AR3

FYP Proposal for

A Turn Based Strategy Game Focusing on Management of Army Logistics

by

HUI Nathan, LEUNG Ho Man Max, LO Yuk Fai, and WIRAWAN Hubert

AR3

Advised by

Dr. Sunil ARYA

Submitted in partial fulfillment

of the requirements for COMP 4981/CPEG 4901

in the

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

2021-2022

Date of submission: September 17, 2021

Table of Contents

1. Introduction

1.1. Overview

The gaming industry has experienced colossal growth over the course of the pandemic. COVID-19 has prompted social distancing measures globally, causing many to find alternative ways to interact with their friends, such as through online gaming. The tremendous growth of the gaming industry can be attributed to the surge in need for online socialization, as well as extra time being spent at home with computer access during lockdown.

Currently, the net worth of the gaming industry is valued at around 300 billion US dollars, surpassing the combined worth of two other entertainment titans, movies and music [1]. The most popular gaming titles pay out millions of dollars to their best players while massive audiences spectate. Currently, first person shooter (FPS) and multiplayer online battle arena (MOBA) games dominate the esports scene. The top eight esports by prize pool in 2020 were all FPS and MOBA games [2].

A notable genre missing from the top is strategy games. Real-time strategy games such as Starcraft were extremely popular at the dawn of esports. However, the popularity and development of strategy games have decreased dramatically. In recent years, most strategy game releases have only been sequels based on previous titles.

While strategy games are not the most popular game genre anymore, there remains a large, dedicated following for them. Most strategy games revolve around creating an army to defeat opponents, whether by strategic positioning or brute force. However, these army units in such games never run out of ammunition or food. Napoleon said, “An army marches on its stomach.” Although it is understandable that strategy games are designed without as much focus on the logistics of managing an army to reduce complexity, it can make a game feel unrealistic.

In this project, our team endeavours to create a strategy game with an equal emphasis on combat and positioning, as well as management of army logistics.

We feel that this adds a layer of complexity that is not as prominent in other strategy games.

1.2. Objectives

The objective of this project is to create a strategy game with an equal emphasis on combat and positioning, as well as management of army logistics. The project hopes to achieve these goals by:

- **Creating a logistics management system in gameplay**

  The main focus of our project is to incorporate a system that manages a player's army logistics. Thus, the project will create and implement a logistics management system for a player's army that is not overwhelmingly complicated, but adds strategic depth.

- **Creating an attractive strategy game**

  The game should be interesting and fun to play. Ultimately, the purpose of a game is to derive enjoyment and pleasure. An enjoyable new player experience is paramount, so the learning curve should be gradual. Through the design of different systems, the gameplay should reward players with higher strategic skill.

- **Developing multiplayer features**

  Alongside a singleplayer historic mode, our project hopes to include multiplayer functionality. Pitting player against player generates more fun gameplay.

- **Implementing the above objectives into a smooth game**

  The game should run smoothly, be appealing visually, and have fun gameplay, in order to create an enjoyable user experience.

1.3. Literature Survey

As there are already plenty of strategy games on the market, this Literature Survey will explore and examine the games that most influenced our design and provided inspiration for this project.

Advanced Daisenryaku

Developed by Sega in 1991, Advanced Daisenryaku is a turn-based strategy game that features a series of World War II battles on the European Front. Players play as main combatants in Europe such as Nazi Germany, Great Britain and the Soviet Union.

In the game, players have to control all units they have. It is tedious to micro-manage every unit a player controls, especially during the late game. In light of this, an “auto-pilot” system was adopted for this project to overcome the problem.

Advanced Daisenryaku

Hearts of Iron IV

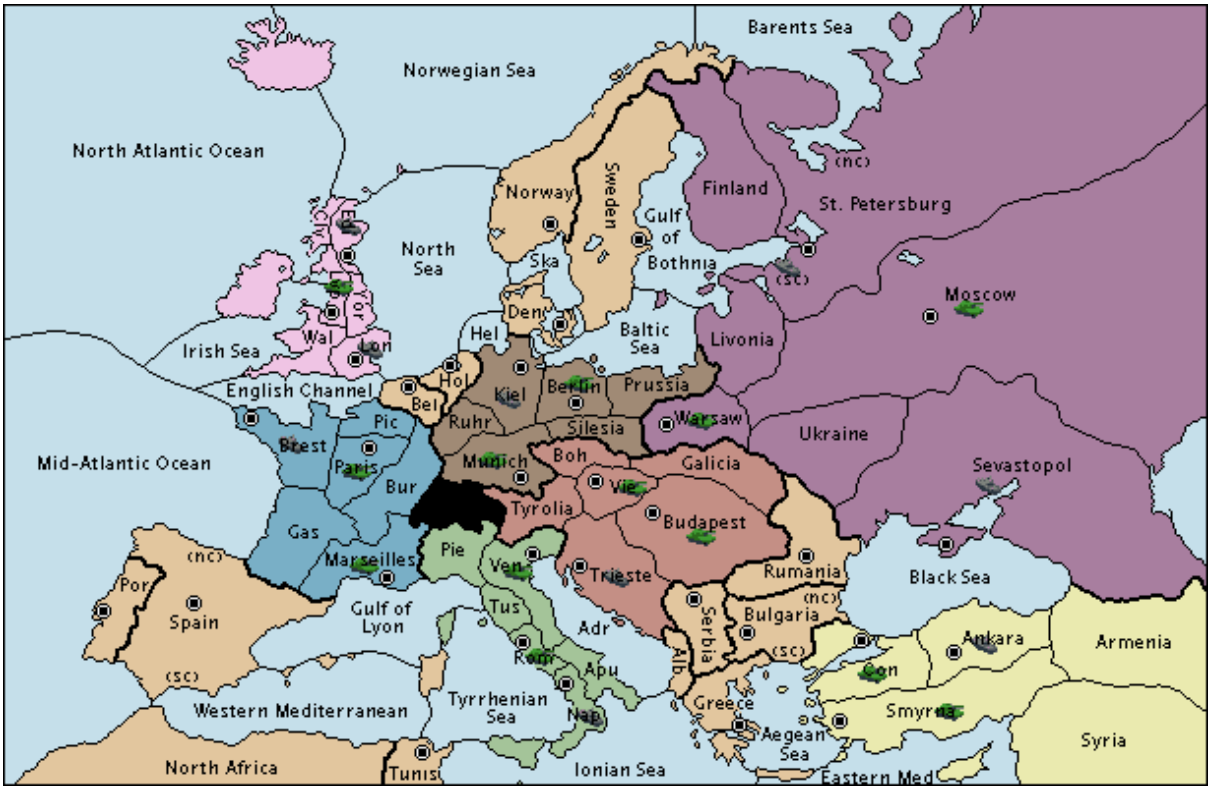
Released in 2016, Hearts of Iron IV is a real-time grand strategy game developed by Paradox Interactive. Players can play as any nation in World War II. The game focuses on various aspects of the War, including military, diplomacy, economy and espionage.

Hearts of Iron IV

Diplomacy

Diplomacy is a strategy board game developed in 1954 by Allan B. Calhamer. Each player represents one of the seven powers in the times prior to World War I. The game focuses on diplomatic relationships instead of military aspects. Players can win the game by negotiating with other players, or even betraying them.

All players take their turns at the same time instead of following a sequence. This is also known as “simultaneous turns”, which can greatly reduce waiting time of all players in each turn.



Diplomacy

Starcraft 2

Starcraft 2 is a fast-paced real-time strategy game developed by Blizzard Entertainment. Unlike other games mentioned in this Literature Survey, Starcraft 2 is played in real time. Real-time strategy games are known more for high mechanical skill and intensity, and less on the actual strategies and tactics.

Since the high mechanical intensity of real-time strategy games can take away from the strategic aspect of the game, a turn-based gameplay format was chosen for the project.

Starcraft 2

2. Methodology

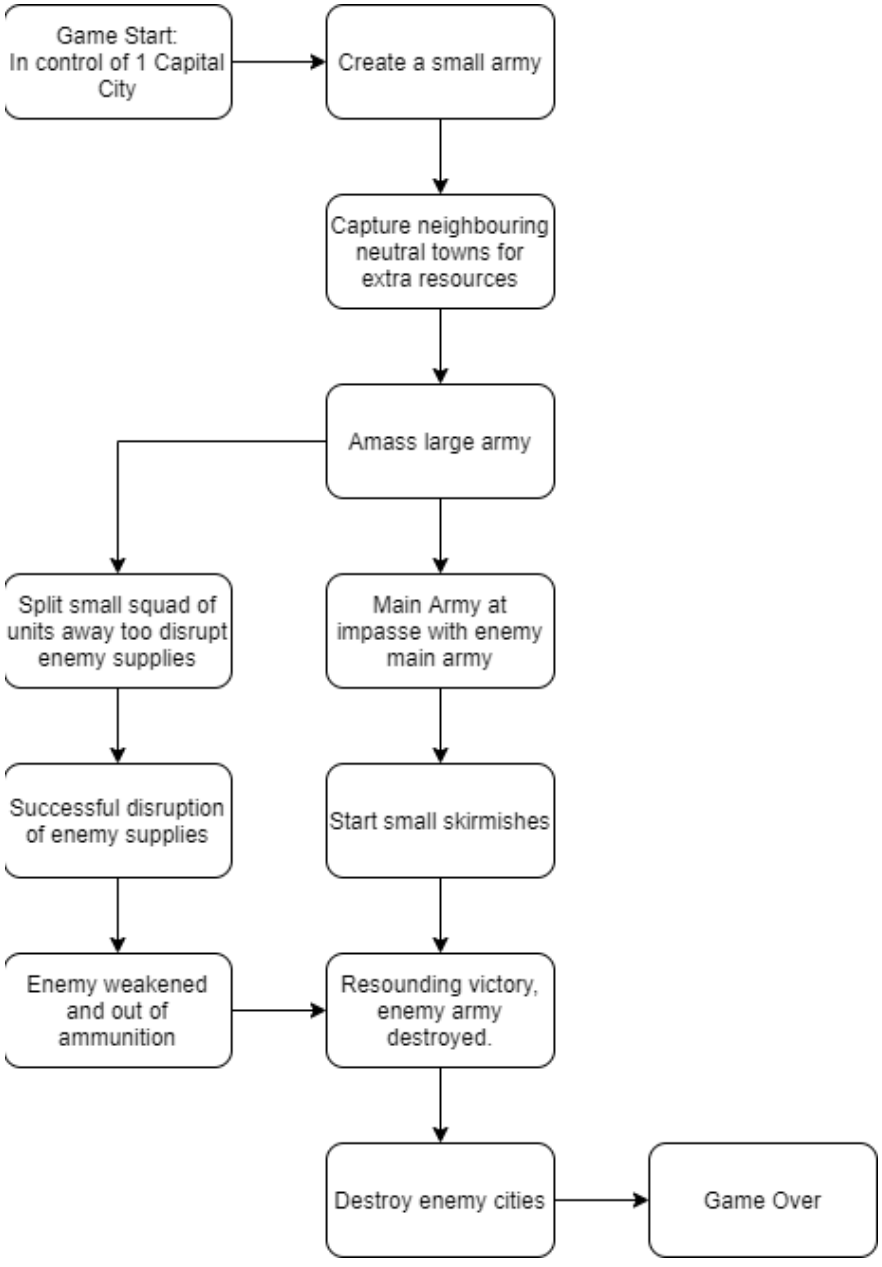
2.1. Design

Our project consists of 3 main parts: Gameplay, User Interface, and Network

Gameplay

Each game will have the same objective: to destroy the enemy's cities. The game will take place on a hexagonal game board.

A sample flowchart for the gameflow of a typical game is shown below.



Our game will also include systems to manage resources, logistics, combat, etc. These systems are currently under development and testing and will be further fleshed out.

Resource Systems

Resources will be automatically generated every turn, based on the amount of cities and villages a player owns.

Logistics Systems

Players can flip between 2 viewing modes, logistics mode and combat mode. In logistics mode, the player can see and manage supply routes, resource collection, and the construction of infrastructure such as unit training facilities.

Unit Creation Systems

Construction of unit training facilities will allow players to train different units, depending on which training facilities were built. Building a unit will consume resources and take time to train, both increasing based on the strength of the unit.

Combat Systems

Each turn, players can designate units to either move, shoot, or ambush. A unit cannot execute more than one of these actions per turn.

A unit ordered to move will move to a designated location, limited by its speed.

A unit ordered to shoot will shoot a target it can see. Shooting at a target moving will halve the damage dealt. Shooting at a target moving out of a unit's range will further halve the damage.

Ambush mode takes a turn to set up, during which a unit will hide. Damage dealt from hiding occurs first. This can allow hidden units to deal lethal damage to an enemy before they can even return fire.

Terrain Systems

The terrain on the map will be randomly generated with Perlin noise. Terrain elements include: plains, forests, swampland, hills, mountains, rivers, and other bodies of water. Some terrain will be difficult for units to traverse, reducing the distance a unit can travel through it. Other terrain will be impossible for units to pass through, unless otherwise specified.

Weather Systems

Every few turns, a season change will occur, causing terrain to have different properties. For example, rivers may freeze and swampland may become extra boggy.

User Interface (UI)

The user interface is an essential part of the game, as it allows players to interact with the game. Creating an understandable and effective UI is critical to the user experience. Various iterations of the UI will be designed as the project proceeds.

Network

In order to allow for multiplayer games, a multiplayer framework must be established. In this project, Unity MLAPI will be used. Unity MLAPI is a mid-level networking library created for Unity that allows programmers to abstract networking [3].

2.2. Implementation

The project is divided into 3 parts for the ease of implementation. They are assets, game mechanisms and integration. Assets include all resources related to our game, namely modules of tiles and units, music and sound effects, and User Interface designs. Game mechanisms include, but are not limited to, all systems mentioned in part 2.1. Integration oversees the merging of the above parts.

Assets

Both free modules from sources like Unity Store, and self-drawn modules will be used. We will use Blender to create simple animations of the modules. Blender is a free and open-source software for 3D model creation. It provides functions like modelling, rigging, and animation.

Royalty-free sound effects will also be used. For creation of background music, Cakewalk will be used. Design of a single track of background music is planned, but more can be designed if time allows.

Game mechanisms

Game mechanisms are mainly logic and rules that can be represented by code. As our game will be coded in C#, Visual Studio is chosen as the Integrated Development Environment (IDE) for code compilation and development.

Integration

We will use Unity for the development of our game. As a game engine, Unity supports various rendering operations such as texture compression, mipmaps and various mapping methods. It also includes libraries for audio mixing and multiplayer and networking. These features make integration of assets and scripts much easier.

2.3. Testing

In order to ensure the game's functionality, different tests will be conducted during different phases of the project.

In the development phase, unit tests will be written in C# using an external library, Unity Test Framework (UTF). Unity Test Framework enables writing test code in both Edit Mode and Play Mode in Unity which ensures each module functions properly and reduces the burden of heavy debug work when the project size grows.

After the User Interface and Interactions are developed, another external library, AltUnity Tester, will be used to provide testing on User Interface elements to ensure every visual element functions as designed.

Software tests will be conducted after the game is developed. Play testers will be invited to provide evaluation and bug reports for bug fixes to minimize unexpected glitches.

2.4. Evaluation

Based on the objectives, we must ensure that the functionalities of the game are working as expected in order to maintain a good user experience.

The code will be reviewed in the framework mentioned in 2.3 in order to reduce debug time. We will also review the user interface and design which is one of the important criteria in making the game more enjoyable.

Additionally, user acceptance tests will be conducted to collect feedback from users to make the game more interesting and user friendly. Using the data gathered, analysis on specific components of our project such as the UI, gameplay, and network will allow for feedback-based improvements.

3. Project Planning

3.1. GANTT Chart

Task	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May
Select game engine											
Design game mechanism											
Design Graphical User Interface											
Design game maps											
Design and develop algorithm for bots											
Develop multiplayer mode											
Design background music											
Design and develop map editor											
Design and develop achievement system											
Develop chat function											
Design and develop record & replay function											
Perform unit testing											
Perform user acceptance testing											
Write proposal											
Literature survey											
Write monthly report											

Write progress report											
Write final report											
Prepare for presentation											
Design video trailer											

3.2. Division of work

Task	LEUNG, Ho Man Max	Marcus	Nathan	Hubert
Select game engine	**L**			
Design game mechanism	**L**			
Design Graphical User Interface		**L**		
Design game maps	**L**			
Design and develop algorithm for bots				**L**
Develop multiplayer mode			**L**	
Design background music		**L**		
Design and develop map editor	**L**			
Design and develop achievement system				**L**
Develop chat function			**L**	
Design and develop record & replay function		**L**		
Perform unit testing	**L**			
Perform user acceptance testing	**L**			
Write proposal			**L**	
Literature survey			**L**	
Write monthly report			**L**	
Write progress report			**L**	
Write final report			**L**	
Prepare for presentation	**L**			
Design video trailer		**L**		

*L = Leader, everyone else assisting

4. Hardware and Software Requirements

4.1. Hardware

Hardware	Requirement
Personal Computer	Operating System: Windows 7 (64-bit) or higher

4.2. Software

Software	Version	Usage
Unity	2020.3.17f1 or later	Game development
Visual Studio 2019	16.11.3 or later	Code development
Blender	2.93 LTS or later	Modules design
Cakewalk	Sonar Platinum	BGM design

5. References

[1] C. Kelly, P. Johnson, and S. Schuler, “The new face of gaming,” *Accenture*, 27-Apr-2021. [Online]. Available: https://www.accenture.com/us-en/insights/software-platforms/gaming-the-next-super-platform. [Accessed: 12-Sep-2021].

[2] A. Stern and T. Murray, “Top 10 Esports games of 2020 by total Winnings – archive - the Esports Observer,” *The Esports Observer*, 03-Jan-2021. [Online]. Available: https://archive.esportsobserver.com/top10-games-2020-total-winnings/. [Accessed: 12-Sep-2021].

[3] Unity, “Getting started WITH MLAPI: Unity multiplayer networking,” *Unity*. [Online]. Available: https://docs-multiplayer.unity3d.com/docs/getting-started/about-mlapi/index.html. [Accessed: 17-Sep-2021].

6. Appendix A: Meeting Minutes

6.1. Minutes of the 1st Meeting

Meeting 1 - Sept 4th, 2021

Attending:

- Nathan
- Marcus
- Max
- Hubert

- Today mainly do Division of Labor (3.2)
- Asterisks in 3.2 mean optional features
- Props:
    - Free assets? Make our own? Buy assets?
    - For now use default/free assets
- BGM:
    - Marcus will try to do
- Multiplayer server & network
    - Agreement that it would be very preferable that multiplayer is implemented
- Map
    - Perlin noise algorithm
- Achievements
    - Yes
- Chat function
    - Save till last
- Record & Replay
    - Yes

Development Schedule:

- Testing/Rebalancing/Bug fixes: Early March 2021

Use Max's Server

To do:
GANTT chart, table for 3.2, 4.1, 4.2

Flowchart of game

Tentative date of next meeting: Sept 18th, 2021 9pm

6.2. Minutes of the 2nd Meeting

Meeting 2 - Sept 8th, 2021 with Prof Arya

Attending:

- Max
- Marcus
- Nathan
- Prof Arya

Main Questions regarding:

- Objectives
    - What sort of game we're developing
    - How will it be different (**unique features**)
    - Motivations for developing the game
    - Inspiration from different games, in what way we're inspired and what features we want to include
- Lit Survey
    - Comparing similar products
    - Comparing weaknesses and strengths
    - Go into more detail
    - The more we explore the better
    - its flexible
    - Can read both papers and look at other products
- Methodology

- Make sure the game is interesting and fun to play
- But also needs to be a little technical, uses graphics, ai, collision detection etc. (something more advanced, requires original thought)

Fog of war

Maybe group units to move them

Researching during a battle may not seem historically accurate

Historical mode - upgrade during battle

Multiplayer mode - certain number of points to put into upgrades, cannot change during match

6.3. Minutes of the 3rd Meeting

Meeting 3 - Sept 13rd, 2021

Attending:

- Max
- Hubert
- Marcus
- Nathan

Things discussed:

- Game Mechanics
    - How to win: Destroy all enemy bases (starting capital + cities/villages)
    - Logistics
        - Has seperate layer (view)
        - Supplied from outposts
        - Player chooses roads (draws them)
        - Going over different terrain makes it slower (3,2,1)
        - Enemy routes are hidden
        - Killing convoys lets you restock

        - Logistics units running around
        - Can build infrastructure for transport units
            - Roads (trucks)
            - Rails (trains)
        - Maybe have map generate some road parts
        - Maybe logistics units can go over rough terrain, some supply lost, takes longer
    - Unit Resource Consumption (supply)
        - Resource consumption based on tiles travelled, but even not moving takes supply
        - If a unit is on a transport, it does not consume supply
    - Resources
        - Occurs when a player owns Capitals/Cities/Villages
        - Resources added each turn

        - Can construct resource buildings to increase production (marginal return is less)
        - Types of resources
            - Ammo: Produced from munitions factory
                - Can be later split into cartridges/shells
            - (discuss later)
    - Weather system
        - Seasons
        - Terrain affected (eg. rivers freezing over, muddy swamps making it harder to travel)
- Overall flow of the game
    - Start with a capital
    - (discuss later)
- Combat
    - Each unit can either move, fire, or hide(ambush mode)
    - Ambush mode
        - Takes a turn to set up
        - When dealing damage, you deal damage first
        - Moving out of ambush your range is halved
    - If a unit has range 2, and is shooting at a moving target:
        - If the moving target is moving to a tile still in range, damage will be lessened (25%?)
        - If the moving target is moving to a tile not in range, damage will be further lessened (50%?)
- Objective of Project
- Methodology (Part 2)
    - Design
        - Describe structure of game
        - Game flowchart
    - Implementation
        - Classifying what we're going to do into small parts
        - Explain what we're going to do for each part and what tools
    - Testing
        - Content:
            - Mechanisms to test
        - Self testing
        - Automated tests?

        - Send to friends
    - Evaluation
- Hardware and Software Requirements

Things to do by Wednesday:

Max: Lit Survey (1.3) for the first 3, Implementation (2.2), Hardware and Software (4.1, 4.2) Requirements

Nathan: Finish Overview (1.1), Objectives (1.2), add Starcraft inspiration to 1.3, Design (2.1)

Marcus: Testing (2.3)

Hubert: Evaluation (2.4)