Bilkent University

Department of Computer Engineering

Senior Design Project

Project Specification Report

Project Name: Espionage

Project Group Members:

Özgür Öney	21101821
Ahmet Safa Kayhan	21001532
Selçuk Gülcan	21101231
Ateş Balcı	21202771
Fırat Özbay	21201683

Supervisor: Can Alkan

Project Website: espionage-game.github.io

Project Specification Report
October 5, 2015
This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Project CS491/2

Contents

1. Introduction

1.1 Description

Video games that get involved our lives in pre 60's, from these day to nowadays confront us as not only a basic show business element but also software that is used by people from every age and worth millions and billions worth market share. Such software that plough through software world with the help of devices called game consoles at the very beginning became very popular just after personal computers proliferation such that there is at least one PC in every single house.

Espionage, basically is an two dimensional (2D) stealth-arcade game which has character left into a map to go from one point to another with the help of supplied inventory and where users directs this character. On his way to the end, computer-controlled units that have own adaptive intelligence attempts to fail player whereas player attempts to complete the game by reacting with reflexive and strategic moves.

1.2 Constraints

1.2.1 Implementation Constraints

- ✓ The video game will be implemented for PC.
- ✓ Adaptive artificial intelligence (AAI) algorithms will also be implemented for project.
- ✓ Unity game engine will be used for the game.
- ✓ Visual implementations of the game will be done in C#.

✓ GitHub will be used as version control tool.

✓ Object oriented design principles will be applied for whole project.

1.2.2 Economic Constraints

As project team, we will need to pay for domain name “www.espionage.com” to advertise and share our video game. In addition to that, we will have to pay for the GitHub private repository for sustainability of implementation. However, video games itself will be free to play, which means that people who wants to play the game will not be forced to pay money to start playing. For future expands, there would be some administration cost to maintain organization in the process. To advertise the product, of course there might be some advertisement costs such as money to pay to internet web-sites, couple of known game experts and screening. For testing, it’s planned to have no cost since game will be working on PC and we have our own.

1.2.3 Ethical Constraints

Since game is free to play, sharing data among users of the game is not prohibited as long as users do not intend to modify source code of the game.

1.2.4 Sustainability Constraints

In order to apply efficiency and usability to video game, there should be sufficient number of players to involve in development phase to increase customer satisfaction. Advertisement of the project will be done through online channels, such as popular game review channels in YouTube or known websites that stands for gaming sector.

We will have our project in just for PC environment. Necessary modifications to game will be supplied with changing versions of major operating systems, especially when major changes are added environment.

A feedback system will be provided to the users/players of the game so that they can rate gameplay, graphics, non-playable unit's AI and difficulty of the game. By using such system, we guarantee that game will be shaped in same direction of player's will. In addition to that, revising gameplay elements that ML algorithms will be continuously checked to prevent any bug or defect.

1.2.5 Social Constraints

Our video game is designed for entertaining people like most of video games created. Game will be served on the internet as free to play and so downloadable and playable to all internet users.

1.2.6 Technical Constraints

Game will use AI and ML algorithms as described above and hereby our main goal in technical part of the game is keeping this sharp. Artificial intelligence algorithms will be managing the non-playable characters that user-controlled character fights and thereby our goal is facing player with most realistic enemies. Machine learning algorithms, however, will be used to adjust the difficulty level of the game and so as project team, we aim to match player with equally talented non-playable characters as much as possible.

1.2.7 Language Constraints

Video game will be serviced to players in English, Turkish, German and Italian to increase popularity of the game in world-wide as much as possible. Also, just after game is on the market, additional language support will be provided in parallel to user's wish.

1.2.8 Legal Constraints

There would be a legal action by the third parties, but in the very beginning, we have not yet decided whole legal actions as well as enforcements.

1.3 Professional & Ethical Issues

Adaptive artificial intelligence algorithms have key roles to assess and define the taxonomy of the game. Actuality, up-to-datedness and compatibility of the game with the performance constraints is indeed a technological issue and such technological aspect impose a professional issue.

2. Requirements

2.1 Functional Requirements

- ✓ The game will be implemented with C#.
- ✓ The game will be controlled by mouse and keyboard.
- ✓ The game will be displayed on a visual display and will be in preferred resolution.
- ✓ The game will feature music and sound effects; however these will not be crucial for gameplay.
- ✓ Speakers or headphones are an optional user requirement.

- ✓ The game must have a title screen with buttons that allow navigation to the game screen, instructions screen and credits screen.
- ✓ The game must have an instructions screen.
- ✓ The user can navigate from the instructions screen back to the title screen.
- ✓ The game will have a credits screen.
- ✓ The user can navigate from the credits screen back to the title screen.
- ✓ The game will feature ‘next level’ screens that appear between the levels of the game. These screens show the current score. A button allows navigation to the next level.
- ✓ The player character can move up, down, left and right, using the arrow keys.
- ✓ The game will have more than one level. Each level must be completed within a time limit or the game will end. Time limit changes from one level to another and also determined with help of Machine Learning algorithms.
- ✓ The player character must achieve assigned mission and reach the trapdoor within the time limit to proceed to the next level. The player can pick of Easter Eggs hidden in the game.
- ✓ Easter Eggs includes different character outfits, hidden messages audial or visual add-ons to title screen.
- ✓ The game ends when the player character completes the last level.
- ✓ Scoring in the game is based on how much time spent to complete a level.
- ✓ The current score and level are displayed on the game screen.

- ✓ The game will feature appropriate sound effects. A sound will play when the player neutralize an enemy. A sound will play when the player uses an inventory item. A 'victory' sound will play when the player successfully completes a level.
- ✓ Appropriate music will play throughout the game.

2.2 Non-Functional Requirements

- ✓ The minimum frame rate (FPS) must be 40-50 per second.
- ✓ Average frame rate (FPS) must be approximately 60 per second.
- ✓ The average response time between click and reaction must be less than 0.5 seconds. The maximum response time between click and reaction must be two seconds. Adding some simple classes and methods that will compute and display the time needed to process any operation can test this requirements.
- ✓ The game must be able to run with minimum 1024 megabytes of RAM.
- ✓ The game must run in Windows. To verify this requirement, installing the game into appropriate environment and check whether it works properly or not would be sufficient.
- ✓ Code written must be maintainable. This can be achieved by collecting metrics, such as DIT (depth in inheritance tree), MPC (message-passing coupling), WMC (weighted method complexity) and DAC (data abstraction coupling). Also adding documentation will improve the maintainability scale of the system.