**project Development plan**

**Battleboats Project**

**CMSC495 - Spring 2014**

**January 26, 2014**

Team Name: GameA (Battleboats)

Team Member: Tyler Cheatham

Team Member: Angelica Mckinney

Team Member: Joshua Mefford

Team Member: Jonathan Menard

**Preface**

This Project Management Plan (PMP) is intended to provide guidance on the management of the Battleboats Project. This document conforms to the Institute of Electrical and Electronics Engineers (IEEE) Standard for Software Project Management Plans, IEEE Std 1058-1998, for format and content.

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

Table of Contents

[SECTION 1. OVERVIEW 4](#_Toc378535756)

[1.1 Project Summary 4](#_Toc378535757)

[1.1.1 Purpose, Scope, and Objectives 4](#_Toc378535758)

[1.1.2 Assumptions and Constraints 4](#_Toc378535759)

[1.1.3 Project Deliverables 5](#_Toc378535760)

[1.2 Evolution of the Plan 6](#_Toc378535761)

[1.3 Document Structure 6](#_Toc378535762)

[SECTION 2. REFERENCES 8](#_Toc378535763)

[2.1 Standards and Documents 8](#_Toc378535764)

[SECTION 3. DEFINITIONS AND ACRONYMS 9](#_Toc378535766)

[SECTION 4. PROJECT ORGANIZATION 12](#_Toc378535767)

[4.1 External Interfaces 12](#_Toc378535768)

[4.2 Project Roles and Responsibilities 12](#_Toc378535769)

[SECTION 5. MANAGEMENT PROCESS 14](#_Toc378535770)

[5.1 Start-up 14](#_Toc378535771)

[5.1.1 Resource Requirements 14](#_Toc378535772)

[5.2 Work Plan 15](#_Toc378535773)

[SECTION 6. TECHNICAL PROCESS 17](#_Toc378535774)

[6.1 Process Model 17](#_Toc378535775)

[6.2 Methods, Tools and Techniques 17](#_Toc378535776)

[6.3 Project Infrastructure 17](#_Toc378535777)

[6.4 Product Acceptance 18](#_Toc378535778)

[6.5 Input/output scenarios 19](#_Toc378535779)

[SECTION 7. SUPPORTING PROCESSES 22](#_Toc378535781)

[7.1 Configuration Management 22](#_Toc378535782)

[7.2 Quality Assurance 22](#_Toc378535783)

[7.3 Problem Resolution 22](#_Toc378535784)

[7.4 Process Improvement 22](#_Toc378535785)

# SECTION 1. OVERVIEW

## 1.1 Project Summary

Battleboats is a two-dimensional, side-scrolling, space-themed computer game. It is designed to be played by one player with level progression and unlockable features. In this game, the player controls a spaceship in a side-scrolling level and must shoot enemy spaceships which fly in from the other side of the screen. The main goal of the game is to make it through each level without being destroyed while trying to destroy as many enemy spaceships as possible.

### 1.1.1 Purpose, Scope, and Objectives

This section describes the purpose, scope and objectives of this project.

**1. 1. 1. 1 Purpose**

The purpose of the Battleboats project is to develop an entertaining computer game in a short time frame with limited resources.

**1. 1. 1. 2 Scope**

The scope of this project will be the development of one computer game with multiple levels and features for the players of the game. The game will be single player and will take approximately eight weeks to develop.

**1. 1. 1. 3** **Objectives**

The main objectives of this project are as follows:

- Develop the main, playable game which will consist of five levels.

- Provide an efficient implementation of the game mechanics.

- Ensure the game is high quality (i.e. responsive, entertaining, aesthetically pleasing).

The optional objectives of this project are as follows:

- Continue creating playable levels while time allows.

- Continue adding extra features while time allows.

- Add extra spaceships for the player to control.

### 1.1.2 Assumptions and Constraints

This section details assumptions and constraints involved in the creation of this project.

**1. 1. 2. 1 Assumptions**

1. Playability. It is assumed that players of this game will be familiar with the basic premise of computer games and will be able to play the game with only the information contained in the user manual and an optional tutorial.

2. System. It is assumed that anyone attempting to play this game will have hardware and software which meet the minimum requirements to play this game. These minimum requirements shall be a recent version of the Windows operating system and the most recent version of the Java Runtime Environment. For input, the user will require a keyboard and a mouse. For hardware, any computer which can efficiently run the latest version of Windows will be suitable to run this game.

3. Language. All documents will be developed in English and it is assumed that the readers of the documents will be fluent in English.

**1. 1. 2. 2 Constraints**

1. Time. This game will be developed in a much shorter time frame than is normally allowed for game development. This time factor will be the single greatest constraint on the project.

2. Resources. Since this is not a professional production, resources (images, sounds, etc.) will have to be gathered from existing materials found on the Web. These resources will have to be free since this project does not have any funds which can be utilized. Some small graphics and sound files can be created from software which some of the project team currently has, but the use of these self-created resources will be limited since the possible “amateur” nature of the designs will detract from the overall quality of the game.

3. Money. As stated in (2), there is no pool of monetary funds which can be utilized for this project. As such, all elements of this project will have to be freely available for use.

4. Tools. The project must be programmed in a high-level, object oriented language. To meet this goal, the project will be coded in Java.

### 1.1.3 Project Deliverables

This section documents the deliverables and the schedule for the deliverables. All deliverables will be turned in to Professor Duchon via the WebTycho interface.

Table 1-1

|  |  |
| --- | --- |
| **Deliverable** | **Date** |
| Project Plan | 01/26/2014 |
| User’s Guide | 02/02/2014 |
| Test Plan | 02/02/2014 |
| Design | 02/09/2014 |
| Phase 1 Documentation | 02/16/2014 |
| Phase 2 Documentation | 02/23/2014 |
| Phase 3 Documentation | 03/02/2014 |
| Final Project Code | 03/09/2014 |
| Final Documentation | 03/09/2014 |

## 1.2 Evolution of the Plan

Meetings with all group members will be scheduled a minimum of twice weekly. These meetings will discuss current progress, issues encountered in the development of the project, possible adjustments to the project timeline or workflow and any other matters which the group feels are pertinent to the completion of this project.

Should any major changes be made to the project plan, the changes will be reflected in the project plan by the Sunday of the week in which the changes were implemented.

## 

## 1.3 Document Structure

This plan is organized as follows:

1. Section 1, Project Overview. This section provides an overview of the scope and objectives of the project, the project’s assumptions and constraints, reference to the project deliverables, schedule, and a description of the evolution of the plan.
2. Section 2, References. This section provides a list of all documents, policies, templates, processes, and other sources of information referenced in the plan.
3. Section 3, Definitions. This section contains the abbreviations and acronyms required to properly understand this planning document.
4. Section 4, Project Organization. This section identifies interfaces to organizational entities external to the project, the project’s internal organizational structure, and defines roles and responsibilities for the project.
5. Section 5, Management Process. This section describes the planning, measurement, tracking, reporting, and risk control mechanisms needed to provide management control over the technical processes and product quality, and appropriate project initiation and closeout procedures.
6. Section 6, Technical Process. This section describes the technical solution in terms of a process model and implementation methods, tools, and techniques to be used to develop the various work products, plans for establishing and maintaining the project infrastructure, and the product acceptance.
7. Section 7, Supporting Processes. This section describes processes that are employed to facilitate and control the technical processes and the state of the product. These include, but are not limited to, configuration management, verification and validation, documentation, quality assurance, reviews and audits, problem resolution and methods to ensure continuous process improvement.

# SECTION 2. REFERENCES

## 2.1 Standards and Documents

The standards and documents listed below are referenced in this document:

[1] Chairperson, J. (1990). IEEE Standard Glossary of Software Engineering Terminology, 610. Retrieved from <http://dis.unal.edu.co/~icasta/GGP/_Ver_2012_1/Documentos/Normas/610-12-1990.pdf>

[2] Oracle. Java. Retrieved from http://java.com/en

[3] Leinich, D. (2013). The Lightweight Java Game Library (LWJGL). Retrieved from http://lwjgl.org

# [4] Slick2D (2013). 2D Java Game Library. Retrieved from http://slick.ninjacave.com

# SECTION 3. DEFINITIONS AND ACRONYMS

This section describes definitions and acronyms for the whole documents.

**ASDW:** Four keyboard keys that are used to interact with video games. The W and S keys control up and down movement, while A and D are left and right. This key configuration is an alternative to using the directional arrow keys.

**baseline:** A work product that has been formally reviewed and agreed upon, and that can be changed only through formal change control procedures. A baseline work product may form the basis for further work activity(s).

**CPU (central processing unit):** The hardware within a computer that carries out the instructions of a computer program by performing the basic arithmetical, logical and input/output operations of the system.

**documentation**: A collection of documents on a given subject. Any written or pictorial information describing, defining, specifying, reporting or certifying activities, requirements, procedures or results.

**GitHub:** A web-based hosting service for software development projects that use the Git revision control system. GitHub offers both paid plans for private repositories, and free accounts for open source projects.

**JDK 7 (Java SE Development Kit 7):** Lets you develop and deploy Java applications on desktops and servers.

**JRE:** Java Runtime Environment. This is the running environment for the Java language which interprets programs written in Java.

**library:** A controlled collection of software and related documentation designed to aid in software development, use or maintenance.

**Lightweight Java Game Library (LWJGL):** provides developers access to high performance crossplatform libraries such as OpenGL (Open Graphics Library), OpenCL (Open Computing Language) and OpenAL (Open Audio Library) allowing for state of the art 3D games and 3D sound. Additionally LWJGL provides access to controllers such as Game-pads, Steering wheel and Joysticks.

**patch:** To perform a modification as a last-minute fix or afterthought.

**side scroller:** A game in which the screen is continuously scrolling in one direction and the player must navigate in the opposite direction.

**slick2D:** An easy to use set of tools and utilites wrapped around LWJGL (Lightweight Java Game Library) and OpenGL(Open Graphics Library) bindings to make 2D Java game development easier.

**sprite:** A computer graphic which represents some visual aspect of a game. These images are usually small and come in large sheets which can be cropped to only utilize certain selections from the sheet.

**test:** An activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component.

**testing:** The process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component.

**user guide:** A document that presents the information necessary to employ a system or component to obtain desired results. Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions.

**validation:** The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements.

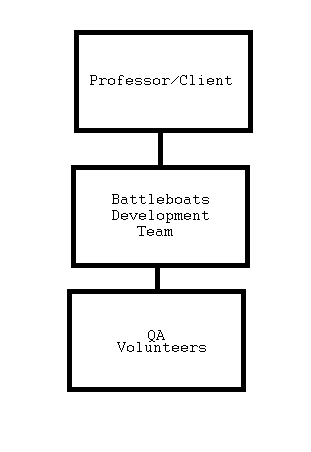
# SECTION 4. PROJECT ORGANIZATION

This section outlines the all the external interfaces relating to this project, as well as the internal team structure and team roles.

## 4.1 External Interfaces

External Interfaces take the form of relationships between three primary groups. Professor Nick Duchon is the recipient of this final project. The Battleboats Development Team is responsible for the majority of the project, and must interface with the professor from time to time. The QA Volunteers will receive project assets and direction from the Development Team, as well as provide their QA testing results to the Development Team.

Figure 4-1



## 4.2 Project Roles and Responsibilities

This section lists team personnel and their responsibilities.

**Project Lead- Jonathan Menard**

Project Lead is responsible for personnel management, schedule creation, project realization and initial project planning.

**Programming Lead- Joshua Mefford**

Programming Lead is responsible for architectural design and implementation, code management, direction and maintenance.

**Documentation Lead- Angelica Mckinney**

Documentation Lead is responsible for creation and maintenance of critical project documentation.

**Testing/Art Lead- Tyler Cheatham**

Testing Lead is responsible for the coordination of QA testing teams and implementation of bug-fixes, as well as obtaining/creating required artwork for the project.

# SECTION 5. MANAGEMENT PROCESS

This section outlines the planning and execution requirements of this project.

## 5.1 Start-up

This start-up section lists all commitments, resources and other materials which are required to begin the project and get it off the ground.

### 5.1.1 Resource Requirements

The following requirements are necessary for the success of the Battleboats project:

Development:

* Windows Computer
* Netbeans 7.4
* Java SE Development Kit 7.51
* Lightweight Java Game Library 2.9.1
* Slick2D Build 237
* GitHub for Windows
* Java RE 7.51

**5.1.2 Project Commitments**

The table below lists all commitments between members of the development team and their due dates.

**Table 5-1**

|  |  |
| --- | --- |
| **Commitment** | **Due Date** |
| **Group Member List** | **1/19** |
| **Project Plan Draft** | **1/26** |
| **User Guide Draft** | **2/2** |
| **Test Plan Draft** | **2/2** |
| **Peer Reviews 1** | **2/2** |
| **Code Review 1** | **2/9** |
| **Design Draft** | **2/9** |
| **QA Round 1** | **2/9** |
| **Code Review 2** | **2/16** |
| **Phase 1 Documentation** | **2/16** |
| **Peer review 2** | **2/16** |
| **Phase 2 Documentation** | **2/23** |
| **Code Review 3** | **2/23** |
| **QA Round 2** | **2/23** |
| **Phase 3 Documentation** | **3/2** |
| **Code Review 4** | **3/8** |
| **QA Round 3** | **3/8** |
| **Peer Review 3** | **3/9** |
| **Project Code** | **3/9** |
| **Final Documentation** | **3/9** |

## 5.2 Work Plan

This section gives a detailed timeline of all project milestones.

**Team Creation 7 days 1/13/2014 1/19/2014**

Obtain Interested Parties 4 days 1/13/2014 1/16/2014 Jonathan

Identify Team Skills 1 day 1/16/2014 1/17/2014 Jonathan

Identify Team Roles 2 days 1/17/2014 1/19/2014 Jonathan

**Project Design 7 days 1/20/2014 1/26/2014**

Create Project Plan Draft 7 days 1/20/2014 1/26/2014 Angelica

Genre Identification 2 days 1/20/2014 1/21/2014 Jonathan

Gameplay/Mechanics Identification 2 days 1/22/2014 1/23/2014 Jonathan

Aesthetic Identification 2 days 1/24/2014 1/25/2014 Jonathan

**Project Documentation and Requirements 7 days 1/27/2014 2/2/2014**

User Guide Draft Creation 3 days 1/27/2014 1/29/2014 Angelica

Test Plan Creation 3 days 1/30/2014 2/1/2014 Tyler

Peer Reviews 1 1 day 2/2/2014 2/2/2014 Team

**Project Creation 28 days 2/3/2014 3/2/2014**

Art Asset Creation Ongoing ?? ?? Tyler

Basic Movement/Projectile Coding 6 days 2/3/2014 2/8/2014 Team

Design Draft 2 days 2/3/2014 2/4/2014 Angelica

QA Round 1 3 days 2/5/2014 2/7/2014 Tyler

Code Review 1 2 days 2/8/2014 2/9/2014 Joshua

Enemy AI Creation 5 days 2/9/2014 2/13/2014 Team

Code Review 2 1 day 2/14/2014 2/14/2014 Joshua

Phase 1 Documentation 1 day 2/15/2014 2/15/2014 Angelica

Peer Review 2 1 day 2/16/2014 2/16/2014 Team

Scoring System Creation 3 days 2/17/2014 2/19/2014 Team

QA Round 2 3 days 2/20/2014 2/22/2014 Tyler

Phase 2 Documentation 2 days 2/20/2014 2/21/2014 Angelica

Code Review 3 2 day 2/22/2014 2/23/2014 Joshua

Level Creation 5 days 2/24/2014 2/28/2014 Team

Phase 3 Documentation 2 days 3/1/2014 3/2/2014 Angelica

**Final Testing Push 4 days 3/3/2014 3/6/2014**

Code Review 4 2 days 3/3/2014 3/4/2014 Joshua

QA Round 3 2 days 3/5/2014 3/6/2014 Tyler

**Final Report 3 days 3/7/2014 3/9/2014**

Report Preparation/Polish 2 days 3/7/2014 3/9/2014 Team

Documentation Release 1 days 3/9/2014 3/9/2014 Team

Final Code Release 1 days 3/9/2014 3/9/2014 Joshua

# 

# 

# 

# 

# SECTION 6. TECHNICAL PROCESS

## 6.1 Process Model

The overall development methodology which will be followed for this project is a loose waterfall development methodology. This is required due to the nature of submissions in an academic environment. The actual coding phases of development will partially implement a spiral development model since it will make the coding process more streamlined and allow for changes to be made easily.

The schedule for the development of this project is contained in the Master Schedule in Section 1 of this document and the Development Plan in section 5.

## 6.2 Methods, Tools and Techniques

This section describes all of the software and hardware tools and techniques which will be used during the development of this project.

**6.2.1 Programming Languages**

The only programming language used in the development of this project will be Java. Specifically, we will be utilizing JDK 7.

The libraries which will be used for this project are also written in Java. Specifically, Slick2d and the Lightweight Java Games Library (LWJGL) are written in Java and are the two main libraries which will be used for the game development. These libraries are, of course, in addition to the standard Java 7 library.

**6.2.2 Version Control and Collaboration**

GitHub will be used for version control and to create a shared repository of all code and documents used in this project. It will allow quick access to files for all group members and will save the changes which are made at different times by different members.

**6.2.3 Development Environment**

All team members will utilize a recent version of the Windows operating system. In addition, team members will also use the latest version of the NetBeans IDE.

## 6.3 Project Infrastructure

This section describes the controls which will be established to ensure that development proceeds smoothly and all team members are always operating with the same tools.

**6.3.1 Software Infrastructure**

Since the project is only eight weeks long and the coding phase is only four weeks long, it is not anticipated that major changes to any software or hardware tools utilized will be implemented. Regardless, each team member will perform weekly checks for updates to the three libraries used in this project as well as to the NetBeans IDE. This will ensure that all team members always have the same version of software for this project.

**6.3.2 Hardware Infrastructure**

No changes to hardware will be implemented during the course of this project. Should a team member’s development environment be rendered unusable, that team member will have to procure other means of development and ensure that the new hardware is up to the same performance standards as the old hardware. The affected team member will also have to ensure that all software is updated to the latest version and that s/he retrieves the latest documentation and code for the project from the GitHub repository.

## 6.4 Product Acceptance

A satisfactory product shall be defined as a product with minimal problems at the time of final submission. Problems which will be seen as acceptable at time of completion include any programming errors which do not cause catastrophic program failure (a crash), do not detract from overall gameplay quality and do not render the game unplayable.

Due to the subjective nature of what makes a good game, most of the metrics by which the final product can be judged must be vague.

**6.4.1 Playability**

Playability shall be judged based on the quality of the graphics, ease of play and the game response speed.

**6.4.1.1 Graphics**

Graphics will conform to the type of game being created and will not appear to be of inferior quality. Uncharacteristically blocky or pixelated graphics will not be acceptable. Jerky animations will also be grounds for rejection.

**6.4.1.2 Ease of Play**

The user should be able to remember all of the commands for a particular stage of gameplay without needing to consult manuals. For this purpose, key commands will be kept simple (no more than a two-key combination per action) and they will conform to standard gameplay conventions to the maximum extent possible (space bar to shoot, arrow keys or ASDW to move, etc.).

In addition to simple key commands, the objectives for the levels shall not be too complicated for the user to remember.

**6.4.1.3 Response Speed**

The response time of the game should be maintained at a suitable speed so that the user does not notice a delay between issuing a command and seeing it realized on the screen. The response speed will be understood to suffer during times of intense CPU usage both within and without the game (i.e. during intense battles in the game, during CPU intense processes outside the game, etc.).

## 6.5 Input/output scenarios

### This section describes the input sequences which are expected in this game and the possible outputs which should follow. Due to the complicated nature of game development and evolving requirements based on gameplay analysis, scenarios and game mechanics can require drastic rework. This section is intended only as a rough guide to how the game should function. The final documentation and user manual should be consulted for the most up-to-date information regarding game functionality.

**6.5.1 Game loading**

Upon running the game, the user will be presented with a menu screen.

1. The user selects to run the game application.
2. The program will begin running.
3. A menu screen will be displayed with options.

**6.5.2 Main Menu**

The user may select various options from a main menu screen.

1. The user is presented with a menu screen.
2. Output is determined based on user choice:
3. The user selects PLAY GAME.
4. The user is taken to the ship selection menu.
5. The user selects HOW TO PLAY.
6. The user is taken into a tutorial state.
7. The user selects CREDITS.
8. The user is taken to the credits state.
9. The user selects QUIT.
10. The program is terminated.

**6.5.3 Ship Selection Menu**

The user is presented with various ships which can be used to play the game.

1. The user is presented with the ship selection menu.
2. The user moves the mouse pointer over a ship in the menu.
3. The highlighted ship is not unlocked and the user is shown a message which reads “NOT UNLOCKED.”
4. The highlighted ship is unlocked and the user is shown the ship’s statistics.
5. The user selects an unlocked ship.
6. The user is taken to the first level of the game.

ALTERNATE

1. The user selects the BACK option on the screen.
2. The user is taken back to the main menu.

**6.5.4 Tutorial**

1. The user is presented with an introduction to the tutorial.
2. The user is shown through simulated gameplay.
3. At certain intervals, text will appear on the screen instructing the user in a specific matter of gameplay.
4. The user must acknowledge the new information in order to proceed with the tutorial.
5. The tutorial ends.
6. The user is taken back to the main menu.

**6.5.5 Credits Screen**

1. The user is shown a screen listing all of the credits for the game.
2. The credits finish.
3. The user is taken back to the main menu.

**6.5.6 Play State**

1. The user begins on the first level of the game.
2. The user’s selected ship is positioned on the left side of the screen.
3. The screen begins scrolling right to left.
4. The user’s ship moves along the screen to the left.
5. Enemies appear from the right side of the screen.
6. The user presses a directional key.
7. The user’s ship moves in the indicated direction.
8. The user presses the spacebar.
9. The user’s ship fires its main weapon.
10. The projectile collides with an enemy ship, dealing damage, or
11. The projectile misses the enemy ship and flies off the screen.
12. The enemies fire their weapons.
13. The projectile collides with the user’s ship, dealing damage, or
14. The projectile misses the user’s ship and flies off the screen.
15. Steps 6 through 10 are repeated until
16. The level ends and the user is taken to the next level.
17. The user takes enough damage from projectiles to be destroyed and is taken back to the beginning of the level.
18. The user chooses to end the game manually.

As noted in the introduction to this section, gameplay elements will be added and subtracted over the course of this project. The fundamental input/output sequences listed above are expected to remain constant. Anything designed after the publication of this project plan should be understood to be in addition to the above lists and will be reflected in these lists in the next update of this project plan.

# SECTION 7. SUPPORTING PROCESSES

## 7.1 Configuration Management

The configuration item is our product, the Battleboats game. It will be easy to play, have challenging progression and give joy and sense of achievement to the end-user. To control these attributes we will have test players and document their reactions to the game. If we need to baseline the attributes again to create a change in satisfaction levels, then we will. We can add more complexity to the game to change to a more desired effect.

Being a game and being spurred from creativity, there is no correct validation except for the interest of our target audience. That is how we would know we built the right software: based on how they react and how much they play the game. Verification can come from testing to assure game play is smooth and that it functions correctly.

## 7.2 Quality Assurance

Multiple code examinations will occur between all four project members during the project development phase.If the game develops outside the project phase, patches will be implemented for any overlooked issues with the game. Also, new features to the game will be added to keep the end-user interested.

## 7.3 Problem Resolution

Problems will be resolved in order of urgency. A problem with the consistency of the code itself would be one of first priority. A dissatisfied test review would be a second priority. The bugs would be resolved before any additions were made. Problems within management or the vacancy of a developer position would cause the responsibilities of said developer split equally among the rest of the team.

## 7.4 Process Improvement

Improvement will be handled first by perfecting the base code then making additions to the software once stability is complete. On each addition, there will be rigorous testing phases until it is certain that addition has been stabilized and runs efficiently. More additions will be made as long as there is a large demand from the end-users for more content.