Bilkent Üniversitesi

Bilkent University

Department of Computer Engineering

CS491 Senior Design Project

High Level Design Report

Project Name: Espionagé

Project Group Members

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

Supervisor: Can Alkan

Jury Members: Uğur Güdükbay, Öznur Taştan

Project Website: espionage-game.github.io

Project Analysis Report November 2, 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

1.0 Introduction

1.1 Purpose of the system

Video games are simply defined as computer programs that are created for one or more purpose, especially for entertainment and education. From 70's to nowadays, video game industry that contains development, advertisement and distribution of the games has grown approximately 1500 percent, become a 21.53 billion dollar industry and become a world-wide admired industrial fact. According to Entertainment Software Association report published on 2014, 59% of people live in the United States play video games and there is an average of two games in each game playing United States household. Again as in said in report, the average game player is 31 years old, which shows us that gaming industry addresses nearly all states of age in society. Therefore, it would be appropriate to say that gaming industry has scope that aims nearly all people in the society –no matter how old people are and what is their social status. Additionally, 51.9% of the games that are played are in genre of shooter and action games.

Espionage is a 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. Map, that player will follow to finish the particular level is not constant structure of course; it will be designed as layered and will be looking like a lateral section of a building. In addition to its lateral section structure, it will contain elements that have different characteristics; such as boxes will be set down to hide behind them, stairs will be there to elevate between floors/layers, vent holes to pass through enemies without getting attention etc. When we take all of this information given just one paragraph above into account, Espionage appeals to approximately half of the gamers in the world with its unique environment and basically aim to entertain its potential customers as much as possible.

1.2 Design Goals

- **Robustness:** Game should be implemented in a way that user/player will not suffer from the algorithms that adjust difficulty of the game. Since average playing time gives strong clues about games playability and such input is directly proportional to robustness of the game, implementation of algorithms will be done heedfully & in scientific manner (by considering scientific articles published for example).

- **User Friendliness:** Players begin to experience the game from very beginning, with opening screen and buttons located there. Hereby, to increase the sympathy of game from the beginning, all components of both home & intermediary screens will be designed in friendly manner.

- **Performance:** Nobody wants to play a game that always crashes, stops executing or freezes no matter how much they enjoy playing it. Therefore, from graphical user interface part to complex machine learning and artificial intelligence algorithms, game will be implemented by considering performance output of code segments.

- **Understandability/intelligibility:** Though game executes some complex algorithms on the background, what we give to players is as much as they can understand. Rules in the game; usage of the items, movement of playable and non-playable characters, some basic rules to finish a level and any other will be easily understood by a player, independently from his/her age.

1.3 Definitions, acronyms and abbreviations

AI: Artificial Intelligence

ML: Machine Learning

MVC: Model-view-controller

DX9: Nineth version of Direct X ©, last number indicates the version, + after that indicates newer version of software.

OS: Operating System

DB: Database

2.0 Proposed Software Architecture

2.1 Overview

In our game, we used three-tier software architectural layer and MVC(Model-View-Controller) architecture.

- **Three-Tier Architecture**

The most important feature of our game is having an artificial intelligence based system. 3-tier lets us to utilise the advantages of decomposing subsystems in a way that the AI unit will be the core of our game low-coupled and high-coherent with user interface and data persistency layers.

- **Model-View-Controller Architecture**

MVC architecture will be used for seperating what the player will see, and the logic of the game. While the game is being player, the adaptive AI will do most of the job according to the playing intrinsic. While implementing and improving the AI, we also have to be able to alter the view of the game without affecting the logic of the game. So we come up with using this pattern, in order to handle those tasks easily.

2.2 Subsystem Decomposition

We chose three-tier architecture among all design architectures, since it is the most suitable architectural layer that best fits our system structure.

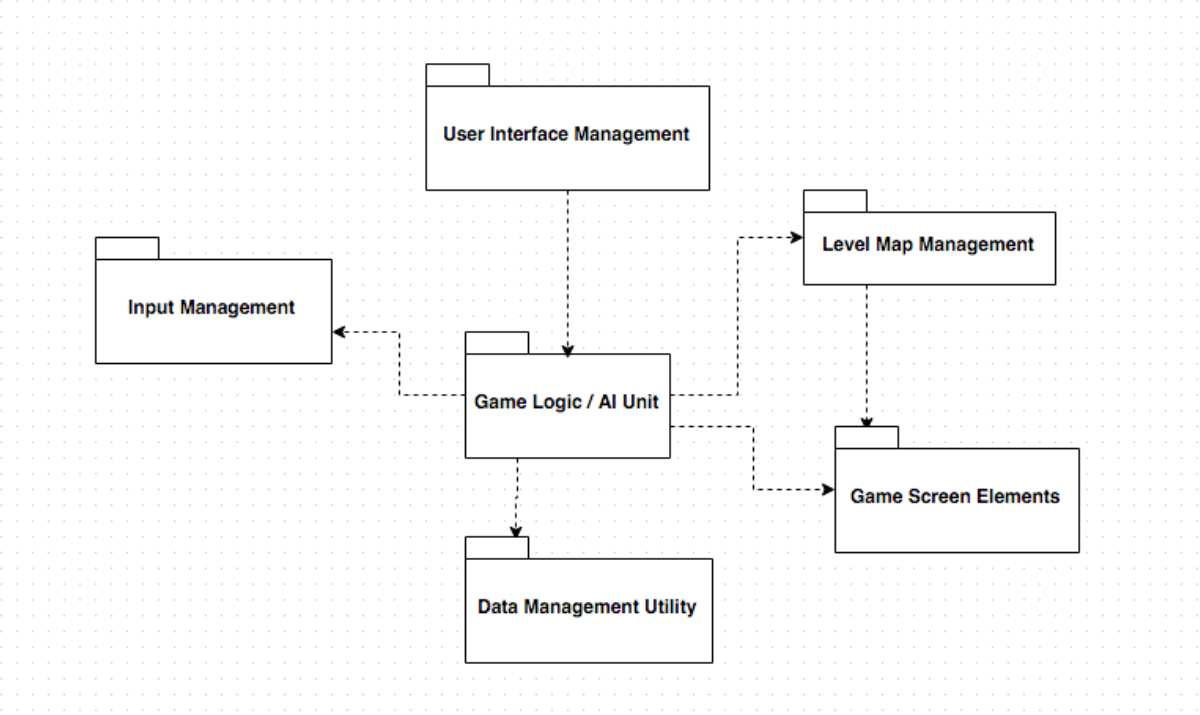

Figure 1: *High Level Representation Of Subsystem Decomposition*

2.3 Hardware/software mapping

We develop our game in Unity environment and our game will be run along with a Unity 3D game engine. Espionagé can be run almost every system, such as Windows XP

SP2+, Mac OS X 10.8+, Ubuntu 12.04+ and Steam OS+. The user should have a graphics card (GPU) that supports DX9 (shader model 2.0) and a CPU that has a SSE2 instruction set support.

In terms of I/O requirements, the game will need a keyboard to be played. We can also plan to add a mouse functionality to our game as we progress.

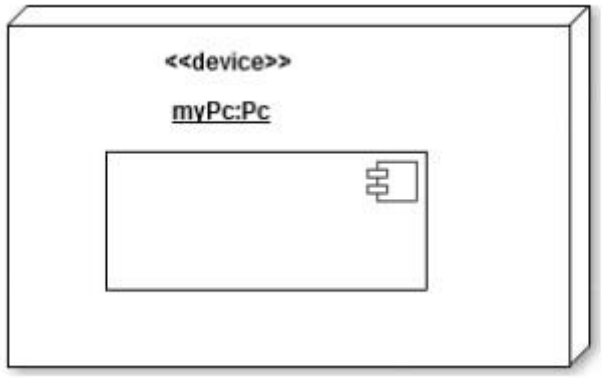

Figure 2*: consists of only one virtual machine which is computer and it has a software whose name is Espionagé.*

2.4 Persistent Data Management

Games includes many persistent data like user preferences, game saves, scores and especially in this project, numbers taken as output from the player to be used in genetic algorithms. Thereby, different applications will be implemented to keep such information. To save the preferences of the player like audio volume, screen resolution and auto-save, a class that is designed in Singleton design pattern that will persist across scenes will be implemented. With the help of serialization of Unity, data taken from this class may easily be used in the future. In addition to that, after a single output is created by the game to measure power of existence of non-playable characters, it will be canalized through a database where we keep the track of every gameplay so that it will be used by genetic algorithms.

2.5 Access, Control & Security

In the game, all users are anonymous which means that the game does not keep any personal (and fragile of course) information about users and so there is no hierarchy among users. Data taken from player's preferences such as nickname or setting preferences will be kept on the device's itself. However, data taken for genetic algorithms will be kept in database, which has any other information except this. Since there is only one type of actor

that interacts with our game (player), we do not have any access control management among actors. All players will be able to reach functionalities, no matter who they are & when they play game. However, we have dynamic access control, which is applied to all users locally. For example, if player is only completed level 1, s/he will be able to access only level 1 and level 2 objects but not level 3 or any other level beyond level 3.

Since the game doesn't keep any fragile information about player that might be corrupted such as credit card information or SSN, we don't have any security protection for player. Game code and resources are also public so we don't have any security precaution for them too.

2.6 Global Software Control

Control flow of the game start with creating a local/domestic account to be used in the currently working system. As new game is started, game logic & artificial intelligence subsystem will take the control to manage the non-playable characters. Thereby, such characters will be created visually –except their drawings, UI subsystem will handle this. However, such trigger on these subsystems will work simultaneously with subsystem that takes input from to user to control his/her character, which is named as Input Management subsystem. In addition to all of these, data tier stands here to collect the data from the player to create an effective genetic algorithm to select which type of non-playables will take role on the next plays.

2.7 Boundary Conditions

- **Initialization Boundary Condition**

Game will be initialized by executing a simple .exe file. Just after that, and installation will meet the player/user and player is required to complete the steps for installation. After such scenario, game will be available on the desktop and ready to be played by clicking on the icon.

- **Termination Boundary Condition**

Quitting game is possible by clicking "Quit Game" button on Main Menu. However, if unexpected situation occurs and game crashes, there will be a pop-up screen that informs people about game's crashing. Also an activity log will record the important game events. If a

fatal exception occurs in game, this activity log will be saved in a file so that player can get help by providing this log file to developers.

3.0 Subsystem Services

Database Management Subsystem: It handles the generic DB operations (adding / deleting comment / post, user registration and adding / deleting post, sending moderation requests) on MongoDB, with (d)encryptions in asymmetric manner.

Game Logic/ AI Subsystem: This subsystem simply handles the control of the enemy & non-playable units in the game. As user-controlled character goes from one side the another, units that s/he met are controlled from this subsystem. System continuosly checks the habits of the player in terms of game progress and behaves accordingly. For example, a player that passes through the map by eliminating enemies with bombs and long shots will face enemies that dressed up anti-bomb suits & kevlar helmets.

Level / Map Subsystem: Subsystem is responsible for the design of the map that player should pass through in each level. As explained in the analysis report, map will consist of boxes to be hidden behind, stairs to be climbed and some air conditioning tunnels to used in passing silently. Hereby, subsystem will design each unique level map in a way that its in increasing difficulty.

Input Management Subsystem: Subsystem is responsible for the management of the input taken from the user. All buttons to be pressed on the keyboards as well as mouse clicks to indicate what to use or what to do are considered as inputs of course and should be arranged in.

UI Subsystem: Such subsystem is responsible for graphical issues from main menu. In menus part, subsystem is responsible for providing basic buttons, radio buttons or checkboxes to be used in basic selections.

Game Screen Elements Subsystem: I gameplay part, responsibility given to the subsystem is graphing the pre-defined game elements including characters, map and enemies as well as animations of moving objects.

4.0 Glossary

Ubuntu: An operating system in Linux environment.

MacOS X: Operating system designed for Apple Inc. © deviceswhich uses UNIX environment.

Windows: Operating system that is created by Microsoft.

Steam OS: Operating system for Steam Box gaming consoles that uses Linux base and is created by Valve Corporation.

Direct X : Collection of application programming interfaces (APIs) for handling tasks related to multimedia, especially game programming and video, on Microsoft platforms.

Unity Game Engine: Cross-platform game engine developed by Unity Technologies and used to develop video games for PC, consoles,mobile devices and websites**.**