points needed to complete each story. That backlog was used throughout the project as the main estimation of how the project progressed and what was left.

Initially the plan was to use `www.trello.com` for prioritizing and keeping track of the tasks, but as the project progressed it was not considered useful as the students were continually aware of what needed to be done through discussion and the backlog gave a better perspective. Using Scrum methodologies for the project was not deemed useful as the project was more about brainstorming and prototyping than working towards a clearly defined product. The conclusion was to have one burndown chart throughout the whole project and inject new stories into the backlog when they had been discussed.

Weekly meetings were held with CCP and MMOS were the project was discussed and all documentation, design discussions and meeting notes were kept on CCP's Confluence system. For time tracking the team used `www.toggl.com`.

4.1 Pre fanfest

The CADIA lab at Reykjavík University was the primary place of work and meetings were held either at CCP or remotely using Microsoft Lync. During the first few months a lot of effort went into quickly getting a working version of the prototype up and running for demonstration at the EVE Online fanfest which was held from March 19th to March 21st.

4.2 Post fanfest

The fanfest was an incredibly important milestone for the project. A presentation was given about the project which was well attended and a roundtable discussion was held afterwards were the prototype was demonstrated. The discussions generated a lot of positive feedback and general excitement from the players of EVE Online and after the fanfest CCP invited the students to join them at the CCP headquarters to work more closely with CCP and implement the mini-game inside the EVE Online client. As the browser-based prototype had really done its job it was abandoned in favor of implementing the first version of the mini-game inside the EVE Online client along with fleshing out the game design.

5 Game design

5.1 Constraints

The main constraints of designing an integration of citizen science research projects into EVE Online were twofold: It could not affect the core gameplay too much, and it had to be flexible with regards to research projects coming and going. The first constraint was a matter of safety for CCP and also a result of the second constraint. If an integration is too tightly coupled with an in-game mechanic it could pose problems if the research project itself was completed or suddenly abandoned.

5.2 Standalone mini-game

In light of these constraints the idea to perform general research in the EVE universe for research points was created. This research module would list the projects that are available to work on from MMOS and would be available at any time from the main user interface so that players could work on problems for example in downtime such as when they are in middle of warp or waiting for their friends. This is flexible with regards to research projects coming and going.

Players who perform this research would receive a new in-game item in the form of research points. How many research points players receive is determined by how well they did with a particular research task. These research points would then be tradable which allows CCP to fully control how the research points will affect the game and making them valuable is open to many ideas, for example:

- When the EVE Online developers release new items, players could donate research points for their faction to have their item released first
- Research points could be used as a resource when manufacturing items from blueprints or developing new blueprints
- Research points could be used for special items in the New Eden Store (a virtual item store inside EVE Online)

Players would also have a personal history of how many research points they have gathered and there would be a scoreboard where players can compare their research proficiency.

5.3 Core gameplay mechanic

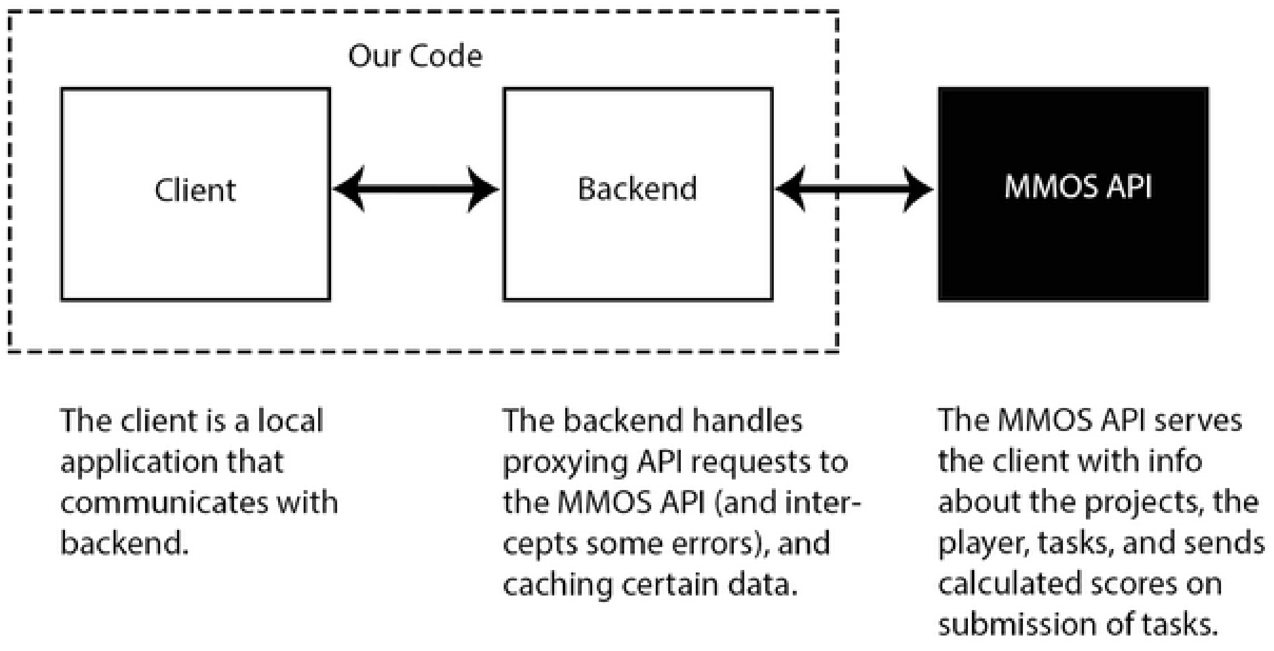
During the roundtable about the project at the fanfest there was considerable interest in making research tasks a part of some core gameplay mechanic. The prototype follows the game design covered above where it is just a window that you can open from anywhere and play for points that could be used for anything. This was deemed more flexible and better serving as a proof of concept. But after this clear go ahead from EVE Online players there will be more work on game design ideas that function more as a core gameplay mechanic. This will also depend on what research projects are available and how they could fit thematically. It is quite possible that both paths will be implemented into EVE Online, where the conservative path fits certain types of research projects and the core gameplay mechanic path fits others.

6 Implementation

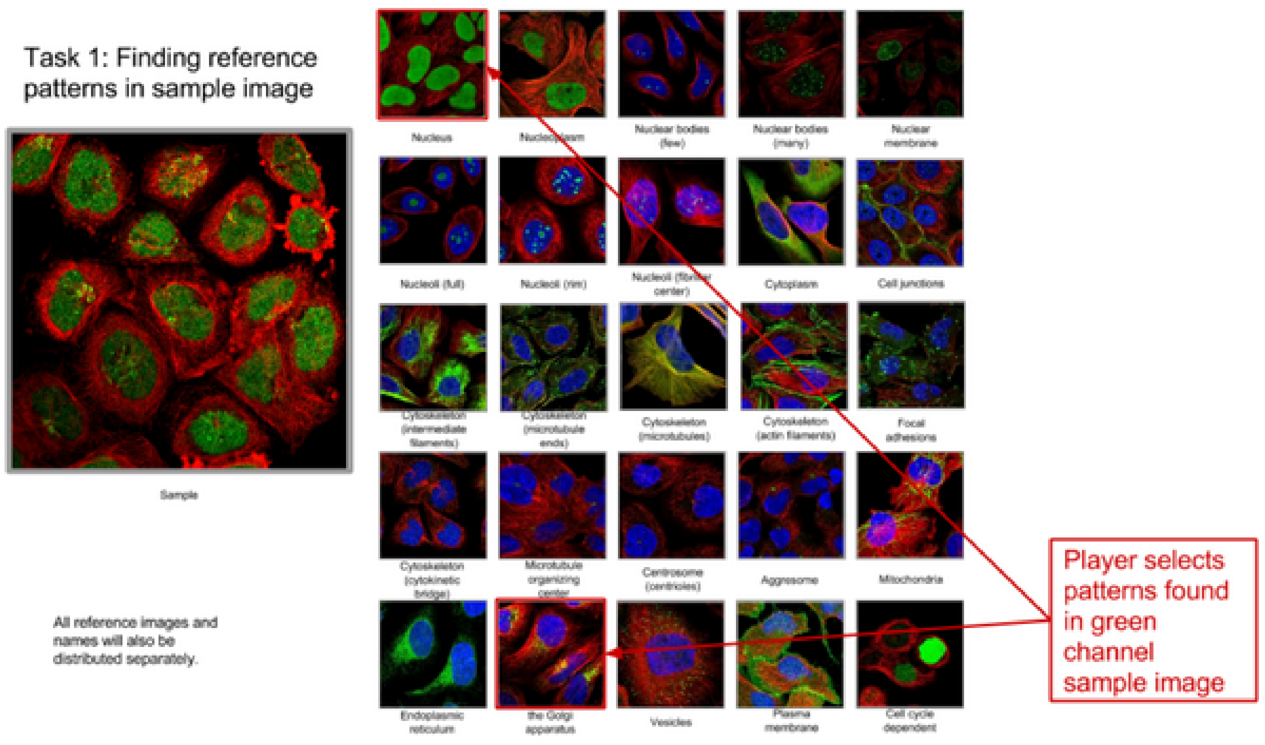
The results of the project (along with the game design report) are the browser-based prototype and also an implementation inside EVE Online.

6.1 Architecture

Both the browser-based prototype and the implementation in EVE Online share the same high-level architecture. The client talks to a backend which acts as a proxy and a buffer for the the MMOS API.

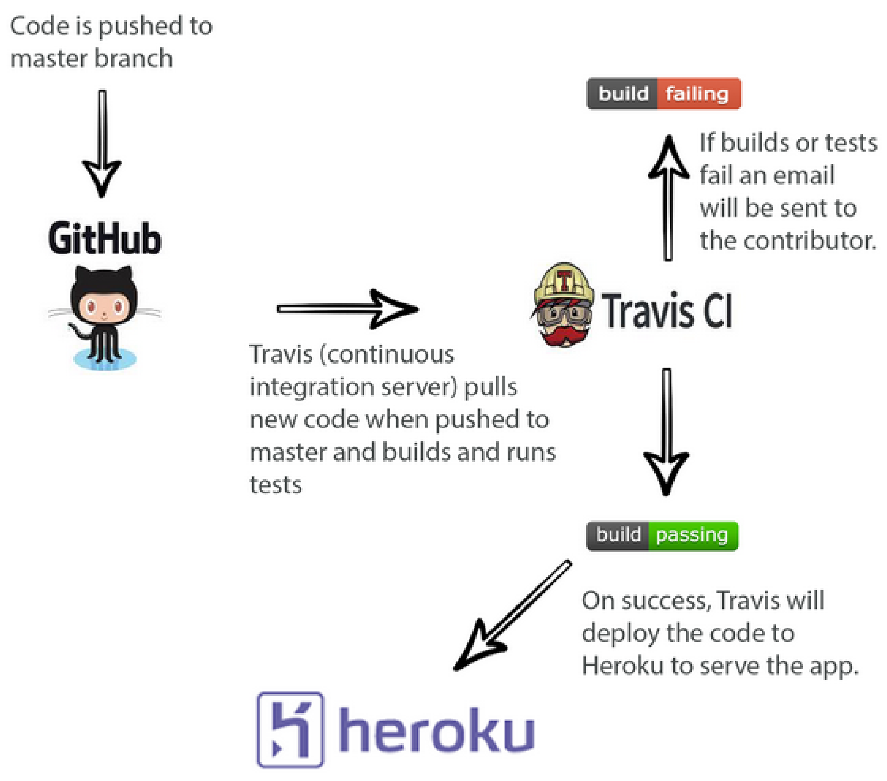

Our Code
Client
Backend
MMOS API
The client is a local application that communicates with backend.
The backend handles proxying API requests to the MMOS API (and intercepts some errors), and caching certain data.
The MMOS API serves the client with info about the projects, the player, tasks, and sends calculated scores on submission of tasks.

The research project the implementations were designed for is the Subcellular Atlas from the Human Protein Atlas in Sweden (`www.proteinatlas.org/subcellular`) where the task is to classify images into one or more of 25 categories. More details about this can be found in the game design report.


Task 1: Finding reference patterns in sample image
Sample
All reference images and names will also be distributed separately.
Nucleus
Nucleoplasm
Nuclear bodies (few)
Nuclear bodies (many)
Nuclear membrane
Nucleoli (full)
Nucleoli (rim)
Nucleoli (fibrillar center)
Cytoplasm
Cell junctions
Cytoskeleton (intermediate filaments)
Cytoskeleton (microtubule ends)
Cytoskeleton (microtubules)
Cytoskeleton (actin filaments)
Focal adhesions
Cytoskeleton (cytokinetic bridge)
Microtubule organizing center
Centrosome (centrioles)
Aggresome
Mitochondria
Endoplasmic reticulum
the Golgi apparatus
Vesicles
Plasma membrane
Cell cycle dependent
Player selects patterns found in green channel sample image

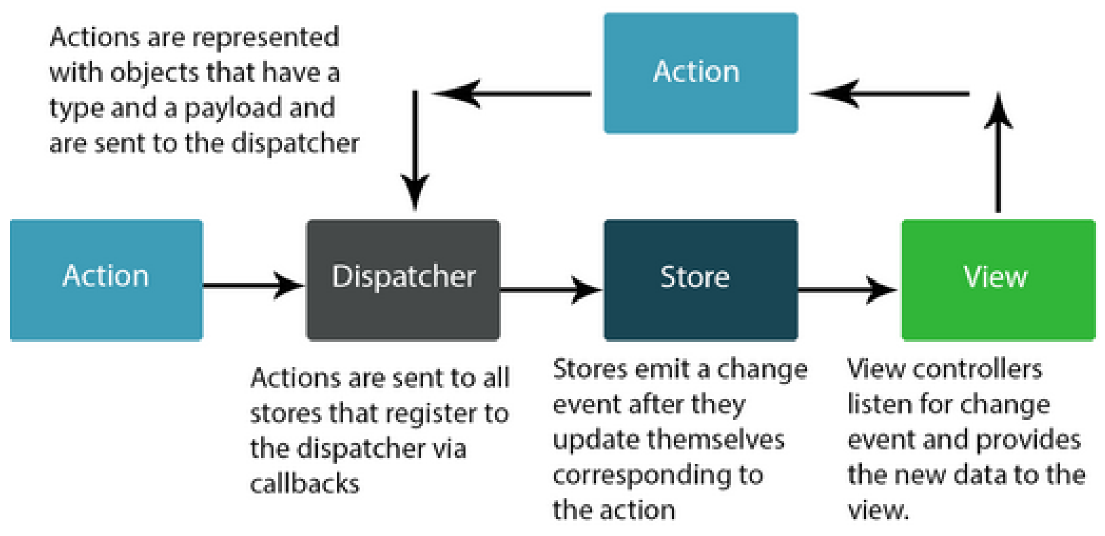
6.1.1 Browser-based prototype

The development of the browser-based prototype used Git and GitHub for version control and all new code was continuously integrated and deployed to a test server.


Code is pushed to master branch
GitHub
Travis (continuous integration server) pulls new code when pushed to master and builds and runs tests
Travis CI
build failing
If builds or tests fail an email will be sent to the contributor.
build passing
On success, Travis will deploy the code to Heroku to serve the app.
heroku

The client architecture is predominantly based on Flux (`http://facebook.github.io/flux`). The main components are:

- **Actions** : Actions represent anything that happens in the system, such as the user submitting a solution or a new task being received from the server. They include an action type (e.g. NEW_TASK_RECEIVED) and a payload (e.g. the task data).
- **Dispatcher** : A singleton item that dispatches actions to the stores that have subscribed to them.
- **Stores** : Stores manage state and data, and they typically subscribe to the dispatcher to update themselves and then emit change events when they have done so. If Image is a model in the application, an ImageStore would manage all instances of that model and any necessary state, for example a variable that points to the currently selected image.
- **Views** : The UI of the application. They subscribe to change events from stores to re-render themselves. They should be relatively pure, maintaining as little state as possible, essentially a function from some data to a rendered view. React (`http://facebook.github.io/react`) was used to render views.


Actions are represented with objects that have a type and a payload and are sent to the dispatcher
Action
Action
Dispatcher
Store
View
Actions are sent to all stores that register to the dispatcher via callbacks
Stores emit a change event after they update themselves corresponding to the action
View controllers listen for change event and provides the new data to the view.

The benefit of this architecture is the easy with which you can reason about the application and extend it. Actions and data propagate through it in a single direction, and given any action the application fully updates itself through stores and views before it handles another action.

The continuous integration, the component-based client architecture and the familiar environment of the browser all contributed to an extremely fast pace of development which was instrumental in shipping a working version of the prototype before the fanfest.

6.1.2 EVE Online implementation

The implementation inside EVE Online closely mimics the browser-based prototype. This is possible because the EVE Online codebase has an event system where UI elements can listen to events from anywhere and react accordingly, which allows the UI elements to follow the design of the view components from the prototype.

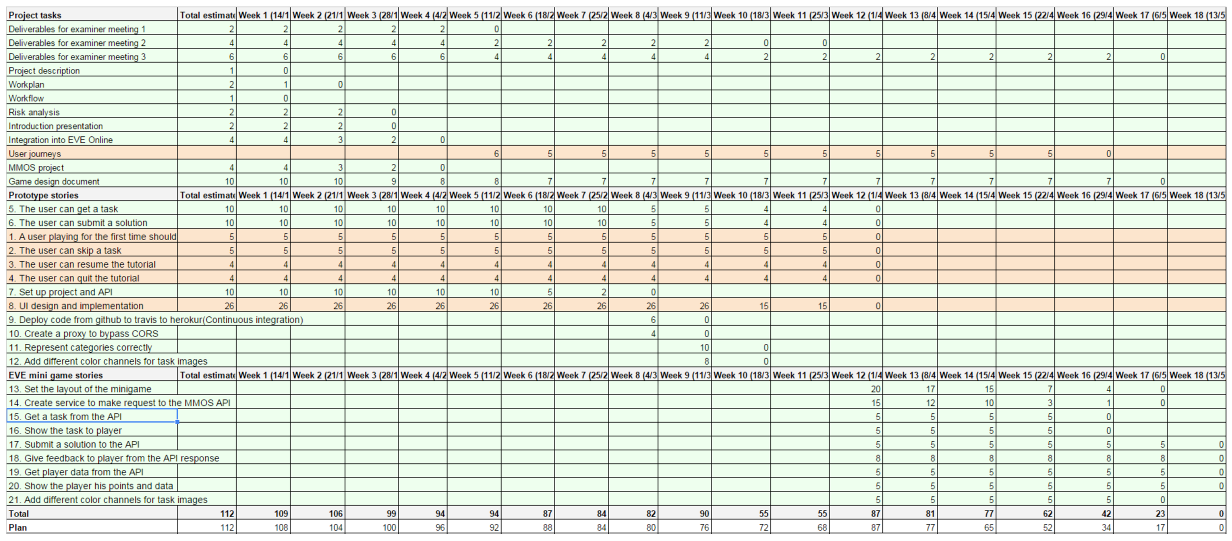
The game is all included in one Python package in the main package directory of the EVE Online codebase. This package is split up into two modules, `client` and `server`. The client package can be called to initiate the game and the server package contains a RPC service that acts as a proxy and a buffer to the MMOS API that the client can call.

The primary difference from the prototype is that the EVE Online implementation is designed to support more research projects within the main window. The main window class initiates an instance of the Project class with the ID of the selected project, and the Project class instance handles delegating the rendering of that particular project to the appropriate code (could be different for each project) and requesting tasks and submitting results to the appropriate channel for the project selected.

Along with the development of the EVE Online implementation considerable work went into figuring out how to quickly explain the task in the Subcellular Atlas research project to players. This involved learning what the images represent, simplifying the category descriptions down to simple visual language and meeting with the scientists of the project to review the material.

7 Backlog and burndown

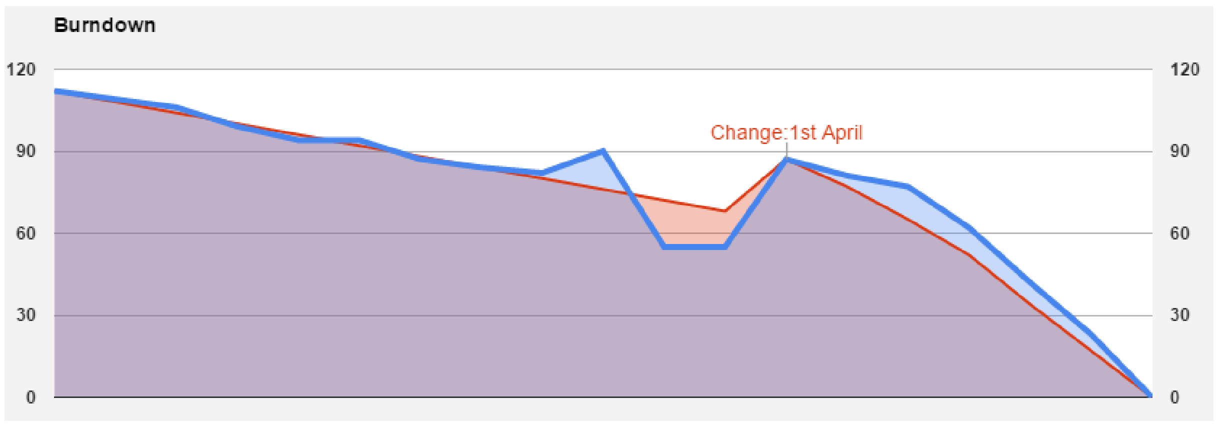
For the backlog a Google Sheets spreadsheet was used where stories were estimated and then reviewed the end of every week. New stories often needed to be added to the backlog and in these cases the story was discussed, estimated and added to the backlog at the point where it was needed. When the team was invited to create the implementation inside EVE Online then the estimation plan was reviewed and changed.



Project tasks	Total estimat	Week 1 (14/1	Week 2 (21/1	Week 3 (28/1	Week 4 (4/2	Week 5 (11/2	Week 6 (18/2	Week 7 (25/2	Week 8 (4/3	Week 9 (11/3	Week 10 (18/3	Week 11 (25/3	Week 12 (1/4	Week 13 (8/4	Week 14 (15/4	Week 15 (22/4	Week 16 (29/4	Week 17 (6/5	Week 18 (13/5
Deliverables for examiner meeting 1	2	2	2	2	2	0													
Deliverables for examiner meeting 2	4	4	4	4	4	2	2	2	2	2	0	0							
Deliverables for examiner meeting 3	6	6	6	6	6	4	4	4	4	4	2	2	2	2	2	2	2	0	
Project description	1	0																	
Workplan	2	1	0																
Workflow	1	0																	
Risk analysis	2	2	2	0															
Introduction presentation	2	2	2	0															
Integration into EVE Online	4	4	3	2	0														
User journeys						6	5	5	5	5	5	5	5	5	5	5	0		
MMOS project	4	4	3	2	0														
Game design document	10	10	10	9	8	8	7	7	7	7	7	7	7	7	7	7	7	0	
Prototype stories	**Total estimat**	**Week 1 (14/1**	**Week 2 (21/1**	**Week 3 (28/1**	**Week 4 (4/2**	**Week 5 (11/2**	**Week 6 (18/2**	**Week 7 (25/2**	**Week 8 (4/3**	**Week 9 (11/3**	**Week 10 (18/3**	**Week 11 (25/3**	**Week 12 (1/4**	**Week 13 (8/4**	**Week 14 (15/4**	**Week 15 (22/4**	**Week 16 (29/4**	**Week 17 (6/5**	**Week 18 (13/5**
5. The user can get a task	10	10	10	10	10	10	10	10	5	5	4	4	0						
6. The user can submit a solution	10	10	10	10	10	10	10	10	5	5	4	4	0						
1. A user playing for the first time should	5	5	5	5	5	5	5	5	5	5	5	5	0						
2. The user can skip a task	5	5	5	5	5	5	5	5	5	5	5	5	0						
3. The user can resume the tutorial	4	4	4	4	4	4	4	4	4	4	4	4	0						
4. The user can quit the tutorial	4	4	4	4	4	4	4	4	4	4	4	4	0						
7. Set up project and API	10	10	10	10	10	10	5	2	0										
8. UI design and implementation	26	26	26	26	26	26	26	26	26	26	15	15	0						
9. Deploy code from github to travis to herokur(Continuous integration)									6	0									
10. Create a proxy to bypass CORS									4	0									
11. Represent categories correctly										10	0								
12. Add different color channels for task images										8	0								
EVE mini game stories	**Total estimat**	**Week 1 (14/1**	**Week 2 (21/1**	**Week 3 (28/1**	**Week 4 (4/2**	**Week 5 (11/2**	**Week 6 (18/2**	**Week 7 (25/2**	**Week 8 (4/3**	**Week 9 (11/3**	**Week 10 (18/3**	**Week 11 (25/3**	**Week 12 (1/4**	**Week 13 (8/4**	**Week 14 (15/4**	**Week 15 (22/4**	**Week 16 (29/4**	**Week 17 (6/5**	**Week 18 (13/5**
13. Set the layout of the minigame													20	17	15	7	4	0	
14. Create service to make request to the MMOS API													15	12	10	3	1	0	
15. Get a task from the API													5	5	5	5	0		
16. Show the task to player													5	5	5	5	0		
17. Submit a solution to the API													5	5	5	5	5	5	0
18. Give feedback to player from the API response													8	8	8	8	8	8	0
19. Get player data from the API													5	5	5	5	5	5	0
20. Show the player his points and data													5	5	5	5	5	5	0
21. Add different color channels for task images													5	5	5	5	5	0	
Total	**112**	**109**	**106**	**99**	**94**	**94**	**87**	**84**	**82**	**90**	**55**	**55**	**87**	**81**	**77**	**62**	**42**	**23**	**0**
Plan	112	108	104	100	96	92	88	84	80	76	72	68	87	77	65	52	34	17	0

The stories marked with green are finished stories while the ones marked with red are unfinished stories. All the red stories are related to the browser-based prototype and were discarded when the work on the EVE Online implementation began.

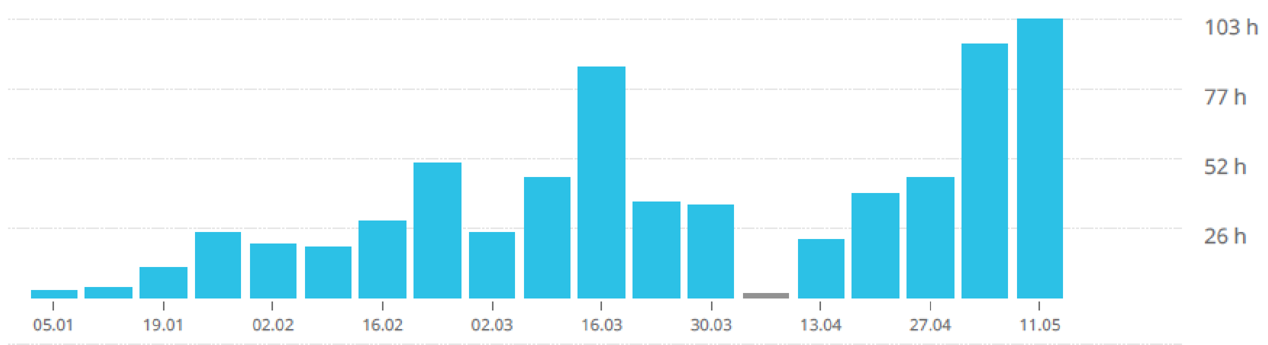
The burndown chart below is from the backlog where the red line represents the estimated plan and the blue line represents the actual progress. Notice that April 1st marks the point where the prototype was discarded and work on the EVE Online implementation started.



8 Time schedule

2015-01-05 - 2015-05-16

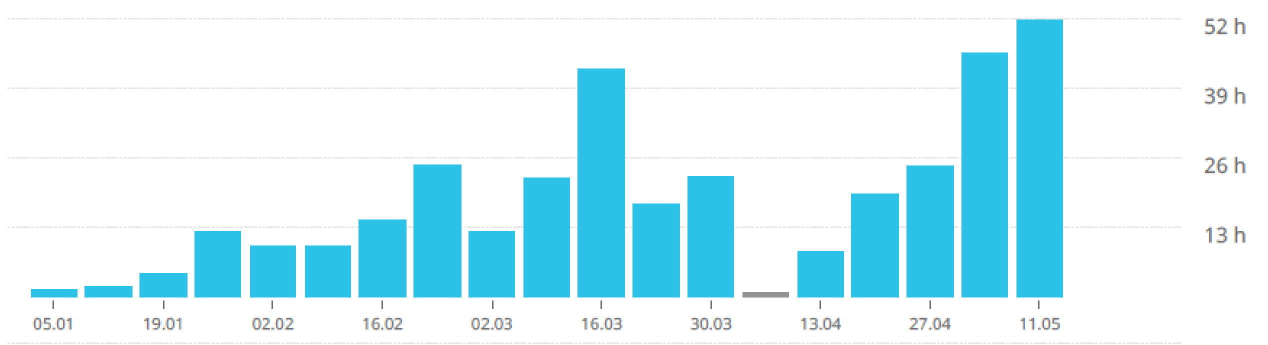
Total 674 h 54 min



2015-01-05 - 2015-05-16

Total 339 h 28 min

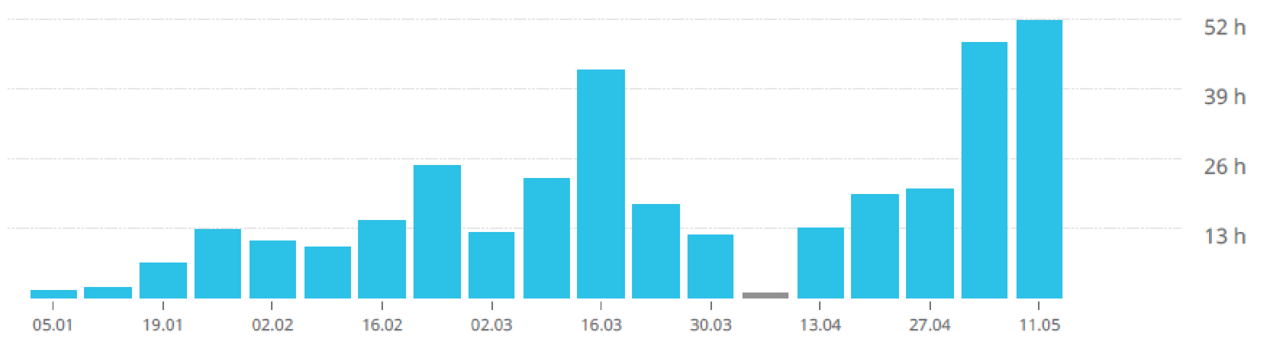
Thor Adam selected as users



2015-01-05 - 2015-05-16

Total 335 h 26 min

Gunnarthors selected as users



9 The future

The project has received a grant from the Rannís Student Innovation Fund to fund three months of development in the summer of 2015. These three months will be used to continue development of the EVE Online implementation with the aim to have it ready for player testing in the end of August, and hopefully to have it deployed to the main server soon after that.

In general the future of citizen science in games is extremely exciting, but also full of challenges. The products of this project are heavily tailored to a specific research problem, but in the future these problems could be generalized to allow MMOS to serve problems from many different research projects into one game, or allow many different games to interpret problems in different ways.

10 Words from owners

10.1 MMOS

During the last semester MMOS Sarl was working together with CCP and Reykjavik University to create a prototype of injecting citizen science problems into EVE Online. From the Reykjavik University Gunnar Þór Stefánsson and Þór Adam Rúnarsson were participating in the project. Gunnar and Þór proved to be a very good partner for us in this project. The design and implementation of their part was created on time and without any problems with an excellent quality. But apart from the implementational parts they also excelled at project management, participating in the creative process and communication tasks when presenting the project at the Fanfest 2015 presentation. We are very happy that we can continue working with them supported by a grant from Rannís Student Innovation Fund.

Attila Szantner

10.2 CCP

When we were put in contact with Gunnar and Þór regarding the citizen science project, I was quickly filled with confidence about the future of this experiment. As proof of concept they were quickly able to deliver a working browser prototype. Then in a matter of days they brought a high quality polished version of it to show to our players at the EVE FanFest where we held a presentation and roundtable on the subject. That prototype played a key role in getting buy-in and excitement from the community about the project which is crucial for us to move forward on something like this.

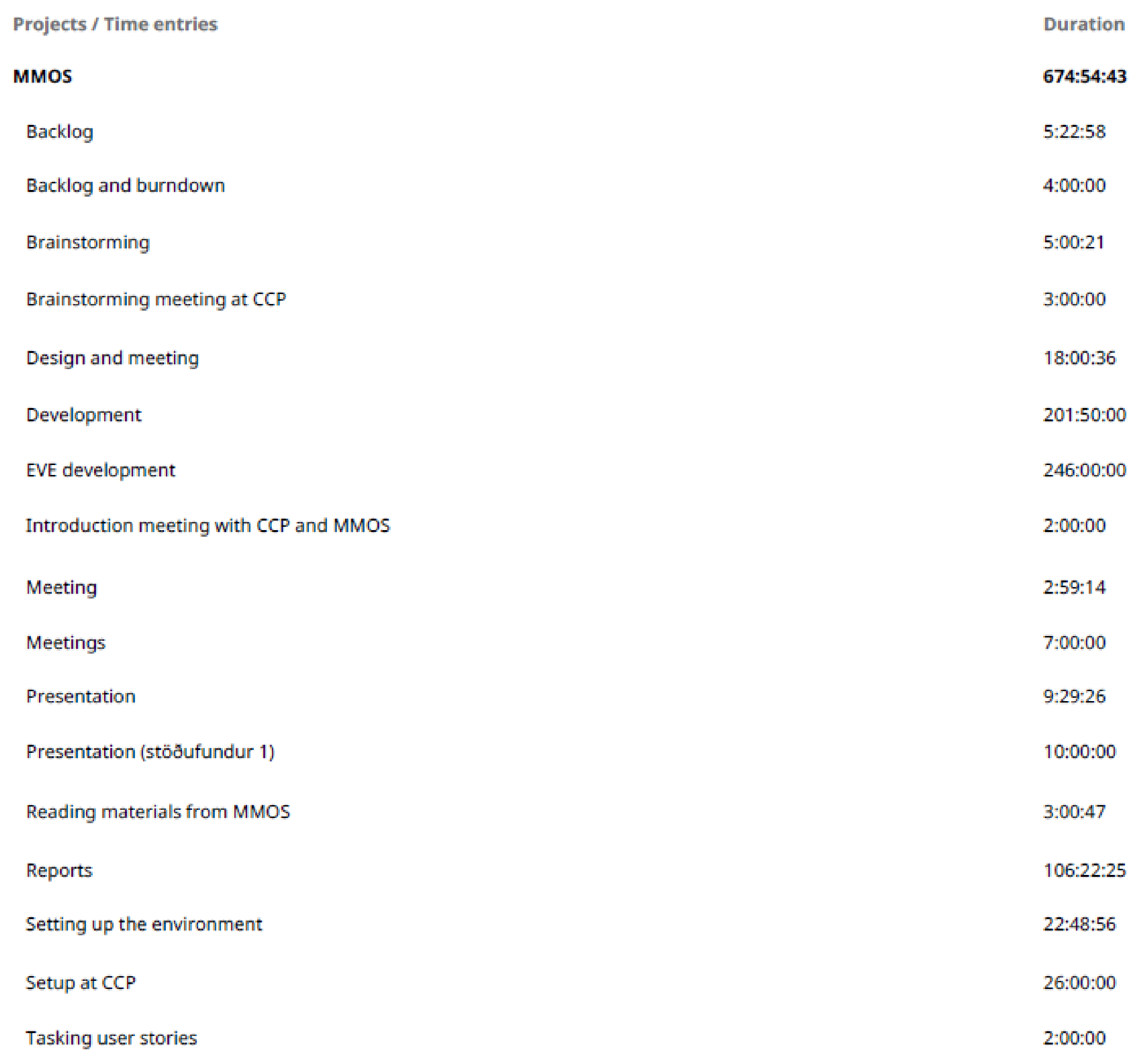
Pétur Örn Þórarinsson

11 Final words

Working on this project has been really inspiring and also a huge learning experience. During the planning and brainstorming it was interesting to see how a company like CCP works on project like these, and getting into the EVE Online codebase was extremely interesting as it is very large and spans more than a decade of development. The team would like to thank CCP and MMOS for their enthusiasm and welcoming support.

Starting with a browser-based prototype and building on existing libraries for fast iteration proved to be a very good decision and was instrumental in getting the prototype ready for fanfest in time, which was very effective in getting player support and feedback. Working side by side for the entire project and having regular meetings with CCP and MMOS was also very helpful as it kept the project constantly on rails and everyone involved was aware of how it stood and where it was going. And finally the fanfest was something that really stood out as an experience and getting the chance to show the work to future users was very inspiring.

12 Appendix



Projects / Time entries	Duration
MMOS	**674:54:43**
Backlog	5:22:58
Backlog and burndown	4:00:00
Brainstorming	5:00:21
Brainstorming meeting at CCP	3:00:00
Design and meeting	18:00:36
Development	201:50:00
EVE development	246:00:00
Introduction meeting with CCP and MMOS	2:00:00
Meeting	2:59:14
Meetings	7:00:00
Presentation	9:29:26
Presentation (stöðufundur 1)	10:00:00
Reading materials from MMOS	3:00:47
Reports	106:22:25
Setting up the environment	22:48:56
Setup at CCP	26:00:00
Tasking user stories	2:00:00