Module Guide

SFWRENG 3XA3

Group 30
Alan Yin (yins1)
Huajie Zhu (zhuh5)
Junni Pan (panj10)

November 09, 2017

Contents

List of Tables

List of Figures

1 Introduction

1.1 Overview

The Tower Defense game project is a partial re-implementation and refinement based on an existing open source project. The completed project allows the user to build towers on the map to defend enemies from a certain route.

1.2 Context

The Module Guide (MG) document is an introduction to modules in this project, where the module functionality and specification are created based on Software Requirement Specification (SRS) document. The MG covers all the anticipated and unlikely changes to functional and non-functional requirements specified in SRS, and provids a modular structure for the entire system. The MG also includes the relationship between modules and the corresponding requirements they achieved. At the same time, a Module Interface Specification (MIS) document is also created in the design documents to provide details on every module which are mentioned in MG.

1.3 Design Principles

Design Principles being used in this project are Information Hiding and Uses Relation. Information hiding ensures that each module design interface will not change even when the module secret changes.

1.4 Document Structure

- Section 1
  (Current Section) A brief review of the project and the introduction of Module Guide Document.
- Section 2
  List anticipated and unlikely Changes from Software Requirement Specification document.
- Section 3
  Give details for the Module Hierarchy, provide the module hierarchy by secrets.
- Section 4
  Give explanation for the Connection Between Requirements and Design, which shows how the software requirements are accomplished by the listed modules.
- Section 5
  Provide the Module Decomposition details, including module names, secrets, service or responsibility, and implemented by information for each module listed above.
- Section 6
  Provide the Traceability Matrices which connect the requirements to the modules, and anticipated changes from Section 2 to the modules. This provides a straightforward representation of how the requirements in SRS are accomplished.
- Section 7
  Provide the Uses Hierarchy for the project, which shows the uses relations between modules.
- Section 8
  Provide the Project Schedule by including up-to-date Gantt chart.
- Section 9
  List the Major Revision History for the document.

2 Anticipated and Unlikely Changes

2.1 Anticipated Changes

AC1: The game start menu and looking of interface.
AC2: The addition of Java gaming library.
AC3: The visual representation of tower and critters.
AC4: The deletion of map editor feature.
AC5: The logic and property of critters and towers for balancing modification.
AC6: The addition of AreaAttack tower.
AC7: The addition of animation effect and background music.
AC8: How wave are grouped into stages with different maps and winning strategies.

2.2 Unlikely Changes

UC1: Integration of swing and Slick2D windows.
UC2: The running environment and devices.
UC3: The algorithm for target selection.
UC4: The integration of map editor and stage selection option.

3 Module Hierarchy

Level 1	Level 2
Hardware Hiding Module	MouseAndKeyboardHandler
Behaviour Hiding Module	Game Module GameController Module ArtistSwing Module Gameclock Module Critter Module Tower Module MapTile Module Player Module GameApplicationFrame Module GameState Module MainMenu Module MenuApplicationFrame Module
Software Decision Module	Point Module IStrategy Module Farthest Module Closest Module Strongest Module Weakest Module SetupClass Module

Table 1: Module Hierarchy

4 Connection Between Requirements and Design

The modules are designed into 5 categories, including controllers, helpers, models, strategies, and views. Game module is the main module where the running request is called, and Game Controller module calls every required frames and panels to form the game window. Modules under models category are abstract data types which store the properties of players, towers, critters, maps, etc. These data can be accessed by ArtistSwing module to convert text data into graphical representations, and then the game panels can use graphical representations to display the game to the users on GUI. All the user inputs are done by mouse and keyboard, and they are collected by MouseAndKeyboardHandler module.

In order to enhance the gameplay experience and to facilitate development process, an external library called slick2D is used in this project. It provides a better way of implementing a clean and unambiguous user interface. Also, buttons, dropdown menus and sliders are included in the game controller window for easier user operation. All the buttons and operation areas are properly sized, and the game window is set in high resolution so that it satisfies the usability requirement. The project can be exported into a jar file. This means it could run on any platform that support Java. It satisfies the requirement where the game is supposed to be cross-platform. The usage of licensed library and open sourced code ensures that the legal requirement is not violated.

5 Module Decomposition

M1: Game Module
M2: Game Controller Module
M3: ArtistSwing Module
M4: Gameclock Module
M5: MouseAndKeyboardHandler Module
M6: Critter Module
M7: Tower Module
M8: MapTile Module
M9: Player Module
M10: Point Module
M11: Closest Module
M12: Farthest Module
M13: IStrategy Module
M14: Strongest Module
M15: Weakest Module
M16: GameApplicationFrame Module
M17: GameState Module
M18: MainMenu Module
M19: MenuApplicationFrame Module
M20: SetupClass Module

5.1 Game Module

Secret: Where and how the game panel is loaded.
Services: This module serves as the initiator of the game project. It starts the application, and sets the resolution to 1000*700.
Implemented By: Java Library

5.2 Game Controller Module

Secrets: Data on map, towers and critters.
Services: Handles all the complex interfacing with map, the towers placed on the map, and critters traversing on shortest path.

Implemented By: Java Library

5.3 ArtistSwing Module

Secret: How the entities are drawn on the panel.
Service: Transform game logic and data into drawable entity information, in order to create graphical representation of game.
Implemented By: Java Libraries

5.4 Gameclock Module

Secret: Drawable entity information.
Service: Determine the refresh rate for redrawing entities, and enable pause function for the game project.
Implemented By: Java Libraries

5.5 MouseAndKeyboardHandler Module

Secret: The user input data.
Service: Provide corresponding functions for mouse operations.
Implemented By: Java Libraries

5.6 Critter Module

Secret: Specific categories and default setting for every critter.
Service: Abstract data type of a critter object.
Implemented By: Java Libraries

5.7 Tower Module

Secret: Specific categories and default setting for every tower.
Service: Abstract data type of a tower object.
Implemented By: Java Libraries

5.8 MapTile Module

Secrets: Usable data from map.
Services: Refers to tiles associated with map.
Implemented By: Java Libraries

5.9 Player Module

Secret: How player information is stored and used in game.
Service: Abstract data type for player to store player information.
Implemented By: Java Libraries

5.10 Point Module

Secret: Where and how points are used.
Service: Simply has an x point, and a y point. Used for positioning
Implemented By: Java Libraries

5.11 Closest Module

Secret: Where and how closest algorithm is used.
Service: Finds the target based on who is closest.
Implemented By: Java Libraries

5.12 Farthest Module

Secret: Where and how farthest algorithm is used.
Service: Finds the target based on who is farthest.
Implemented By: Java Libraries

5.13 IStrategy Module

Secret: How the target is chosen.
Service: Find a target critter.
Implemented By: Java Libraries

5.14 Strongest Module

Secret: Where and how strongest algorithm is used.
Service: Finds the target based on who is strongest.
Implemented By: Java Libraries

5.15 Weakest Module

Secret: Where and how weakest algorithm is used.
Service: Finds the target based on who is weakest.
Implemented By: Java Libraries

5.16 GameApplicationFrame Module

Secret: Detailed data in the frame.
Service: Swing implementation of game window.
Implemented By: Swing

5.17 GameState Module

Secret: Data in game states.
Service: Transition from main menu to game application.
Implemented By: Slick2D

5.18 MainMenu Module

Secret: How main menu is called and used.
Service: Slick2D main menu setup.
Implemented By: Slick2D

5.19 MenuApplicationFrame Module

Secret: Where menu frame is used.
Service: Set default map and all button application.
Implemented By: Slick2D

5.20 SetupClass Module

Secret: How the two windows are linked.
Service: Create link between main menu and game application.
Implemented By: Slick2D

6 Traceability Matrices

6.1 Modules and Requirements

Requirement	Modules
Functional Requirements	
FR1	M16, M17, M18, M19, M20
FR2	M3, M6, M7, M8
FR3	M3, M7, M10, M16
FR4	M2, M9, M17, M18, M19
Non-functional Requirements	
NF1	M17, M18, M19
NF2	M16
NF3	M1
NF4	M2, M4
NF5	M17
NF6	M1
NF7	M1, M3
NF8	M1-M20
NF9	M17, M18, M19, M20

Table 2: Trace Between Requirements and Modules

6.2 Modules and Anticipated Changes

Anticipated Changes	Modules
AC1	M1, M17, M18, M19, M20
AC2	M17, M18, M19, M20
AC3	M3, M6, M7
AC4	M2
AC5	M7, M8
AC6	M7, M8
AC7	M17, M18
AC8	M17

Table 3: Trace Between Requirements and Modules

7 Uses Hierarchy Between Modules

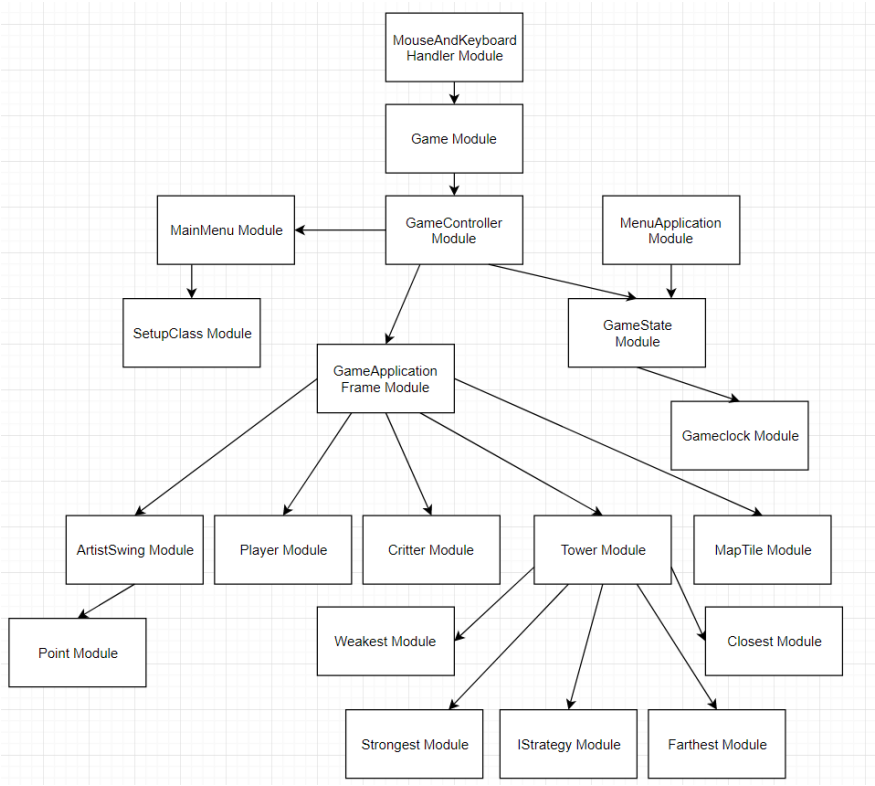

Figure 1: Uses hierarchy between modules

8 Schedule

8.1 Gantt

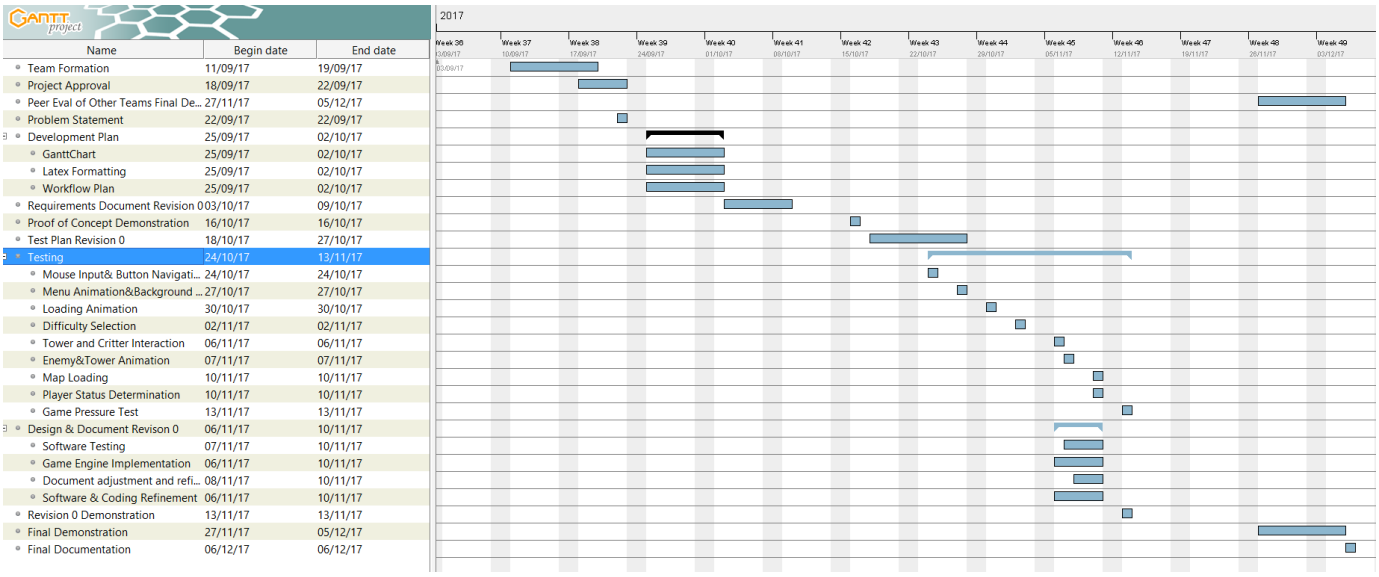

Figure 2: Gantt chart of project schedule

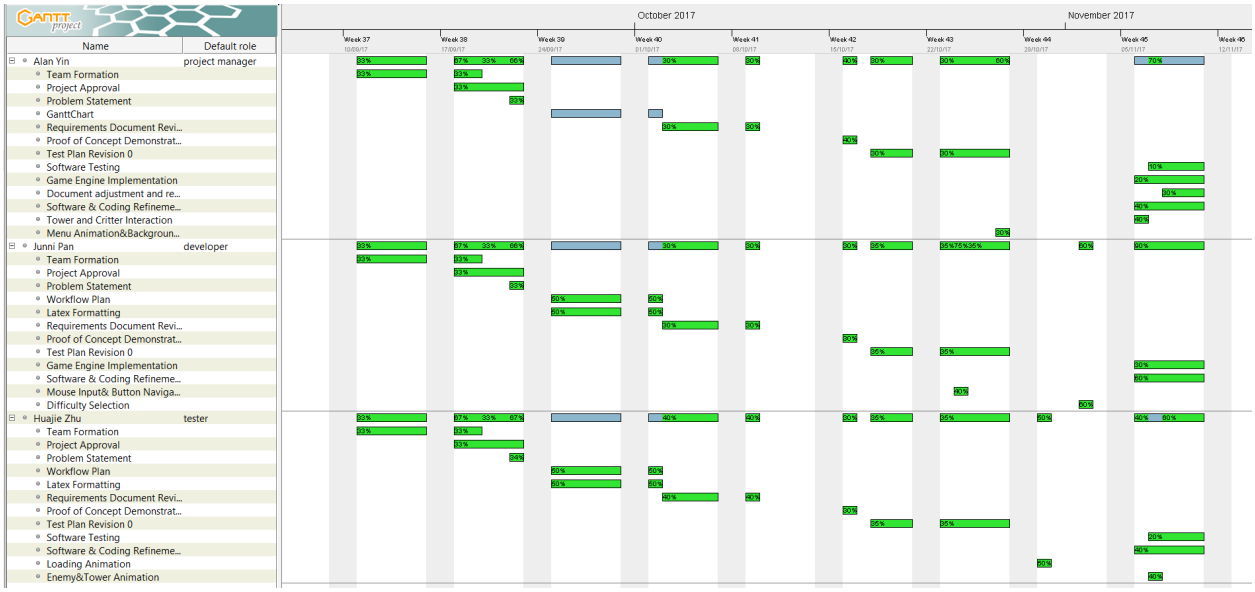

Figure 3: Gantt chart(resources) of project schedule

9 Major Revision History

No revision yet.