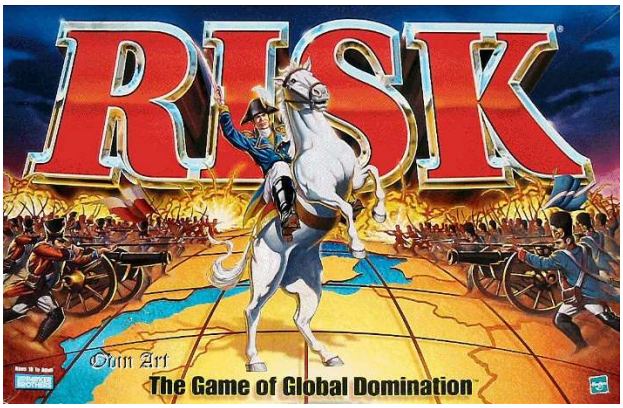
WORLD CONQUEST DESIGN DOCUMET

A Design Document for the Game ‘World Conquest’ for Raffle Games



FEBRUARY 8, 2024

TEAM ONE
University of Sussex

Contents

Introduction & Aims

A collection of all the early & late-stage designs of the digital game "World Conquest" – inspired by the popular board game 'Risk'. This product has been instigated by Raffle Games, a board games company wanting to go into the video game industry via adaptions of their Board Games products.

The aims here are to outline the design process of this project and also as a log of the changes in design over the course of the development process.

Overall Requirements

Here we will outline the basic requirements set out by the customer and what needs to be met

- GUI implementations of any physical Components of the Board Game Risk
  - Board/Map
  - Army (Troops/Canons/Cavalry)
  - Dice (???)
  - Cards

Sprint I

Here are the design, drafts and diagrams for the first sprint. This sprint being to build a very basic MVP where the user is able to attack and reassign troops

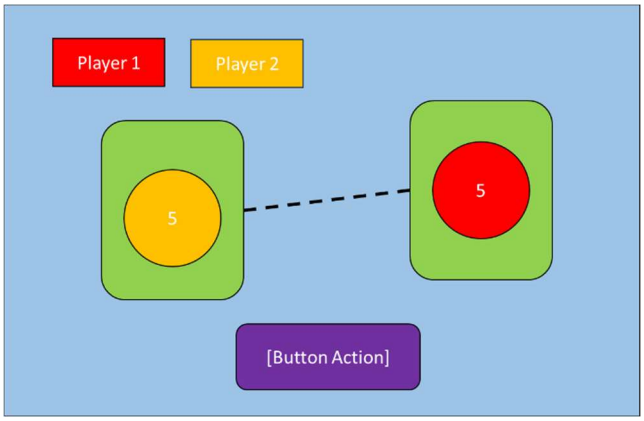
Design Objectives

For this first sprint, we will have:

- A basic UI
- 2 countries
- 2 players
- Win by having all countries occupied by a player's troops (if opponents troops = 0)
- Draft army (troops can be added to player country)
- Attack opposing players (no dice at this point, with a fixed number of attacks and defence)
- Fortify (troops can be moved around to other territories, at this point there is no ability to do so but)
- Mouse control to perform actions
- Changing phases (a)

UI Design

This is the initial UI layout, using our design objectives for this sprint. It is meant to portray all the necessary mechanics we need to make this first prototype work while discovering what is and isn't necessary for future sprints.



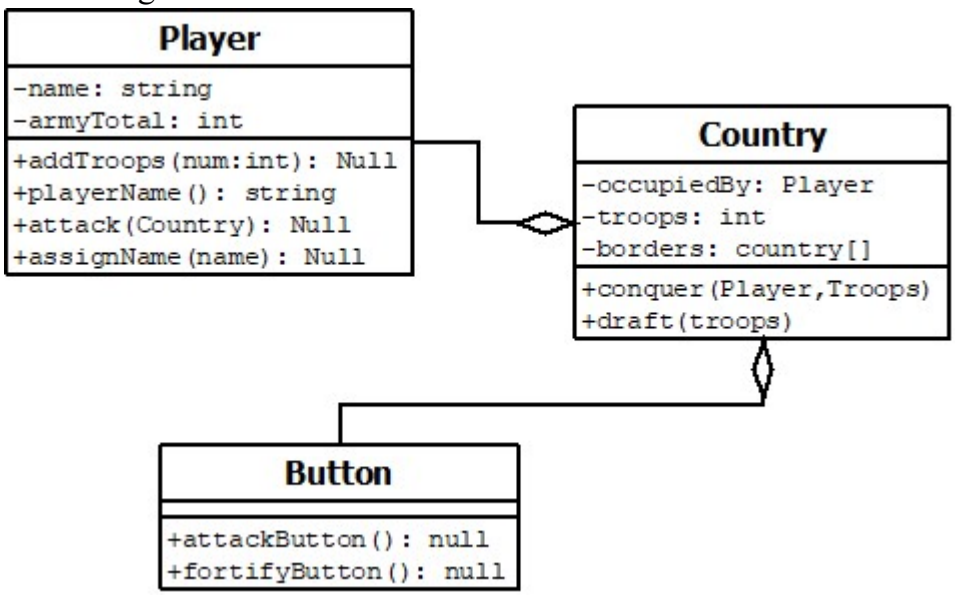
Class table

Class No.	Class Name	Attributes
1	Player	• armyTotal – int • armyAssign - int
2	Territories	• occupiedBy – Player • troops – int • borders – country[]
3	Button	• null
4	gameManger	• players – Player[] • phase – Enum
5	troopController	• country • troops

Method Table

Method	Class No.	Comment
addTroops(int)	1	This will add troops to the player class and will return the
playerName()	1	Will return the player's name
attack(country)	1	Deducts troops from opponents troops (no dice)
assignName(string)	2	Will assign name to player
conquer(player, troops)	2	Will assign occupiedBy and troop number
draft(troops)	2	Adds troops to a player country
attackButton()	3	Player interface so as to choose to move into attack stage
fortifyfButton()	3	Player interface so as to finish fortifying their territories and moving to the next players turn
gameLoop()	4	Loops through the phases of play for each player until a win condition is met

UML Diagrams



The first attempted at the UML diagram had three classes, Player Country and Button. This was changed to be more commensurate with the coders class names and to make more sense when relating to components and the mechanics of the actual board game. A later UML diagram here, better illustrates the final Class structure of sprint ones prototype:

gameManager

```
/**
 //currentPlayer;
 //Player player1;
 //Player player2;
 **/

 // Start is called before the first frame update
 void Start()
 {
     // createPlayer();
 }

 // Update is called once per frame
 void Update()
 {

 }

 /**createPlayers()
     //Player name  = player1
     //Player name  = player2
 **/

 /**gameloop():
     // check win
     // startPlayer turn
     // runPhases
     // end player turn
 **/

 /**runPhases()
     //player.addTroops()
     //player.attack()
     //player.fortify()
 **/
}
```

Player

```
public class player : MonoBehaviour
{
    //string name;
    //int troopCount = 5;
    /**
        public Player(String name)
        {
            this.name = name;
        }
    **/

    // Start is called before the first frame update
    void Start()
    {


    }

    /**addtroops()
        //skips for the moment
    **/


    /**attackcountry()
        //player1 attacks player2
        //for this phase player2 loses troops on each turn
    **/

    /**fortify()
        //for this sprint this phase will just print fortified
    **/

    /**gettroopsTotal()
        //skips for the moment
    **/

    /**getPlayerNamel()
        //skips for the moment
    **/
}
```

Territory

```
{
 //Name
    //Sections = 1
    //troopPerSection = 5;

    // Start is called before the first frame update
    void Start()
    {

    }

    /**setTroops()
        this.troopsPerland = random assigned.
    **/

    /**getTroops()
    **/

    /**getName()

    **/
```

Button

troop controller

```
public class NewBehaviourScript : MonoBehaviour
{
    // Start is called before the first frame update

    //displayTroops
    //modifyTroops
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }
    /**modifytroops
        //subtracts troops(number)
    **/

    /**display()
     //land.getTotaltroops
    **/
}
```

Sprint II

Sprint III

Sprint IV

Conclusion

[Text here is for the developers to read, whether it should done before or after production of the software will be discussed in the meeting]

References

Parker Brothers, 1993. *RISK - The World Conquest Game,* Beverly: Tonka Corporation.