SE 3XA3: Design Document
PROJECT TETRIS

Team #11, Team Tetris
Daniel Agostinho agostd
Anthony Chang changa7
Divya Sridhar sridhad

December 7, 2016

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Developer(s)**	**Notes**
11/02/2016	0.0	Anthony Chang	Initial Draft
11/02/2016	0.0	Daniel Agostinho	Initial Draft
11/02/2016	0.0	Divya Sridhar	Initial Draft

1 Introduction

This is the Module Guide (MG) for PROJECT TETRIS. PROJECT TETRIS is a reimplementation of the classic arcade game Tetris. The user moves pieces called Tetraminos into a game board called a "well" and stacks them. When a row of the well is completely filled, that row is cleared and the user gets points. The purpose of the game is to obtain a high score by clearing as many rows as possible without letting the stack of pieces reach the top.

This project was designed with key programming principles in mind. These principles are: information hiding, modularity, and design for change. We divided the project into modules which are independent sections of code that have their own unique purpose. Modules support good programming principles such as information hiding. The modules abstract their information from other modules such that the other modules need not worry about that information. It does not concern them and allows for the modules to be modified independently and changes can be made without affecting the whole system.

The rest of the document is organized as follows. Section 2 lists the anticipated and unlikely changes of the software requirements. Section 3 summarizes the module decomposition that was constructed according to the likely changes. Section 4 specifies the connections between the software requirements and the modules. Section 5 gives a detailed description of the modules. Section 6 includes two traceability matrices. One checks the completeness of the design against the requirements provided in the SRS. The other shows the relation between anticipated changes and the modules. Section 7 describes the use relation between modules.

2 Anticipated and Unlikely Changes

This section lists any possible changes to the system. According to the likeliness of the change, they are classified into two categories: anticipated changes are listed in Section 2.1, and unlikely changes are listed in Section 2.2.

2.1 Anticipated Changes

A design for change paradigm was utilized during our design process for PROJECT TETRIS. The following anticipated changes are design choices that are made to elements hidden within modules and thus are easy to alter without affecting other components of the project.

AC1: The specific hardware on which the software is running.

AC2: The specific Operating System on which the software interfaces with.

AC3: The language of Java (future versions/releases).

2.2 Unlikely Changes

The following design choices are incorporated in multiple components of the system. If a decision should later need to be changed, many different modules will need to be modified. Thus, these design decisions are unlikely to change.

UC1: Input/Output devices (Input: Mouse and/or Keyboard, Output: File and/or Screen).

UC2: The Algorithm for tetromino interaction (setting to well and/or clearing a row).

UC3: Goal of the software: Playable Tetris game.

3 Module Hierarchy

This section provides an overview of the module design. Modules are summarized in a hierarchy decomposed by secrets in Table 4. The modules listed below, which are leaves in the hierarchy tree, are the modules that will actually be implemented.

M1: Hardware-Hiding Module

M2: Output Module

M3: Input Module

M4: Game Logic Module

M5: Game Graphics Module

M6: Input Control Module

Level 1	**Level 2**
Hardware-Hiding Module	
Behaviour-Hiding Module	Output Module Input Module Game Graphics Module Input Control Module
Software Decision Module	Game Logic

Table 2: Module Hierarchy

4 Connection Between Requirements and Design

The design of the system is intended to satisfy the requirements developed in the SRS. In this stage, the system is decomposed into modules. The connection between requirements and modules is listed in Table 3.

5 Module Decomposition

Modules are decomposed according to the principle of "information hiding" while following the MVC software design pattern. The *Secrets* field in a module decomposition is a brief statement of the design decision hidden by the module while the *Services* field specifies *what* the module will do without documenting *how* to do it. For each module, a suggestion for the implementing software is given under the *Implemented By* title. Only the leaf modules in described in the hierarchy will have to be implemented.

5.1 Hardware Hiding Modules (M1)

Secrets: The data structure and algorithm used to implement the virtual hardware.

Services: Serves as a virtual hardware used by the rest of the system. This module provides the interface between the hardware and the software so the system can use it to accept inputs and display outputs.

Implemented By: OS

5.2 Behaviour-Hiding Module

5.2.1 Output Module (M2)

Secrets: The format and structure of any output data.

Services: Converts the graphics created by the application into a viewable output to the user.

Implemented By: PROJECT TETRIS

5.2.2 Input Module (M3)

Secrets: Handles the acception of input.

Services: Allows the reception of inputs from different sources (i.e. keyboard/mouse).

Implemented By: PROJECT TETRIS

5.2.3 Game Graphics Module (M5)

Secrets: Creates and converts the game into a graphical representation.

Services: Takes the logic of the game and transform it into graphical pieces that can be viewed and understandable to the user.

Implemented By: PROJECT TETRIS

5.2.4 Input Control Module (M6)

Secrets: The format and structure of the input data.

Services: Converts the input data into the data structure used by the input parameters module.

Implemented By: PROJECT TETRIS

5.3 Software Decision Module

5.3.1 Game Logic Module (M4)

Secrets: Algorithm that controls the behaviour of the game.

Services: Details rules on tetromino interaction as well as scoring. This can include piece movement, collision, row clearing, and etc.

Implemented By: PROJECT TETRIS

6 Traceability Matrix

This section shows two traceability matrices: between the modules and the requirements and between the modules and the anticipated changes.

Req.	Modules
FR1	M3, M5, M6, M4, M2
FR2	M3, M5, M6, M4, M2
FR3	M3, M5, M6, M4, M2
FR4	M3, M5, M6, M4, M2
FR5	M5, M4, M2
FR6	M4
FR7	M4, M2

Table 3: Trace Between Requirements and Modules

7 Use Hierarchy Between Modules

A	**Uses B**
M6	M2, M3, M4, M5
M5	M2
M6	M3

Table 4: Module Hierarchy