# Week Four Milestone Report

Capstone Project

*159.356*

*9th August 2015*

*Aidan Houlihan, Ashraf Alharbi, David Dudson, Josh Baker & Mitchell Osborne*

## Table of Contents:

## Brief:

To create a web based integrated development environment which facilitates the teaching of code to children between the ages of 10 - 14 years old. They should be taught the basic concepts of coding through gamification using a visual programming language.

## Specification:

The environment must be built on Massey’s SoGaCo framework.
The user interface must be designed for children aged 10 -14 years old.
The game should run in a web browser on desktop and mobile.
A visual language must be used.
The environment must be easily usable by children aged 10 - 14 years old.
The environment must stimulate the children to learn coding.
The underlying game coding language should use Java code and may later be extended to Python code.
The overall project should abide by all pursuant laws and have correct licensing and copyrights for code used.
The game used will be a variation from the popular ‘BattleShips’ game.

***Not within the scope:***
Creating the framework for the integrated development environment to work on.
Creating a server for the framework to be built on.
Implementing a login system.
Implementing security features of the framework.
Creating a mobile application for the integrated learning environment.

## Project Infrastructure:

The project has been divided into four major sections; user interface design, visual language implementation, gamestate design and integration, build and deployment. This was done to break the project down so that members of the team could focus on a single part of the larger problem. The project will be monitored by a team leader keep an overall focus on the project and how we are tracking. For this purpose we have decided to use Google Spreadsheets as it provides a neutral platform in which all of us can easily access without the need to install proprietary software. It has built in sharing functionality which enables other users to comment and edit as necessary. A comparison chart further discussing the reasons we chose these technologies over others can also be found at the bottom of this document here. The conclusion we came to was to use a free, familiar and simple technology as we are a group of student colleagues who have no monetary resources to complete only a single project. We decided that Google Sheets combined with Google Groups would provide all of the necessary features in an easily accessible platform with no unnecessary features, hassle-free installation and familiarity of the product.

To manage the constant flow of information, ideas, communication and changes to the project we have set up a Google Group which acts as a forum for discussion of issues relating to the project. This forum allows for anyone to voice their concerns and ideas without being in person which can be daunting. To further aid in discussion we have a Facebook chat conversation which helps us remind each other of important deadlines, give quick updates on where we are currently at with our code and general information. This is where a lot of the informal ideas and communication occur.
For coding related issues we have set up a git repository which contains a clone of the SoGaCo framework so that we may be able to deploy, test and change the code without affecting the current framework. We chose Github as our repository as it provides us with an easy collaborative communication platform allowing us to review changes, comment on lines of code, report issues, and plan the future of our project with discussion tools.

To keep track of what each member is working on for any given week we have a timeline planning document which displays information on each members goals and whether or not they completed last weeks task. Milestones are also marked out at weeks 4, 8 and 12.The Timeline Planning can be found in the appendices section here.

## Life Cycle Model and Project Plan:

The life cycle model is an important process to be decided on as it determines the overall course to be followed for the development of the software. As such we decided on our life cycle model by discussing among ourselves and putting it to a vote. We settled upon the ‘Iterative model’ scrum. This is because it provides a collaborative, agile approach to software development and enables us to self organise by encouraging face to face and online communication. A key feature of scrum is that it recognises that the requirements for the project can change and that unpredicted challenges can be more easily addressed. We decided upon this model during week four through a series of face to face discussions. The main reason for choosing scrum was its use of these discussions have been summarised in the comparison chart found in the appendices section here.

## Roles Specification:

These roles were determined collectively by the group in week one of the project.
Each member will work on their respective sections of the project individually, however they will communicate with one another regarding certain aspects which may overlap between their sections. As we are a team of five with only 4 modules, we have an additional coder to help ensure that the merging of any two sections are done effectively and in parallel. This will primarily be done by the team leader.

**Josh Baker:**
*Leader / Coordinator*
Ensure work gets done
Communication with Jens
Write weekly reports
Document all communication and milestone stages
Floater coder helps anyone who needs it

**Mitchell Osborne:**
*Visual language implementation*
Researches the implementation of turning the visual language to Java
Responsible for the API that changes the coding blocks into usable Java code
Will communicate with David as to integrate his part with the front end and Ashraf for the UI

**Aidan Houlihan:**
*Game State*
Creates the gamestate and the graphical user interface for the environment
Responsible for the gamestate and the ships and the grid (Partly UI)
Responsible for integrating the bots into the gamestate
Will communicate with David as to integrate his part with the front end

**Ashraf Alharbi:**
*User Interface*
Creating the user interface using HTML, CSS, JSP and JS
Responsible for creating the build environment
Responsible for choosing the theme and how the product will look
Will communicate with Aidan and Mitchell to ensure consistency and seamless integration

**David Dudson:**

*Integration, Build and Deployment:*

Manage all build scripts.

Manage Git repository + pull requires.

Manage QA (Code coverage, testing etc.)

Manage CI/Deployment
Makes sure everything is Integrated and working at all stage.

Communicates with everyone for git pushes to master.

## Technology selection:

For the user interface, we decided on a number of front end technologies, Java Server Pages, Angular JavaScript, HTML and CSS. All of these technologies provide a wide range of attributes and qualities which will provide a solid user experience. SoGaCo is built primarily with JSP's so the use of them in our project is essential to keeping with consistency of design. HTML and CSS are both the cornerstones of web pages and provide a the foundation in which to build our project on. AngularJS provides the animations and pop ups necessary for the game to be entertaining and to notify the user if something goes wrong. For the backend gamestate and block creation, we have used Blockly, Java Server Pages and Java. As this project specifies the use of a visual language, we have decided to use Blockly because it has native web integration and a built in compiler. Both of these things will prove incredibly helpful with the integration of the visual language into our project. An API called EasyJ will supply the basis of the change from the visual language into usable Java code. More information on these technologies can be found in the form of comparison chart in the appendices section here.

## Architecture / High level design:

From week one, we broke the project into four major constituent parts; gamestate, user interface, visual language and front end integration. Each on of these parts were then assigned to the member who was the most knowledgeable and had the most skill in. Each member needs to be aware of how their module fits into the overall project and who they should integrate their modules with. Further information in the form of a UML diagram on the modules and their interactions can be found in the appendices section here.

## Risk Management:

The following are risks which can be foreseen and managed in an appropriate manner. There will be some unforeseen risks in which we will have to act accordingly as a team and with our best judgement.

**Member leaving the group:**

When a member unexpectedly leaves the group or paper.

-Each remaining member will have to take over a part of the leaving members work. We will allocate the new workload to each member according to how busy they would be at the point in time and to which their skills lie.

**Code loss:**

Where any code for the project goes missing or is lost.

-The member responsible for the loss of code will have to redo the work lost, to prevent code loss we will keep regular backups of code and have a central repository of code where code can be kept.

**Feature ballooning:**

Where the requirements for the project continually expand and the features of the product become too extravagant.

- Can be managed by setting strict guidelines to our requirements and ensuring that we finish (or come close to finishing) a feature before starting another.

**Specification Breakdown:**

Where after a time of coding and integration of the product result in the specification becoming insufficient or incomplete to work to.

-This risk will be mitigated by having the project manager make executive decisions when required and or communicate within the group to redefine the specifications by impersonating the stakeholder.

**Poor Productivity:**

Working within a group always poses the risks of some members not completing the work to the specified deadline.

-This risk will be mitigated by using our chosen scrum method of software development and its short iterations of development. When setting a deadline, people often start work close to the deadline rather than further ahead of time, by having short iterations of development work is timeboxed into manageable sections and thus increasing productivity.

**Technology Failure:**

As we are using so many different technologies and softwares as services there always is the risk that their services go down for an unforeseen amount of time.

-As this risk is unable to be predicted there is no way to mitigate this risk other than know that it can happen and keep all of our work in local storage where possible. We will use github which is a decentralised repository which allows for the user to have a local copy of the current working build offline.

## Testing:

As testing is an integral part of our chosen software development methodology we must ensure that we are testing everything that is necessary to ensure the success of our product. Each module will be JUnit tested using the JUnit libraries imported into our chosen IDE. All build scripts and commits to the git repository will be auto tested by Travis CI. Each time a git push is made it is tested before it is deployed, to ensure that the code is valid. If the code is invalid the build fails. Gradle is a build automation system which tests our code on build and generates code coverage and test reports. This is done to ensure that we know what is working and what is not and so that we have a log of what tests were run. For our XML verification we will use the blockly schema they have built for the API. This will verify that our blocks that we create will work with the blockly visual language. The front end testing will be done by HTML and CSS validators found online at validator.w3.org. For each JSP on the front end we will run tests to validate that the information that is parsed as Java is correctly translated into our respective web technologies. Once the project is built we will complete end user testing, we plan for this to occur in week 8 of the project as this should give us enough time to internally alpha test, make further changes to the product and then deploy some beta tests. Our end user testing will be tested with the key stakeholders (10-14 year olds). We will set up a meeting to share our work with kids in the classroom so that we get some real world use cases. This implies some ethical issues which will be worked on with our project supervisor.

## Issue Tracking:

Bugs and issues will always appear throughout the timeline of the project. To ensure that each bug/issue is addressed properly we have decided to use GitHub as our main repository as this repository provides built in issue tracking including the ability to add new issues, mark current issues as complete and assign issues to specific members of the group. GitHub also provides a Wiki page for our group to lay out a roadmap of our product, show the current status of the project and document our project together.

## Appendices:

*Timeline Planning:*

https://docs.google.com/spreadsheets/d/1AWDUAikX68S3jjjZb_aQyp-35hKiNGO59wKV4aAwJU8/edit?usp=sharing

*Project Management Software Comparison:*

https://docs.google.com/spreadsheets/d/1vgTs9FM70Gdumatlo2v13aWmxEuc0qq1ddnKxjz6RAs/edit?usp=sharing

*Software Development Methodologies Comparison:*

https://docs.google.com/spreadsheets/d/1-BnKNw6Asyq1ghioHFOLzGbvZ7C7WLSfIYaILCG6pls/edit?usp=sharing

*Technology Features Comparison:*

https://docs.google.com/spreadsheets/d/1a2SKUH6Te4PcZY_ISWps7acFzkHhxzeubbzKQwzd-qE/edit?usp=sharing

*Architecture High Level Design:*

https://www.lucidchart.com/invitations/accept/43304020-2aff-4df5-b721-f21700d7d539

## References:

Project link:
capstone.daved.nz
GitHub:
https://github.com/
TravisCI:
https://travis-ci.org/
Java:
https://www.oracle.com/java/index.html
W3Validator:
https://validator.w3.org/
HTML 5:
http://www.w3.org/TR/html5/
AngularJS:
https://angularjs.org/
CSS:
http://www.w3.org/Style/CSS/Overview.en.html
Facebook:
https://www.facebook.com
Google Groups:
https://groups.google.com
Blockly:
https://developers.google.com/blockly/