Software Engineering Group Project

Design Specification

|  |  |
| --- | --- |
| Author: | George Cooper [gwcl] |
| Config Ref: | SE\_ |
| Date: | 7th February 2023 |
| Version: | 1.0 |
| Status: | Draft |

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Copyright © Aberystwyth University 2021

CONTENTS

CONTENTS 2

1. Introduction 3

1.1 Purpose of this Document 3

1.2 Scope 3

1.3 Objectives 3

2. Decomposition Description 3

2.1 Programs in system 3

2.2 Significant classes in each program 3

2.3 Modules shared between programs 3

2.4 Mapping from requirements to classes 3

3. DEPENDENCY DESCRIPTION 3

3.1 Component Diagrams 4

3.2 Component Diagram for Program 1 4

3.3 Component Diagram for Program 2 4

4. INTERFACE DESCRIPTION 4

4.1 Class 1 interface specification 4

4.2 Class 2 interface specification . . . 4

5. DETAILED DESIGN 4

5.1 Sequence diagrams 4

5.2 Significant algorithms 4

REFERENCES 4

# Introduction.

## Purpose of this Document

The purpose of this document is to outline the design specifications for the Software Engineering Group Project 2022/23.

## Scope

## Objectives

# Decomposition Description

## Programs in system

## Significant classes in each program

|  |  |
| --- | --- |
| Class Name | Description |
| Setup | Setup will handle all methods required to setup the game for the user. The class will handle creating a new game or restoring a previous game. The class will build the board using JavaFX, handle usernames and colour choice. The class will also handle restoring a game if the program crashed. |
| Game | Game will handle all systems required to run the chess match. These include piece movement, detecting checks and piece removal. |
| Main | Main will call the program. |
| Quit | Quit will handle all functions relating to saving and exciting the program. The class will handle storing all moves in the game to allow for the user to restore and step through the game in its entirety. |
| Replay | Replay will handle loading and displaying previously played games. It will hand systems required to allow the players to step through previously played games. |
| Piece | Piece will act as a superclass to all specific piece classes. It will include attribute shared by all chess pieces. |
| Pawn | Pawn is a subclass of Piece. It will act as a blueprint for all pawn pieces in the game. The class will store pawn specific attributes such as legal moves, model and point worth. |
| Knight | Knight is a subclass of Piece. It will act as a blueprint for all knight pieces in the game. The class will store knight specific attributes such as legal moves, model and point worth. |
| Bishop | Bishop is a subclass of Piece. It will act as a blueprint for all bishop pieces in the game. The class will store bishop specific attributes such as legal moves, model and point worth. |
| Rook | Rook is a subclass of Piece. It will act as a blueprint for all rook pieces in the game. The class will store rook specific attributes such as legal moves, model and point worth. |
| Queen | Queen is a subclass of Piece. It will act as a blueprint for all queen pieces in the game. The class will store knight specific attributes such as legal moves, model and point worth. |
| King | King is a subclass of Piece. It will act as a blueprint for all king pieces in the game. The class will store king specific attributes such as legal moves, model and point worth. |

## Modules shared between programs

## Mapping from requirements to classes

# DEPENDENCY DESCRIPTION

## Component Diagrams

## Component Diagram for Program 1

## Component Diagram for Program 2

# INTERFACE DESCRIPTION

## Class 1 interface specification

## Class 2 interface specification . . .

# DETAILED DESIGN

## Sequence diagrams

## Significant algorithms

Significant data structures

REFERENCES

[1] DOCUMENT HISTORY

| *Version* | *Issue No.* | *Date* | *Changes made to document* | *Changed by* |
| --- | --- | --- | --- | --- |
| 1.0 | N/A | 7/02/23 | N/A - original version | GWC1 |