Final Project

Project Report

Procreate

Author: Jared Schroeder CIT025989

Due: 12 Nov. 2014

Abstract

Glossary of Terms

Acronym	Meaning
AIE	Academy of Interactive Entertainment
CIT	Canberra Institute of Technology
GUI	Graphical User Interface
WBS	Work Breakdown Structure
WPF	Windows Presentation Foundation

Table of Contents

1. Project Overview

1.1. Goals

1.2. Risks

There are several risks associated with this project. Each is detailed below, with information including description, likelihood, impact, avoidance and remediation actions.

1.2.1. Risk Listing

R1IncompleteDesign

Description: Procreate program design is missing or has overlooked some required functionality.

Likelihood: Medium

Impact: Medium

Avoidance: Difficult to avoid, however care has been taken to assure that the design meets most of the required functionality, so any cases of this risk eventuating shouldn't alter the design too much.

Contingency plan: If a change in design is required, changes should be fully considered so that the best option is chosen. The changes will be recorded in Section 5 (Document History) of this document, and the development blog must be updated to describe what changes were made and why.

R2C#DesignIssues

Description: The Procreate program design is incompatible with certain design areas of WPF, C# and .NET.

Likelihood: High

Impact: Low

Avoidance: The best way of avoiding this is to research the WPF, C# and .NET design, and then design Procreate to mix well with these designs. This has been done to an extent, however some areas like the GUI elements of WPF weren't researched before Procreate was designed, so they will probably be incorrect and need to be updated.

Contingency plan: The only way to meet this risk is to alter Procreate's design to suit the other designs. Changes to Procreate's design need to be well considered before they are applied, and once applied the changes need to be noted section 5 (Document History) of this document, and the development blog must be updated to describe what changes were made and why.

R3AlgorithmOvertime

Description: A procedural generation algorithm takes longer to implement than expected. This is possible because the algorithms are complex and will need to be debugged to make sure they're working correctly.

Likelihood: Medium

Impact: Medium

Avoidance: The algorithms used have been researched to gain a full understanding of how they work. If an algorithm is more difficult than expected, other programming students and teachers at AIE will be asked for their input to solve the issue. Also, finding code samples of the algorithm (particularly in C#) will provide good guidance for implementing the algorithms.

Contingency plan: If an algorithm is taking longer to implement than expected, it can be granted an extra week in the implementation plan/WBS. This will affect Section 3.1. (Implementation Plan) of this document, so if an extra week is allocated this must be updated in the Implementation Plan.

R4CantDoAlgorithm

Description: A procedural generation algorithm cannot be implemented for whatever reason, even with an extra week allocated to it (as described in R3AlgorithmOvertime).

Likelihood: Low

Impact: High

Avoidance: The algorithms used have been researched to gain a full understanding of how they work. Also, finding code samples of the algorithm (particularly in C#) will provide good guidance for implementing the algorithms.

Contingency plan: Help from AIE teachers and students, as well as the CIT mentors will be asked for, to complete the algorithm. If it still cannot be implemented, it will have to be removed from the design. In accordance, the design documents will need to be updated, Section 5 (Document History) of this document will need to be updated, a blog post must be written describing the removal of the algorithm, why it was removed, and the steps that were taken to avoid its removal. Any parts of the Implementation Plan that are altered will also need to be updated in Section 3 of this document.

R5Behind Schedule

Description: Development is lagging behind the schedule mentioned in the Implementation Plan.

Likelihood: High

Impact: Medium

Avoidance: The Implementation Plan must be strictly adhered to, to avoid any chance that development starts lagging behind. Trello will also be used to notate what parts of the development should be complete at which week of the course.

Contingency plan: If the time behind schedule is only a few days, it can be left as is. However if it is weeks behind, some features and elements in the Implementation Plan will have to be decreased or removed completely. Section 2.1.2. indicates the less important program modules that should be decreased if this risk eventuates. Any changes made here need to be updated in sections 3.1, 4.1 and 5 of this document. Information about what is being decreased/removed and why also needs to be noted in the blog.

R6OtherWork

Description: Other work takes up time that was allocated to Procreate development schedule.

Likelihood: High

Impact: High

Avoidance: The workload for the Semester 2 2014 time period will be thoroughly examined, scheduled and planned out at the start of the semester. This schedule will be constantly referenced and updated to make the best use of available time.

Contingency plan: If time for this and other work cannot be correctly managed, the less importance work (probably the other work) will have to be reduced in order to prioritise the more important work. If the workload is shared between equality important work, features may have to be reduced from both to ensure that both will be completed on time. Any changes made here need to be updated in sections 3.1, 4.1 and 5 of this document. Information about what is being changed/removed and why also needs to be noted in the blog.

R7ComputerFail

Description: The development computer breaks, either temporarily or permanently.

Likelihood: Low

Impact: High

Avoidance: The computer will not be used in any way that may damage it or cause it to break.

Contingency plan: If the computer breaks temporarily or permanently, a computer at AIE or CIT will have to be used. This required contacting either institute to organise it. This will probably affect the project's costs because extra money will be spent on car travel. This will also affect the ease of access to development, and section 3.1.1. of this document will need to be updated to account for the extra time required for travelling to CIT or AIE.

R8WorkLoss

Description: Some of the work is lost for whatever reason.

Likelihood: Low

Impact: High

Avoidance: Online backups of work will be stored on GitHub whenever changes are made, and local backups will also be stored on the development computer, on a Macbook, and on a USB. This will hopefully decrease the chance of a large amount of work being lost.

Contingency plan: The latest existing backup will be located and used. If this backup is missing important recent work, discussion with HC Lim will take place to see if extra development time can be allocated due to the work loss. If this cannot be arranged, the Implementation Plan in section 3.1. of this document will have to be revised to see where time needs to be reallocated to catch up. Any changes made here must be noted in sections 3.1, 4.1 and 5 of this document, and the blog will also require updating to note what was removed, and why.

R9NoTesters

Description: The 2 testers aren't available to test the Procreate program.

Likelihood: Low

Impact: High

Avoidance: The testers will be notified of upcoming test dates a few weeks prior to the date. If they can't test on that date, changes will be made so that they can test on a close date before or after the original.

Contingency plan: If both testers still cannot test on the designated dates or any nearby dates, searching for another tester will take place for a week. If no alternative tester can be found, the testing will be left until any tester can be found. At this point development work will continue without user testing. Any changes made here must be reflected in sections 3.1, 4.3, and 5 of this document, and the blog will also require updating to noted the changes made and why.

R10SlowTester

Description: A tester takes longer to test than expected.

Likelihood: Low

Impact: Low

Avoidance: Testers will be notified of how much time they have been provided to complete the testing. The will also be encouraged to notify the project manager if anything is impeding or slowing down their testing so that new arrangements can be made.

Contingency plan: The tester will be given as much as a week longer to complete the testing, but at this point any feedback they provide will not affect the test result. Depending on if the project is on schedule, an extra week or two can be provided to the tester, however in this case the development will continue simultaneously. Any changes made here must be detailed in sections 3.1, 4.3, and 5 of this document. The blog will also need to be updated to explain the changes and why they were made.

R11TestNotReady

Description: The Procreate program doesn't contain the test criteria required for a test to take place.

Likelihood: Medium

Impact: High

Avoidance: Prior to the test date, if test criterion aren't completed and don't rely on incomplete predecessors, full focus will be placed on these criterion. Any work skipped by this process will be completed after user testing.

Contingency plan: Testers will be notified that the Procreate program isn't ready for testing, and testing will be rescheduled for a week later, or the closest date when the testers are available.

R12ScopeCreep

Description: Too much user feedback is being developed, rather than the original parts of the design. This is taking up too much time and causing the original parts to lag behind their scheduled completion dates.

Likelihood: Low

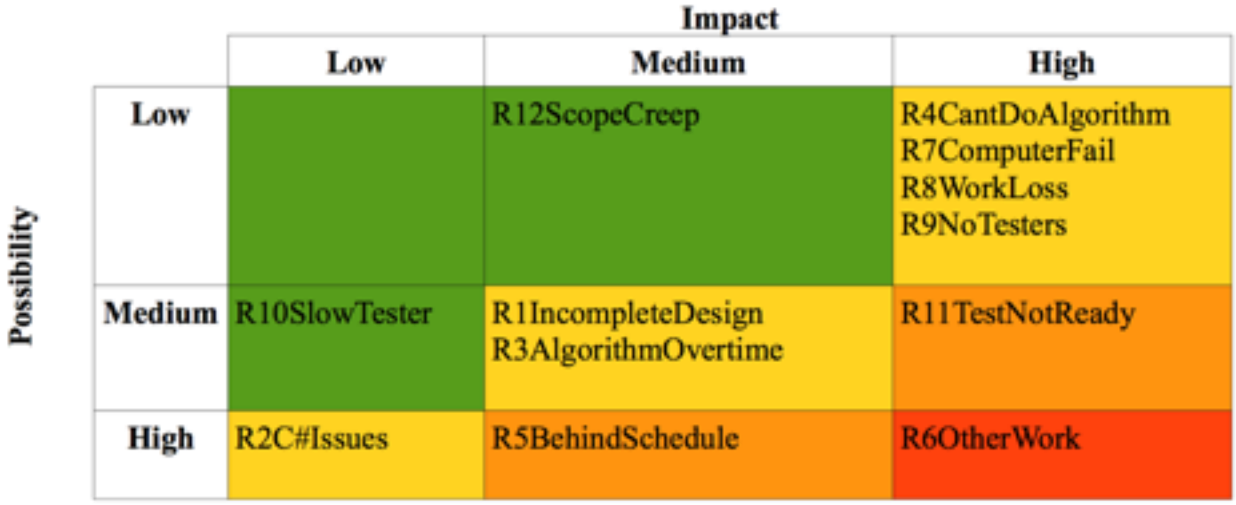
Impact: Medium

Avoidance: No suggestions for new features will be considered. Any feedback will be considered thoroughly in regards to whether it can be achieved within the project schedule, and if it is of great worth to the current project. Any interesting proposal will be placed on a backlog, which will be considered after the project's lifetime has concluded (i.e. after the Final Project course has finished).

Contingency plan: Any incomplete feedback-based work will be stopped, and Procreate will be restored to a previous version prior to when this work was started. If a significant amount of time has been lost due to working on user suggestions, some of the original work in the Implementation Schedule will have to be removed to compensate for the lost time. Any changes here must be detailed in sections 3.1, 4.1 and 5 of this document. The blog will also need to be updated, to indicate what parts of the schedule were removed, and why.

1.2.2. Risk Matrix

The following risk matrix indicates the most important risks, according to their general likelihood/possibility and impact values. These values may change for different stages of development, meaning that this matrix is only a general indication and doesn't represent the every stage of development.



		Impact		
		Low	Medium	High
Possibility	Low		R12ScopeCreep	R4CantDoAlgorithm R7ComputerFail R8WorkLoss R9NoTesters
	Medium	R10SlowTester	R1IncompleteDesign R3AlgorithmOvertime	R11TestNotReady
	High	R2C#Issues	R5BehindSchedule	R6OtherWork

FIGURE 1: RISK MATRIX

2. Program Design

sdfsdfsdfsdfsdfsdfsdfsdfsdfsdf

sdfsdfsdfsdfsdfsdfsdfsdfsdfsdf

2.1. Modules

The Procreate program is broken up into 5 separate modules that are shown in the following figure.

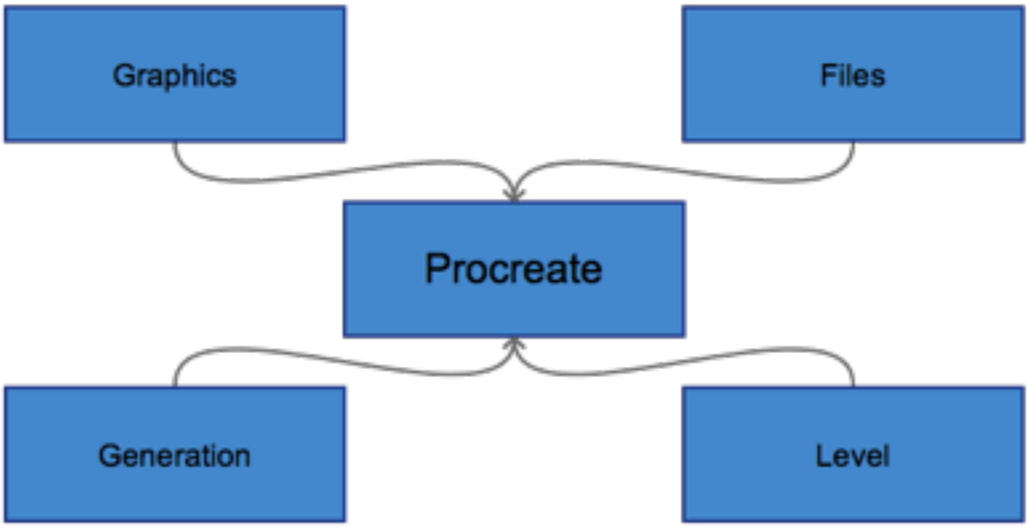


FIGURE 2: PROCREATE'S SYSTEM MODULES

As the figure shows, every module is a part of the Procreate module. As such, the Procreate module is the most important because it runs the whole application. Each module is detailed in the following section.

2.1.1. Module Descriptions

Procreate

Graphics

Level

Generation

Files

2.1.2. Order of Importance

In terms of importance, the modules are listed in the following order:

1. Procreate
2. Graphics
3. Level

4. Generation
5. Files

Procreate is the most important for the reason mentioned previously. The graphics module is the next important because without it the user won't be able to see the program at all. Next is the Level module, because if the Level doesn't exist the application won't be capable of creating levels, which is its main purpose. Following this is the Generation module, which isn't required for creating levels, however the user won't be able to use any procedural generation methods on the Level without it. The least important module is the Files, because the user can still use the application without it, however they won't be able to save or load their levels if this is the case.

2.1.3. Order of Risk Factor

In terms of risk factor, the modules are listed in the following order:

1. Generation
2. Graphics
3. Files
4. Level
5. Procreate

The Generation module is the most risky because it contains complex procedural generation algorithms that will be difficult, and time-consuming to implement. This module represents the hardest part of implementing the application. The Graphics module is the second-riskiest, because it requires not only the drawing of the level, but also linking level, procedural generation algorithm and game object data with the GUI so that any changes made are reflected in the GUI and the level. The large amount of application data will have to be correctly saved and loaded within the Files module, which makes it the next riskiest module. The Level module is less risky because it is simply a container without any complex functions, however it must be implemented correctly so that the level is stored correctly. The Procreate module is the least risky because it doesn't perform any complex actions - it passes on any work to the other modules.

3. Project Implementation

3.1. Implementation Plan

3.1.1. WBS

The project's WBS is shown in Figure 3. It starts on the 25th of August and concludes on the 16th of October. In most cases, the large sections of work have been classified by module, however in some cases the activities for a particular module are scattered across the WBS, so these could not be grouped by module.

The WBS also details the start and finish times for each activity, however these won't represent the actual times, because the work times for each day will be a bit more flexible. Each time was generated automatically by ProjectLibre, and they are not really important because the WBS was created according to the day rather than the hour. This means that the dates are more important information than the times.

3.1.2. Gantt Chart

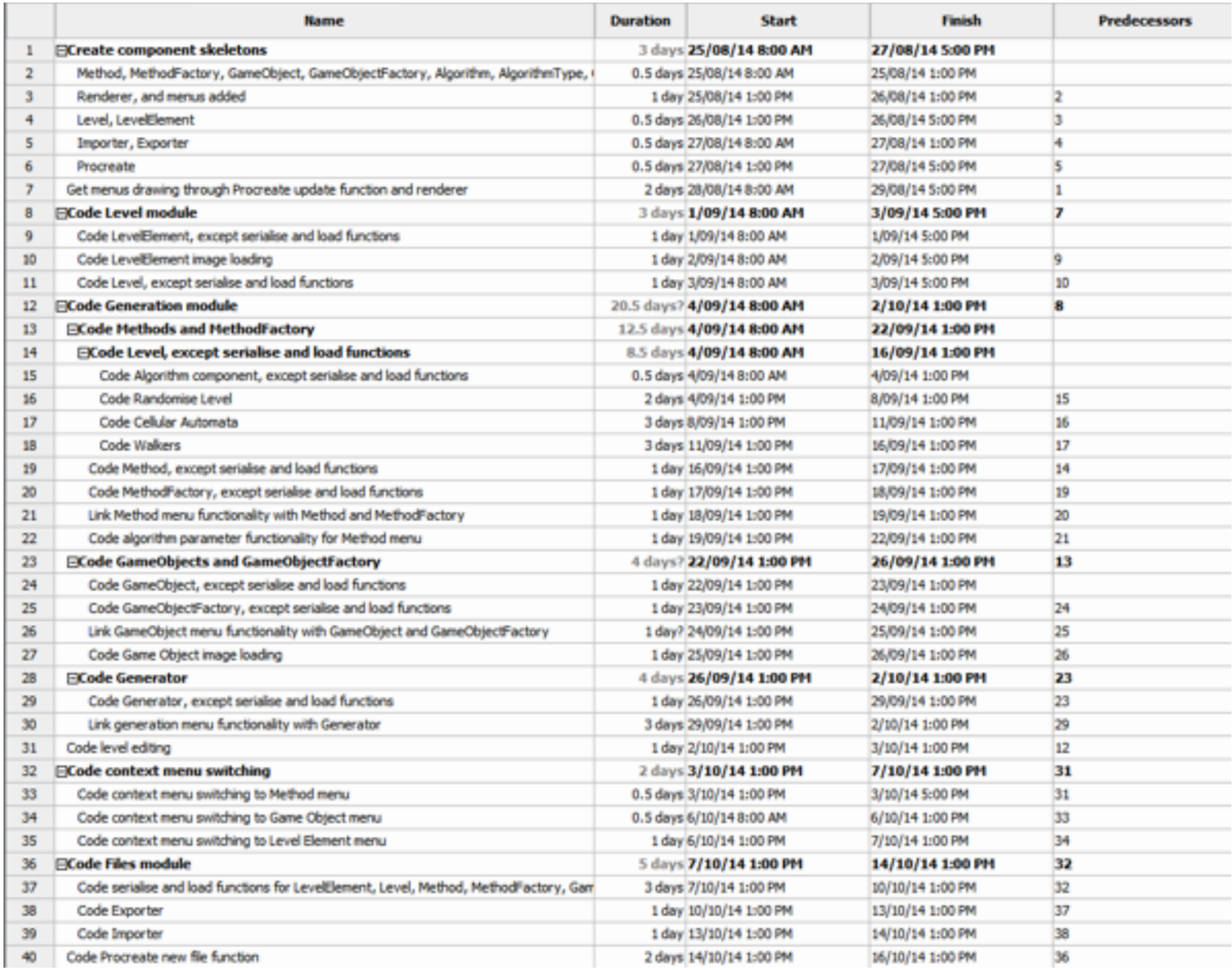
The project's Gantt Chart is shown in Figure 4. The red boxes indicate individual activities, and the black boxes indicate groups of individual activities. The chart was created in accordance with the WBS, so the information in each should line up perfectly.

Project milestones are represented by the dates scattered throughout the chart. There are 7 milestone dates in total.

3.1.3. Milestones

The project has 7 milestones, as are indicated by dates in the Gantt Chart. These milestones are listed below in chronological order.

Milestone	Due Date
Menus are all drawing	30th August
Level is coded and drawing	4th September
Procedural generation methods coded and correct	22nd September
Game objects coded	26th September
Level generation via generate button working	2nd October
Level editing working	3rd October
Files saving and loading (and project completion)	17th October



	Name	Duration	Start	Finish	Predecessors
1	⊟**Create component skeletons**	3 days	**25/08/14 8:00 AM**	**27/08/14 5:00 PM**	
2	Method, MethodFactory, GameObject, GameObjectFactory, Algorithm, AlgorithmType,	0.5 days	25/08/14 8:00 AM	25/08/14 1:00 PM	
3	Renderer, and menus added	1 day	25/08/14 1:00 PM	26/08/14 1:00 PM	2
4	Level, LevelElement	0.5 days	26/08/14 1:00 PM	26/08/14 5:00 PM	3
5	Importer, Exporter	0.5 days	27/08/14 8:00 AM	27/08/14 1:00 PM	4
6	Procreate	0.5 days	27/08/14 1:00 PM	27/08/14 5:00 PM	5
7	Get menus drawing through Procreate update function and renderer	2 days	28/08/14 8:00 AM	29/08/14 5:00 PM	1
8	⊟**Code Level module**	3 days	**1/09/14 8:00 AM**	**3/09/14 5:00 PM**	**7**
9	Code LevelElement, except serialise and load functions	1 day	1/09/14 8:00 AM	1/09/14 5:00 PM	
10	Code LevelElement image loading	1 day	2/09/14 8:00 AM	2/09/14 5:00 PM	9
11	Code Level, except serialise and load functions	1 day	3/09/14 8:00 AM	3/09/14 5:00 PM	10
12	⊟**Code Generation module**	20.5 days?	**4/09/14 8:00 AM**	**2/10/14 1:00 PM**	**8**
13	⊟**Code Methods and MethodFactory**	12.5 days	**4/09/14 8:00 AM**	**22/09/14 1:00 PM**	
14	⊟**Code Level, except serialise and load functions**	8.5 days	**4/09/14 8:00 AM**	**16/09/14 1:00 PM**	
15	Code Algorithm component, except serialise and load functions	0.5 days	4/09/14 8:00 AM	4/09/14 1:00 PM	
16	Code Randomise Level	2 days	4/09/14 1:00 PM	8/09/14 1:00 PM	15
17	Code Cellular Automata	3 days	8/09/14 1:00 PM	11/09/14 1:00 PM	16
18	Code Walkers	3 days	11/09/14 1:00 PM	16/09/14 1:00 PM	17
19	Code Method, except serialise and load functions	1 day	16/09/14 1:00 PM	17/09/14 1:00 PM	14
20	Code MethodFactory, except serialise and load functions	1 day	17/09/14 1:00 PM	18/09/14 1:00 PM	19
21	Link Method menu functionality with Method and MethodFactory	1 day	18/09/14 1:00 PM	19/09/14 1:00 PM	20
22	Code algorithm parameter functionality for Method menu	1 day	19/09/14 1:00 PM	22/09/14 1:00 PM	21
23	⊟**Code GameObjects and GameObjectFactory**	4 days?	**22/09/14 1:00 PM**	**26/09/14 1:00 PM**	**13**
24	Code GameObject, except serialise and load functions	1 day	22/09/14 1:00 PM	23/09/14 1:00 PM	
25	Code GameObjectFactory, except serialise and load functions	1 day	23/09/14 1:00 PM	24/09/14 1:00 PM	24
26	Link GameObject menu functionality with GameObject and GameObjectFactory	1 day?	24/09/14 1:00 PM	25/09/14 1:00 PM	25
27	Code Game Object image loading	1 day	25/09/14 1:00 PM	26/09/14 1:00 PM	26
28	⊟**Code Generator**	4 days	**26/09/14 1:00 PM**	**2/10/14 1:00 PM**	**23**
29	Code Generator, except serialise and load functions	1 day	26/09/14 1:00 PM	29/09/14 1:00 PM	23
30	Link generation menu functionality with Generator	3 days	29/09/14 1:00 PM	2/10/14 1:00 PM	29
31	Code level editing	1 day	2/10/14 1:00 PM	3/10/14 1:00 PM	12
32	⊟**Code context menu switching**	2 days	**3/10/14 1:00 PM**	**7/10/14 1:00 PM**	**31**
33	Code context menu switching to Method menu	0.5 days	3/10/14 1:00 PM	3/10/14 5:00 PM	31
34	Code context menu switching to Game Object menu	0.5 days	6/10/14 8:00 AM	6/10/14 1:00 PM	33
35	Code context menu switching to Level Element menu	1 day	6/10/14 1:00 PM	7/10/14 1:00 PM	34
36	⊟**Code Files module**	5 days	**7/10/14 1:00 PM**	**14/10/14 1:00 PM**	**32**
37	Code serialise and load functions for LevelElement, Level, Method, MethodFactory, Gar	3 days	7/10/14 1:00 PM	10/10/14 1:00 PM	32
38	Code Exporter	1 day	10/10/14 1:00 PM	13/10/14 1:00 PM	37
39	Code Importer	1 day	13/10/14 1:00 PM	14/10/14 1:00 PM	38
40	Code Procreate new file function	2 days	14/10/14 1:00 PM	16/10/14 1:00 PM	36

FIGURE 3: PROJECT WBS

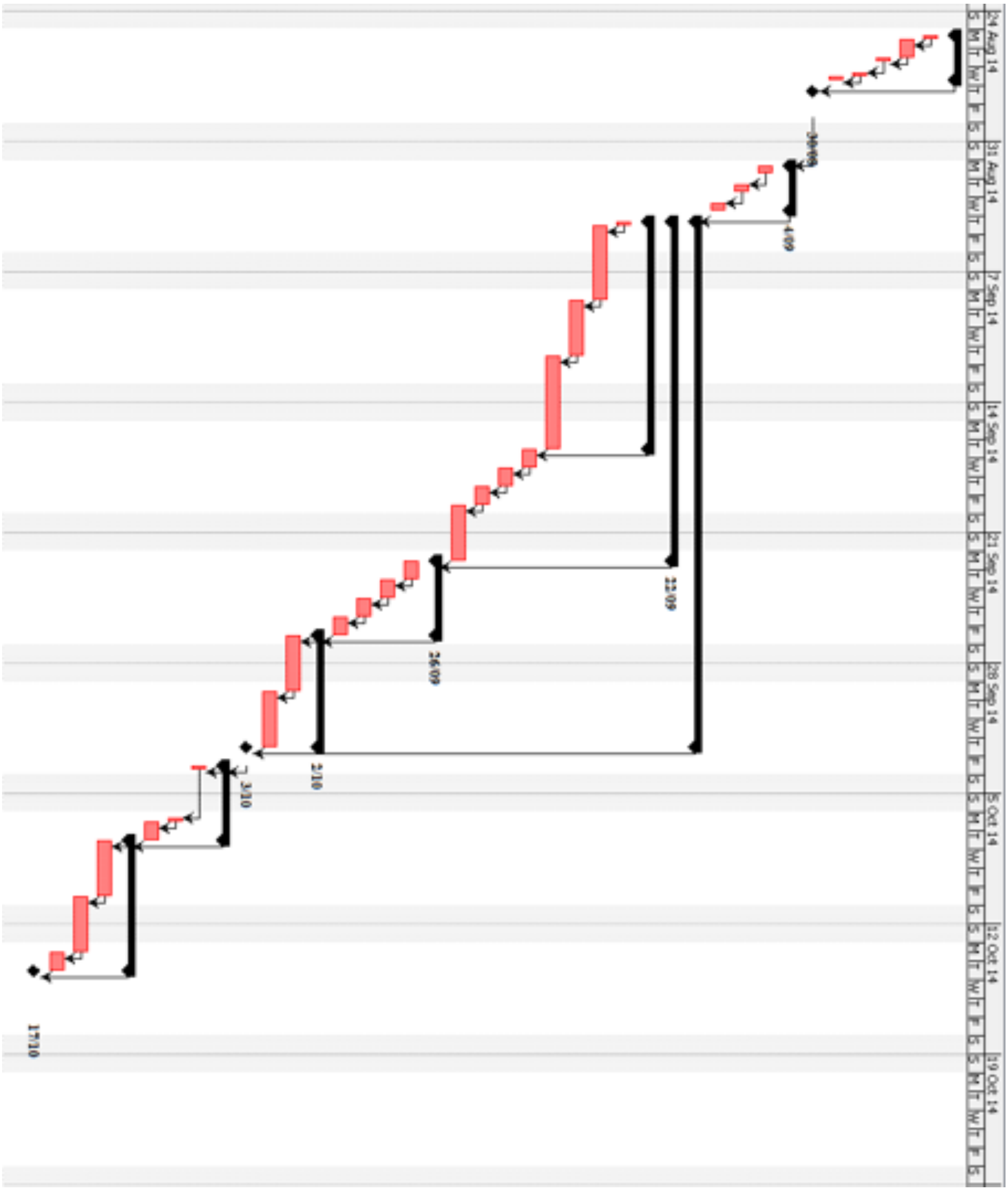


FIGURE 4: PROJECT GANTT CHART

3.2. Blog

The blog for Procreate’s development project is located at http://jaredschroeder.wordpress.com/category/procreate/.

3.3. Version Control

All Procreate’s development files and documentation will be version-controlled with GitHub. The GitHub repository for the project is located at https://github.com/jarschroe/Procreate.

4. Project Events

4.1. Major Design Changes

4.2. Eventuated Risks

This section details the risks that took place during the project's lifetime, with information including their impact on the project, how they were handled, and what changes were made to decrease the their chance of repeating.

4.3. User Testing Events

5. Document History

This section notes the changes that have been made to this document throughout the project's lifetime.