UCL Peach Group 39: Reality
Bi-weekly Report #6

Timur Kuzhagaliyev
Fraser Savage
Laura Foody

27th January 2017

1 Overview of the last 2 weeks

Over the last 2 weeks we held 3 meetings on 17/01, 19/01 and 27/01. Below you can find the overview of topics discussed and decisions made during said meetings.

- As was mentioned in the previous report, we've been using Atlassian JIRA to manage our project. In the beginning of Term 2 we've switched from Kanban to Scrum and are now working in 2-week sprints, reporting progress to our client in the end of every sprint and establishing the list of tasks for the next week on our lab session every Friday.
- We've begun merging the code of our React-based web app and the Java API in a single repository to make maintenance easier.
- Our client suggested we come up with contents for a short Medium article which would serve as an overview of our project and progress we've made so far. At this point in time the article has already been published and can be found here.
- Lorenz Berger from the Translational Imaging Group (TIG) has agreed to provide us with Docker image of a neural network that could create bitmaps from CT scans hence making it possible to generate 3D models. Unfortunately, due to a recent change in the tools he was using for development that will no longer be possible, at least not any time soon.
- We've decided to research how we could integrate our application into the existing the NHS infrastructure, namely how we can communicate picture archiving and communication system (PACS) with to fetch DICOM images and how our system can fit into N3 network.
- Thanks to our client we might have a chance to have a short meeting with a number of urinal surgeons that can give us feedback and useful insight into the potential applications of our system, as well as take a short survey to help us better understand the needs of our users and confirm that we're going in the right direction.
- We decided to look into licensing our code and patenting the seamless pipeline we're developing to what options we have and what will be the most feasible strategy.
- After discussing it with our client, we agreed to stick to soft-deleting in our application, to make sure sensitive medical data can be recovered if necessary.
- We also agreed that our file format should support files with arbitrary extensions that will not necessarily be displayed on HoloLens but are an important part of a holographic patient case.
- We've established the list of improvements we're planning to make to our HoloLens application, including better animations, tooltips for buttons, better model appearance.

- We've contacted Team 36 about potential collaboration and integration of their tokenised role-based authorisation system in our application, but sadly their solution is not yet ready for production. Additionally, we met the masters students responsible for the development of the PEACH backend to discuss the dependencies of our system and how they can be integrated into said backend. We have agreed that file system, database, authorisation and PACS access are the bare minimum required for our application to function as intended within the PEACH infrastructure.

2 Tasks completed

- Prepared a Medium article about our project and future plans
- Held a short elevator pitch about the current state of our system, the challenges we encountered and further plan of actions
- Established the user representation in our system and designed the concept for a holographic patient case
- Setup continuous integration for the backend API and begun designing a formal API specification based on experiments conducted during Term 1
- Merged code base for webapp and backend API in a single repository

In the last 2 weeks we mostly focused on research, evaluation of existing system and planning of further actions, so we haven't produced that much code. Now that we're finished our first 2-week sprint and most of the planning is complete, we plan to begin refactoring our existing code and adding missing features.

3 Problems encountered

- Lorenz from TIG has experienced some issues with his existing trained models so it will most likely take longer than we expected to implement in our system, hence we'll have to find a way to mock this functionality.
- Making our system scalable involves a lot of research and might require a vast amount of tweaks and additions to be made to the source code, which might not be possible during the time span we have for development. We should decide which features are the most feasible and abandon the other ones due to time constraints.
- The development of the web app is progressing very slowly due to lack of technical expertise of team members assigned to its development, so we will have to adjust our team structure and take necessary action to speed up the development since the web app is the core of our application.

4 Plans for the next 2 weeks

- Prepare the first draft of the formal API specification using swagger.io
- Improve the performance and the visuals in the HoloLens application
- Prepare a simple prototype for the web app, to begin the iterative development process
- Produce a short paper describing the progress made in the development of our system, current architecture, challenges encountered and future plans
- Prepare questions and demos for the urinal surgeon meeting
- Research suitable licenses for our code and feasibility of patenting the concept

5 Individual reports

5.1 Timur Kuzhagaliyev

I've spent the last 2 weeks doing research, planning out the schedule for development in Term 2 and organising various meetings to discuss collaboration in the development of our system. My research was mostly focused on the development of the file format for holographic patient cases and improving performance and maintainability of HoloLens applications.

For the former, I've read about general file format development, existing file formats used for medical data and various strategies for designing file formats that serve as a collection of files with some meta data attached to them. At the moment I'm leaning towards using an archive (e.g. ZIP) containing all files relevant to the holographic patient case and a JSON file that holds all of the meta data for the case and strictly follows the JSON schema we will define. I've produced several drafts of the specifications for a holographic patient case and the user representation in our system which we'll use later when finalising said specifications.

As for HoloLens application development, I've looked into a number of ways of improving the performance of our application without losing (and hopefully even improving) texture and model quality. I've also researched automated testing for HoloLens applications developed in Unity. Sadly, Unity Cloud Build platform doesn't support building for HoloLens yet and there don't seem to be any other feasible solutions for the full continuous integration pipeline for Unity/HoloLens development at this point in time. That said, it is still possible to run unit tests using the Unity Testing Tools available in the Asset Store, and an instance of Unity can be run in headless mode to execute the tests automatically, potentially using NUnit.

Finally, I've organised and attended a number of meetings, including team meetings where we discussed the plans for development and decided on tasks for our sprints, meetings with our client where we got feedback about our application and could voice any issues we had, meeting with other PEACH developers to discuss potential collaboration and, last but not least, a meeting with Lorenz Berger who updated us on the current state of his system and agreed to give us access to the source code so we can implement it in our project.

5.2 Fraser Savage

Since the start of this term, my focus has been on setting the foundation for a more structured development approach with the API. As outlined on our project website from December, this was to take the form of test-driven development (TDD). In pursuit of this I have set up the Continuous Integration (CI) pipeline on the GitLab repository, using a docker image with Maven and JDK 7 as the job runner. Every time a change is pushed to a branch which contains a .gitlab-ci.yml file, the CI pipelines will trigger and start to run builds and tests against the commit, alerting team members of the result. This may be changed to run on pull requests only in the future, depending on how it pans out.

Further down the line, it is likely that I will set up a deployment stage in the CI process to allow for Continuous Delivery (CD) of the application as features and functionality pass their tests and are brought into the master branch of the repository.

As part of the scope for a structure development approach, I have incorporated the swagger-core library into the application and will be documenting the API as I forge onwards. The purpose of this is to reduce the time needed to document the API for our team, as well as to improve the experience of the developer consuming the API by providing good quality generated documentation. Swagger takes the documentation @annotations in the source, and using the source generates a API specification which can be viewed in a doc browser such as Swagger UI. We've decided to take this approach because if another team has to pick up our project we want to avoid the situation where the code is poorly documented and has to be discarded.

In addition to what is listed prior, I have started to write some of the unit tests for application components so that I can start adding core functionality to it. As it currently stands, user authentication and listing of Holographic Patient Cases will be the first two components added.

Finally, I made some small changes to the way that the currently deployed initial prototype works, changing authentication to take encoded form parameters instead of JSON. This was done so that Laura could authenticate any the API prototype in a standard manner.

5.3 Laura Foody

My goal for this two week sprint was to implement user authentication for the user interface (webapp). With the help of Tim I set up a new directory for the webapp which used Fraser's back-end server. The directory used gulp dependencies to automatically re-deploy everytime something was changed in a file. I then needed to find a way to transpile my es6 code into es5 which is the language that browsers understand. In previous experiments I had been using webpack to build my directories and it configures the transpiling methods for you. However webpack also creates server.js files and middleware which I do not need in this current version of the webapp so I therefore had to look at other ways to do it.

I did a few searches and discovered that the babel dependency was the most common way to transpile jsx into js so I tried to set it up in my gulpfile but it didn't work. I then did some further research and discovered the browserify dependency which can be used alongside gulp and babel to transpile code; but I still had no luck compiling the code. Over the course of the last two weeks I have exhausted every online tutorial, and stack overflow post related to the issue I am experiencing but have had no luck finding a method that works. So therefore I have been unable to implement the user authentication task. My plan is to see if I can find a way to potentially use webpack, otherwise I will just have to keep trying different configurations of babel, browserify, and other related dependencies.