**Software Requirements and Design Document**

**For**

**Group “Riskier”**

Version 3.0

**Authors**:

Bradley Hodson

Michael Styron

Antonio Vidal

Grayson Wagstaff

Wesley Watkins

# Overview (5 points)

Riskier is a 2-D turn-based command and control game. It acts an improved edition of the classic Risk boardgame, replacing the random elements of combat in the original with a more in-depth turned based combat system which has more emphasis on player skill rather than player luck. Riskier is a standalone game that implements a central board that is sectioned off into multiple regions. Players compete with one another to control these regions by placing and moving soldiers onto the board. When two players come into conflict over a region, play switches over to a turned based combat system. Here, players can select soldiers and issue orders in order to defeat their opponent. Once a battle is resolved, play returns to the main board. The game end when either all players are eliminated, or a single player control the entire board.

# Functional Requirements (10 points)

Blue – High priority

Black – Medium priority

Red – Low priority

1. Title Screen:
   1. When the game is launched, the main title screen must be displayed
   2. New Game:
      1. A “New Game” button must be present on the main title screen
      2. When the button is pressed, a new game of Riskier must begin and the scene must transition to the game board
   3. Options:
      1. An “Options” button must be present on the main title screen
      2. When the button is pressed, a new options menu must be displayed to the player
      3. Difficulty:
         1. Within options, a difficulty menu must be present
         2. The difficulty menu must allow the player to choose between “easy”, “novice”, and “hard” difficulties
            1. The difficulty choice should be made via clicking buttons labelled “easy, “novice”, and “hard”
   4. Quit:
      1. A “quit game” button must be present on the main title screen
      2. When the button is pressed, the game must successfully close itself
2. Game:
   1. Game board:
      1. When “new game” is selected from the title screen, a new game instance is created
         1. The player must be prompted to select a starting region
         2. The AI must choose a starting region after the player
            1. Starting regions must be easily identified and be displayed differently from other controlled regions
            2. If one side captures their opponent’s starting region, that side must win the game
            3. If one side loses their starting region, that side must lose the game
      2. When the player is on the game board screen, the game board must be visibly present
         1. The game board must be presented at the center of the screen
      3. The game board must consist of multiple, separate “regions”
         1. Regions must be easily distinguished from one another with clearly defined shapes and borders
         2. All regions must be connected and accessible
            1. Every region must be connected to at least one other region
            2. Regional connections must be defined as either sharing a border or having a connecting line between them
         3. Any region that has soldiers present on it must display which player owns it via a colored marking
            1. The player’s colored markings must incorporate the color green into them
            2. The AI’s colored markings must incorporate the color red into them
         4. Any region with soldiers present on it must display the number of soldiers present
            1. This numerical display must be on the colored ownership marker
   2. UI:
      1. The game board must have certain UI elements present and visible on the game board screen
      2. A “Reserve Troops” counter must be present in the upper left corner of the screen
         1. The counter is to display the number of soldiers available to the player that can be placed on the board during that turn
         2. The counter must update itself as soldiers are placed on the board during that turn
         3. The counter must update itself at the beginning of every player turn
      3. A “Moves Left” counter must be present in the upper right corner of the game board screen
         1. The counter is to display the number of move orders available to the player during their turn
         2. The counter must update itself as move orders are given during that turn
         3. The counter must update itself at the beginning of every player turn
      4. A “Place Troops” button must be present in the bottom left corner of the game board screen
         1. When pushed, the player should be allowed to place soldiers on any player-controlled region
         2. The player should only be able to place at most as many soldiers as they have available in reserve
      5. A “Move Troops” button must be present in the bottom middle of the game board screen
         1. When pushed, the player should be allowed to move a soldier from a player-controlled region into any adjacent region
         2. The player should only be allowed to make at most as many move orders as they have available in the “Moves Left” counter
      6. An “End Turn” button must be present in the bottom right corner of the game board screen
         1. When pushed, the player’s turn should end and the AI’s turn should begin
      7. A text display must be present in the upper middle of the game board screen
         1. The text display should accurately reflect what is currently happening during a turn
            1. During the player’s turn, the text display should prompt the user for specific actions
            2. During the AI’s turn, the text display should state what the AI is currently doing
         2. The text display should change its contents when appropriate

# Non-functional Requirements (10 points)

*List the* ***non-functional requirements*** *of the system (any requirement referring to a property of the system, such as security, safety, software quality, performance, reliability, etc.) You may provide a brief rationale for any requirement which you feel requires explanation as to how and/or why the requirement was derived.*

# Use Case Diagram (10 points)

*.*

# Class Diagram and/or Sequence Diagrams (15 points)



*.*

# Operating Environment (5 points)

*Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.*

# Assumptions and Dependencies (5 points)

1. Python 3.6
2. Pygame 1.9.6