



**Bilkent University**

**Department of Computer Engineering**

## **CS491- CS492 Senior Design Project**

**Project Final Report**

**Project Name:** Espionagé

**Project Group Members:**

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

**Supervisor:** Asst. Prof. Dr. Can Alkan

**Jury Members:**

Prof. Dr. Uğur Güdükbay

Asst. Prof. Dr. Öznur Taştan

**Project Website:** [espionage-game.github.io](https://github.com/espionage-game)

<b>1. Introduction.....</b>	<b>4</b>
1.1 Object Design Trade-Offs.....	5
<i>High quality graphics vs. Performance</i> .....	5
<i>Cost vs. Scalability</i> .....	5
<i>Complexity vs. Playability/Usability</i> .....	5
1.2 Interface Documentation Guidelines .....	5
1.3 Engineering Standards .....	6
1.4 Definitions, Acronyms and Abbreviations .....	6
<b>2. Final Architecture &amp; Design.....</b>	<b>7</b>
2.1 Overview.....	7
2.2 Subsystem Decomposition.....	7
2.2.1 Artificial Intelligence (AI) subsystem .....	8
2.2.2 Machine Learning (ML) subsystem.....	10
2.2.3 Enemy subsystem.....	10
2.2.4 Gameplay subsystem .....	11
2.3 Hardware/software mapping.....	12
2.4 Persistent Data Management.....	13
2.5 Access, Control & Security.....	13
2.6 Global Software Control.....	13
2.7 Boundary Conditions .....	14
<b>3.0 Final Status.....</b>	<b>14</b>
3.1 Development Phase.....	14
3.2 Game integrity .....	14
3.3 Test.....	15
<b>4.0 Impact of Engineering Solutions .....</b>	<b>15</b>
4.1 Global Impacts.....	15
4.2 Economic Impacts.....	15
4.3 Impacts on entertainment industry.....	16
4.4 Social Impacts.....	16
<b>5.0 Contemporary Issues.....</b>	<b>17</b>
<b>6.0 New Tools and Techniques Used .....</b>	<b>18</b>
<b>7.0 Software/Hardware Systems.....</b>	<b>19</b>
<b>8.0 User Manual .....</b>	<b>19</b>
8.1 Espionagé activities .....	19
8.2 Unit types.....	20
8.2.1 <i>Main Character: Mr. Herméz</i> .....	20
8.2.3 <i>Enemies</i> .....	22
8.2.4 <i>Laser scanners and turrets</i> .....	22
8.3 Controls.....	22
8.4 Tips .....	24
8.5 Support.....	25

Figure 1: High Level Representation of Subsystem Decomposition.....	7
Figure 2: Detailed component diagram of AI package of the game .....	9
Figure 3: Detailed component diagram of ML package of the game. ....	10
Figure 4: Detailed component diagram of Enemy package of the game.....	11
Figure 5: Detailed component diagram of Gameplay package of the game.....	12
Figure 6: Game virtual machine .....	12

## 1. Introduction

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. 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. Needless to say that, number of elements will differ from level to level with the help of machine learning algorithm described below to change difficulty.

On player's way to the end, computer-controlled units that have own adaptive intelligence attempts to fail him whereas player attempts to complete the game by reacting with reflexive and strategic moves. As character is controlled by player, enemy units that aim to fail player will be controlled by artificial intelligence algorithm, which means that they will also behave in natural ways to give reaction to player's character. For example, whenever character is seen by an enemy unit, such enemy unit will alert all other enemy as well as change their constant formation (such as patrol route or condition of being armed etc.). In addition to that, a machine learning algorithm will detect the player's style by gathering data like time spent, number of tools used or number of enemies neutralized. Such processed data will be used to determine the level of difficulty of course, hereby play a critical role on playability and enjoyment.

Most importantly, a genetic algorithm will be implemented for the game. As the name suggest, a spectrum basic artificial intelligences for every enemy unit will be implemented first and game will be opened to many players to play. As time goes and players play the game to finish, just enemy units that have adequate intelligence will be survived, while others were simply eliminated. To do this, of course a score mechanism for enemy units to detect ones those are able to survive more. At last, a combination of surviving ones will be created to construct a sound basis artificial intelligence.

## 1.1 Object Design Trade-Offs

### High quality graphics vs. Performance

Espionagé game aims to satisfy the potential customer-gamers during gameplay. To do such, graphical content -including character that player control, non-playable enemy units and even objects to interact- of the game should be drawn in high resolution. To satisfy this requirement, a professional help from a graphic designer has taken. However, GPU of a single computer is directly responsible from drawing of elements to the screen and due to high load on drawing, performance problem can occur. Hereby, balance of such trade-off should be achieved through a wise policy to sacrifice neither performance nor quality.

### Cost vs. Scalability

Espionage game keep the data of the users in its community such that it records every single player's level passing time, number of gadgets used and enemies eliminated to optimize the difficulty level of the game. As number of players and number of levels played increases, machine learning algorithm's approximation for difficulty approaches to optimal value. However, keeping track of such attributes simply requires a working database on the background. As pre-defined numbers increase, since expenses are directly proportional, they are also increases. To cope with such trade-off, optimal number of players should be determined.

### Complexity vs. Playability/Usability

Espionage game includes much more traits compared to a basic adventure game: enemy dynamics that uniquely affects level difficulty, gadgets that changes playability of the game from head to toe as well as environmental factors that stands here to determine level structure. As combination of these values are getting complex, playability of the game is dramatically decreases for the player because of the fact that people tend to understand basic combinations much more than complex ones. Hereby, as a trade-off, elements defined above should be spread over the game so that they do not make game difficult to understand and easy-to-play.

## 1.2 Interface Documentation Guidelines

Classes are named with singular nouns, multi worded classes are named as consistent as with the naming convention of Unified Modelling Language. Indentation style is the single-tab

convention. Errors are returned via an exception, not with error codes.

In updating client side data (cache), update is occurred if the data exists, create is occurred if not. Data is checked every night at 12AM for updating machine learning ratios for Hidden Markov Model which will reflect in the entered content by the user.

Methods are named with verb phrases, fields, and parameters with noun phrases. Confidence levels of the users are reflected in the interface through a colour scale, red for not trusted user for example.

### 1.3 Engineering Standards

**UML:** Unified Modelling Language is a standard that is intended to provide a way to visualize the design of a system.

**OpenGL:** Basically a standard maintained by the OpenGL Architecture Review Board (ARB) for rendering 2D and 3D vector graphics.

**IEEE Xplore Standard:** Naming conventions from web service to machine learning algorithm is covered by this standard.

### 1.4 Definitions, Acronyms and Abbreviations

**UML:** Unified Modelling Language.

**SQL:** Structured Query Language.

**DB:** Database

**AI:** Artificial Intelligence

**GPU:** Graphical Processing Unit, a part of computer which is responsible for rendering graphical objects.

**CPU:** Central Processing Unit

**Python:** Object-oriented extensive programming language

**ML:** Machine Learning

**GUI:** Graphical User Interface

**2D:** Two dimensional. Designed in a way such that its only possible to see width and length of a single object.

**SP:** Service Pack, name used for updates by Windows.

**SSE2:** Streaming SIMD Extensions 2, is one of the Intel SIMD processor supplementary instruction sets.

**OS:** Operating systems, which is basically responsible for working of a computer.

## 2. Final Architecture & Design

### 2.1 Overview

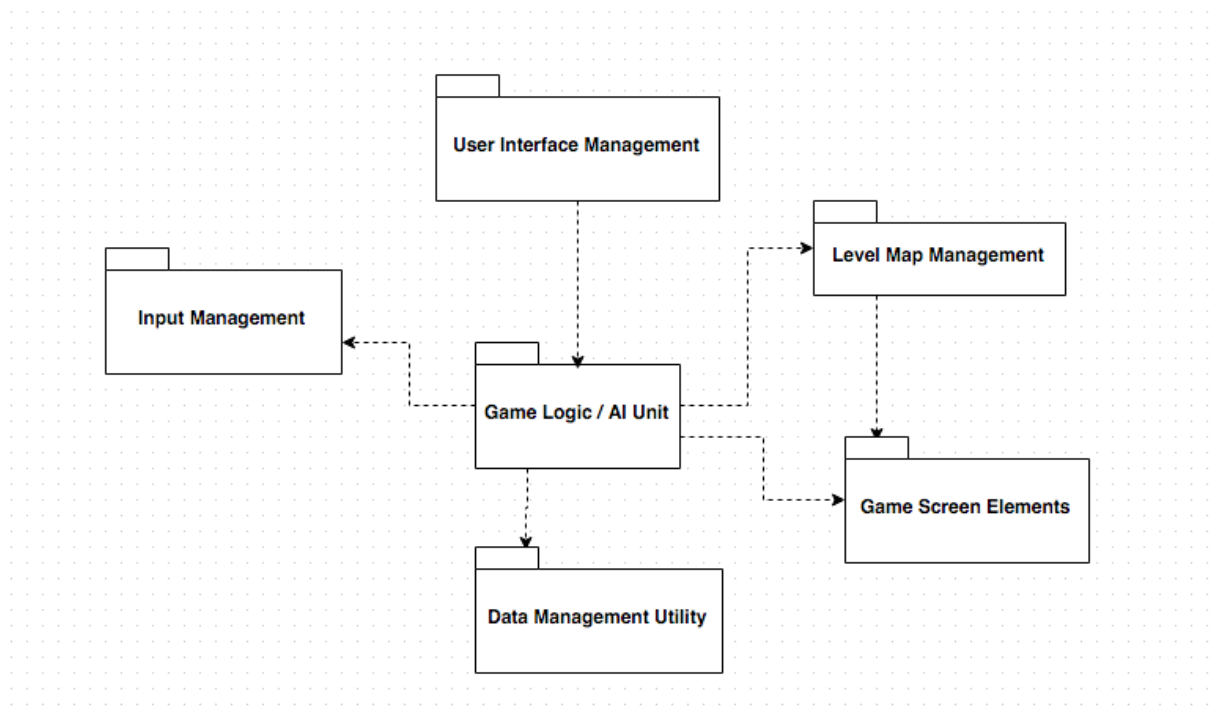
In our game, we used three-tier software architectural layer. Basic explanation will be given below.

- **Three-Tier Architecture**

The most important feature of our game is having an artificial intelligence based system. 3-tier lets us to utilize 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.

### 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*

### 2.2.1 Artificial Intelligence (AI) subsystem

Package that is responsible for every single action of non-controllable enemies.

Explanation of classes in this package is given below.

**Node:** Grids defined in each frame of a level which is used in calculations of possible paths that non-playable enemy unit can follow.

**Graph:** Class that manages interactions contain nodes, such as choosing a route from current location to destination or selecting a place to hide.

**ReactionAI:** Class that keeps algorithms responsible for reactions of non-playable characters.

**State:** Non-playable characters have their unique states depending on the player's interaction with them, such as one enemy should be in alert state whenever our player-controlled character is seen by him. This class herein is responsible for hosting algorithms that determines states of non-playable characters.

**ApproachAction:** Class responsible for guardians'/enemies actions whenever they meet our player controlled character such as going towards source of a unidentified sound.

**ChaseAction:** Class responsible for guardians/enemies whenever they meet a player who is escaping from them.

**FireAction:** Class responsible for guardians/enemies whenever they are forced to use their weapons. In a scenario when player escapes from shoots, non-playable units will be managed in a way that they will shoot in the direction where player escapes to.

**ActionManager:** Class that contains a priority queue for the events so that event mechanism is arranged in a logical order. For example, whenever a series of events occur, their respective drawings and triggers should be materialized in a chronological order.



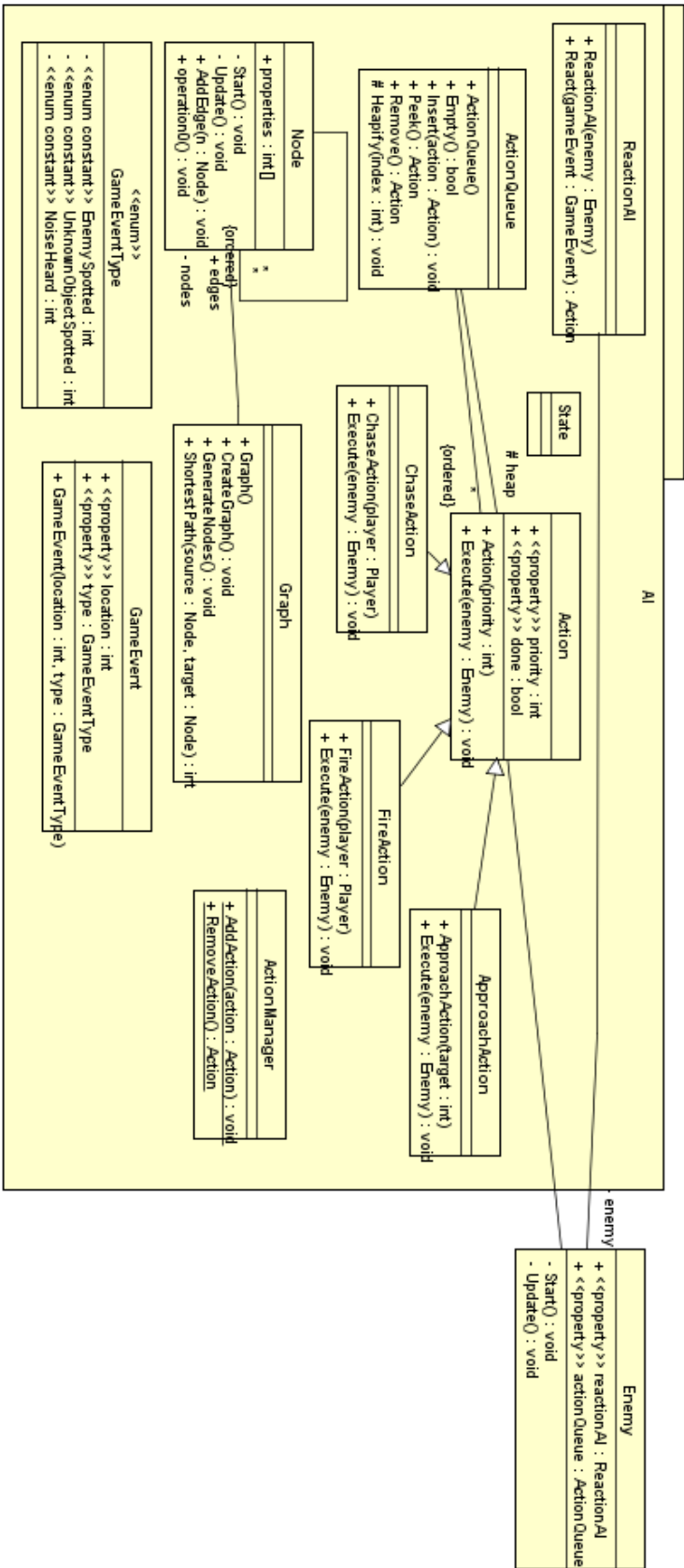


Figure 2: Detailed component diagram of AI package of the game

### 2.2.2 Machine Learning (ML) subsystem

This package contains classes that are responsible for every single Machine learning action. Explanation of classes in this package is given below

**MLCommunicator:** Class required to communicate with machine learning module.

**MLConfig:** Class that contains basic configuration information needed to communicate with ML module.

**MLLevelStats:** Class keeping data of level statistics like time spent, number of enemies eliminated and number of gadgets used during a single level.

**MLLogger:** Class that keeps the log information which will be later used to update MLLevelStats' information.

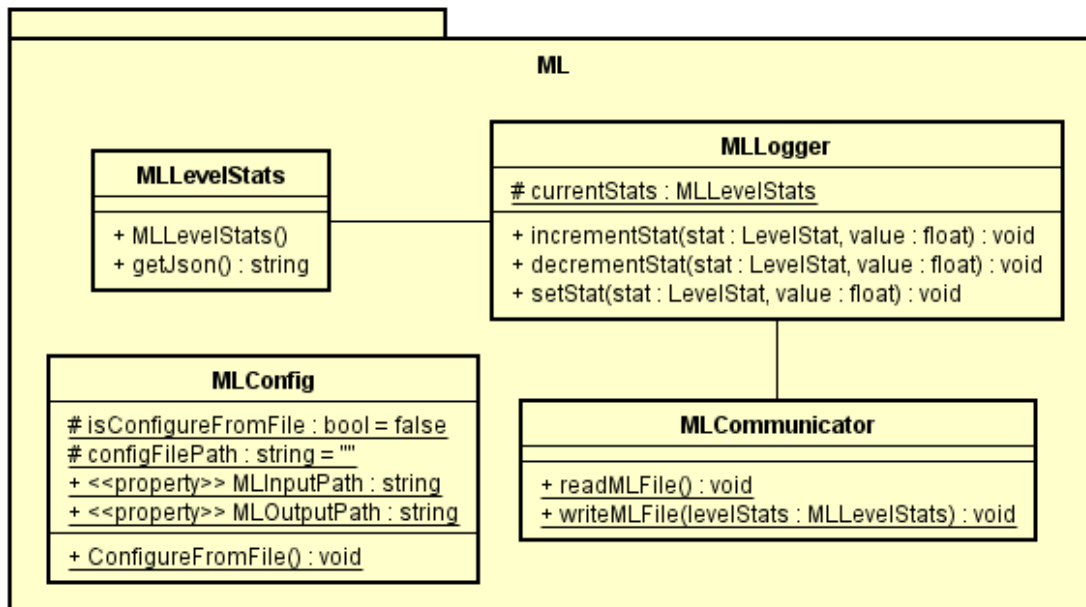


Figure 3: Detailed component diagram of ML package of the game.

### 2.2.3 Enemy subsystem

Enemy package can be seen as a bridge between AI package and game itself. Decisions taken by AI package are implemented/drawn/applied through Enemy package. Therefore, respective classes to connect AI & game each other will be implemented for each AI package class.

**IApproachable:** Interface responsible for executing commands taken from PathingAI class defined above.

**IFirable:** Interface responsible for executing commands taken from ReactionAI class defined above.

**ISpotable:** Interface responsible for executing commands taken from ChaseAction & FireAction classes defined above.

**Enemy:** Class responsible from initializing & updating enemy attributes.

**Turret:** Subclass responsible for actions of turrets, which are immobile enemies that attacks whenever it sees player controlled unit.

**Guardian:** Subclass responsible for actions of shifting enemies that not only attacks whenever they see player controlled unit but also chase him down and warn any other guardian.

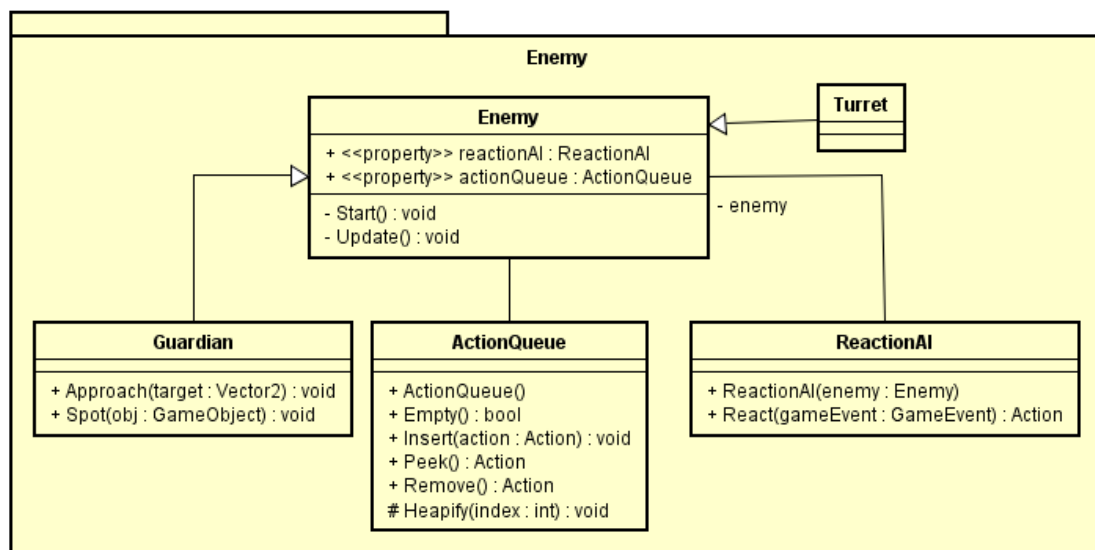


Figure 4: Detailed component diagram of Enemy package of the game.

## 2.2.4 Gameplay subsystem

Package that contains necessary classes for precise player-game interaction.

**Player:** Class that keeps the information about player's visual attributes such as width & length as well as necessary methods/operations to start visualizing.

**Rock:** One of the many gadgets we plan on implementing for the player.

**RockState:** State of the rock whether it hit something or still airborne

**EventManager:** A class which interfaces between gameplay and AI, game adds events, and those events are sent to the AI to be evaluated.

**LevelStat:** An enumeration which holds level specific parameters.

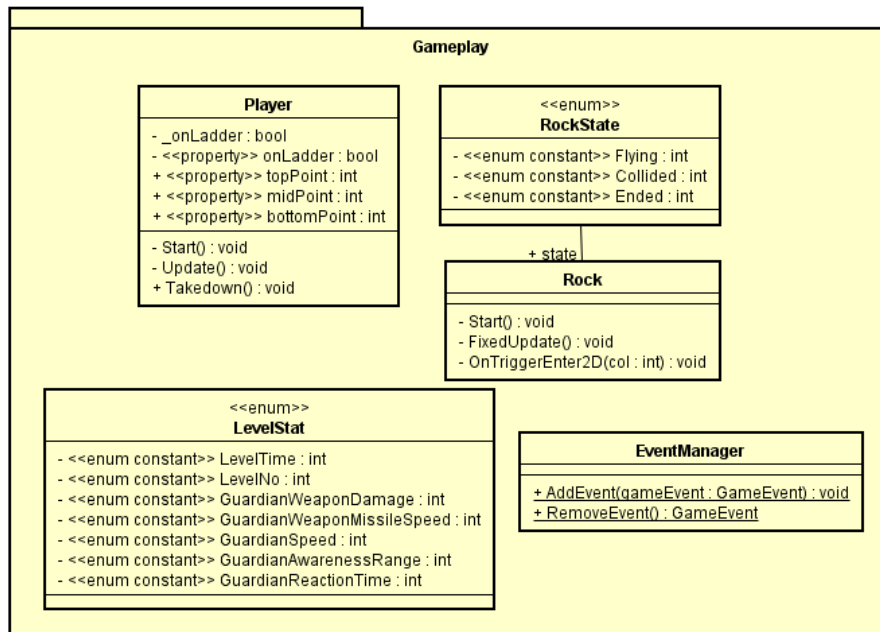


Figure 5: Detailed component diagram of Gameplay package of the game.

## 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.

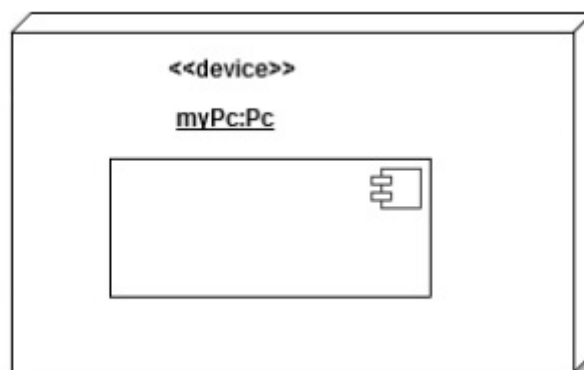


Figure 6: Game virtual machine

## 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 the 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-playable characters 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 Final Status

### 3.1 Development Phase

In development phase, approximately hundred percent of the predicted code is completed. However, since game is open to bugs and some errors can occur in upcoming days, development phase should not be considered as totally completed.

### 3.2 Game integrity

Integration part is simply divided into two parts: integration of external game elements with base code line and integration of back-end side, where user data is collected to implement genetic algorithm. First part, which is integration of external elements; such as soundtrack of menu, sound effects directly involve in gameplay and other visual supplementary (in-game animations, in-game buttons), is fully completed. Though second part, which is the back-end side is available to use and meet the demand for now, it is open to be developed.

### 3.3 Test

At current time; unit, integration and system test are completed. However, since game is a continuous testing environment that would shape with the help of players, test phase is never fully ends.

## 4.0 Impact of Engineering Solutions

### 4.1 Global Impacts

Video games have been changing the way of production and consumption of media, from music to film. Education has been also changed by the gaming culture, because the use of technology inspired by video games helps students and teachers to communicate in new ways.

In the early 80's, increasing number of kids were spending time on consoles influenced by the cartoons such as Pac-Man and Super Mario Bros, and some of these characters have been made cultural icons by them. By means of this, companies started modern making games by appealing demographics inspiring those kinds of statistics. However, in 21th century, what makes a video game modern is its technological substantiality rather than its historical success because a part of people that are to play video games expect that game to be easy on the eye, as well as to be different to the player while it's being played.

Espionage can be the game where it balances modern graphics and latest technology. However, in this era, most of the video games have either really good graphics or a really good game story. Only a few games can be in harmony with respect to that reason. Espionage offers to balance between the most important aspects of what a video game has to have, and utilize them in order to become popular globally.

### 4.2 Economic Impacts

Economic impact of a particular product divided into two pieces in terms of scope: economic impact on individuals and whole industry. According to Entertainment Software Association's report, video games industry grew approximately ten percent every year, which is four fold more than overall economical grow rate over the world<sup>[1]</sup>. Hereby, it would be appropriate to say that Espionage game provides a valuable momentum on both entertainment

and overall industry grow rate in Turkey. In terms of individuals, it is hard to say much about economic impact of a game, therefore Espionage's affect could be considered as absent.

#### **4.3 Impacts on entertainment industry**

As we are in the era of where technology highly impacts the entertainment industry, Espionagé can be considered as a game equipped with technology, so that makes the game attain a place in the market.

The way Espionage entertains people is that, it is a game that can be transformed with respect to the skills of the player which can be considered as a new concept in the game industry. Adaptive AI and machine learning concepts have been recently used by game producers, and the application of those concepts are not that practical in terms of processing the data thoroughly. While the game is being played by a player, s/he will not know how the system is getting the parameters from the gameplay, and this isolation will provide a smooth game play which the player is having fun with.

The interesting thing about the project is that the player won't feel out of it while playing the game, and the game shapes in the direction of his/her playing characteristic. That is the most significant aspect of Espionage, where player would feel the game as if it is produced specific to that person. As each level is being passed, the coherency between player and the game will increase which makes it addictive and entertaining for the player.

#### **4.4 Social Impacts**

Gaming industry has a considerable affect on social life on its own since games are tools that directly interact with human beings. Therefore, it is excepted from Espionage game to effect social life of course.

If we take a closer look on social impacts of Espionage game, we may provide a couple of sub-headings. First, game forces players to think in a creative way to finish a level. Instead of actually controlling the player through a game, time and skills of main character needs to be taken to make the correct decision to guide the main character, so that a time and resource management strategy is gained which can also be used in real-world decision making. Also,



strategy creation can help players become more adaptable and become used to think outside the box.

In addition to that, game's itself has an impact on mainstream application development. When we take a look back to the very early computer systems, we see hard-to-operate text based systems. However, early examples of video games such as Pong or Space Run helped operating systems to come up with a more graphical based interface to make things easier. On the basis of this example, thanks to Espionage's artificial intelligence and machine learning modules as well as genetic algorithm implementation, approach to software development can change a little bit.

## 5.0 Contemporary Issues

Espionagé game brings two different approaches to video game industry: first, game provides a dynamic difficulty control, which has not been implemented with such simplicity yet and second, game uses genetic algorithms to determine best enemies that a single player can meet, which has also not yet been applied into a two dimensional arcade game.

From the perspective of dynamic difficulty control, just a few games released up until now have such feature. However, when we take a closer look, games including this feature (like Diablo, Resident Evil or Fallout series) are usually considered "big games", because of the fact that minimum system requirements are high and of course they are expensive. On the other hand, casual or "just play for fun" games do not care about the increase the playing time, which is directly proportional to user satisfaction comes from difficulty. Espionage game is located between of these two sides; although game cannot be considered "big" in terms of financial and system requirements, user experience is highly valued. Hence, a unit that gathers all gameplay elements, from playing time to number of items used, and use these parameters to optimize gameplay difficulty was implemented. After pre-processing, machine learning algorithms, which uses ensemble methods consist of  $k^{\text{th}}$ -Nearest Neighbour, Support Vector Machine and Classifier, detect difficulty.

However, when it comes to other contemporary issue that is regarding the game, genetic algorithms to determine best enemies that a single player can meet was implemented. Working principle of a genetic algorithm mimics the natural selection process: map is filled with all possible specification combinations of enemies and players are expected to finish the game. Since success criterion of non-playable characters is not letting player to finish the

game; artificial intelligence controlled units will be considered as “successful” whenever they “resist” during a pre-determined time period- which can be defined as a fitness expression of genetic algorithm. During each successive generation, a proportion of the existing artificial intelligence unit population is selected to breed a new generation of artificial intelligence units.

## 6.0 New Tools and Techniques Used

For the Espionagé game, new tools and techniques used can be studied under two headings: machine learning algorithms and genetic algorithms. ML is used for dynamic difficulty adjustment, as mentioned in previous reports and genetic algorithms are implemented to face player with best enemies that they can meet.

One of the challenges that a computer game developer faces when creating a new game is getting the difficulty “right”. Providing a game with an ability to automatically scale the difficulty depending on the current player would make the games more engaging over longer time.<sup>[2]</sup> For the sake of fulfilling this as Espionagé team, we firstly focused on and later implemented a dynamic difficulty adjustment algorithm that can be used as a black box: universal, nonintrusive, and with guarantees on its performance. While there are a few commercial games that boast about having such a system, as well as a few published results on this topic, to the best of our knowledge any of them satisfy all three of these properties.<sup>[3]</sup>

To develop machine learning components, we used Python programming language and we used *ahorasick*, *python-scikit*, *python-language\_tokenizer*, *python\_string\_distance*, *kth-nearest-neighbor*, *Support\_Vector\_Machine* and *classifier* Python libraries. Digitalotion is used to realize data flow from machine learning components to game database.

Second challenging issue in implementation of the game was constructing the bridge between machine learning and artificial intelligence. You can collect the data from the player so that you can learn playing habits of him/her, but managing enemy units in a way that it reflects only what they need to do, but no more. However, when we take already-implemented examples of artificial intelligence modules, we sadly see that using these module as black-box is time consuming. Hence, an implementation of a AI module became a

necessity. For the sake of this situation, we have dedicated a short period of time for reverse engineering and tried to optimize hand-in algorithms in our own black-box module.

To develop artificial intelligence subsystem, we first looked at logic engines that work on the .NET platform since it should be easy to create C# application that interfaces with a one of these. Creating a library from scratch was really hard, so we took help from already implemented libraries and fulfilled some sort of reverse engineering. Libraries FOIL, Lime, Inthelex and Imparo were used for this.

Last but the harsh problem of the project was creating a genetic algorithm base, so only enemy units that have adequate combinations of skills and items should have survived. To do this, we have created as basic database that keep the collection of enemy units, and then just open the floodgate. By considering enemy unit's death rate, as the game is played during a two-week period, weak enemies are eliminated in each step and deleted from the database.

## **7.0 Software/Hardware Systems**

Espionagé game runs 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.

## **8.0 User Manual**

### **8.1 Espionagé activities**

Espionagé is a simple logic adventure game. You will be considered as successful whenever you pass through the map, take script from the hidden room and carry it until the end. However, enemy units do not hesitate to use every single possibility to stop you, with the help of their ammunition and any other helping objects- like lasers and turrets.

## 8.2 Unit types

### 8.2.1 Main Character: Mr. Herméz

*“It is impossible to suffer without making someone pay for it; every complaint already contains revenge.”*

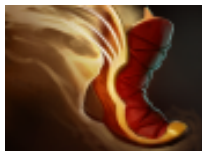
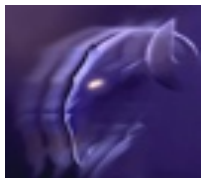
*Friedrich Nietzsche*



Nobody knows what happened to Mr. Herméz, who was working in an iron and steel plant as a regular worker. All we know about him is he got lost just after an unearthly cruelty separated from his family. Seeking for revenge? Curiosity to overpowering any foe? Desire to have speed beyond measure? Hunger for the answer of who and what questions? Maybe all of them, as I said before, we'll never know this. All we know is Mr. Herméz is here and waiting for you to help him through his mission. You can

use skills, which were listed below with their definitions, you can run and you can hide. All you have to do is helping him to get a script that he wants so much, Mr. Herméz will take care of rest.

#### Skills

Velocity		Run faster than bullets, they can't catch you. You'll be gone by the time they come.
Invisibility		Into the shadows. They neither can see you nor catch you for a period of a time.

Berserk		Become a Nordic God, whose stamina and fire power is barely infinite. Of course, this will last for a short time.
Chronos		With the help of Greek myths, now you are able to stop the time. Everything is stopped, you are able to pass through them while they are frozen.
Stone Gaze		If they look you, they are bunch of stones but nothing.
Dionysus		If you were killed? Doesn't really matter. Reincarnate after death.
Hermes		Pass through walls, become a phase shifter.
Cassandra		See the future and past, so that you will be able to develop a strategy before enemies are seen.

### 8.2.3 Enemies

*“We pay a price for everything we get or take in this world.”*

L.M. Montgomery

Enemies are one of the units that withhold Mr. Herméz from taking his revenge. By knowing that Mr. Herméz is alive and right by their side, mission is simple: protect the script from access. However, practice is always different from application, as always. They adapt their items depending on your choices. You select your weapons from range to don't get harmed? They wear armours. You are making too much noise? They use radio communication to warn each other. You activated the alarm? They increase patrol frequency. They are not idiots, as stated, and do not expect them to behave in this way.



### 8.2.4 Laser scanners and turrets

*“Any sufficiently advanced technology is indistinguishable from magic.”*

Arthur C. Clarke



Laser scanners and turrets are tools that can be named as enemies. If you get caught by a laser scanner, red alarm will be activated and warn all enemies alive. By this way, they will increase their patrol frequency and you will be more inclined to get caught of course. However, turrets working principle is more aggressive: if they detect you, they shoot you.

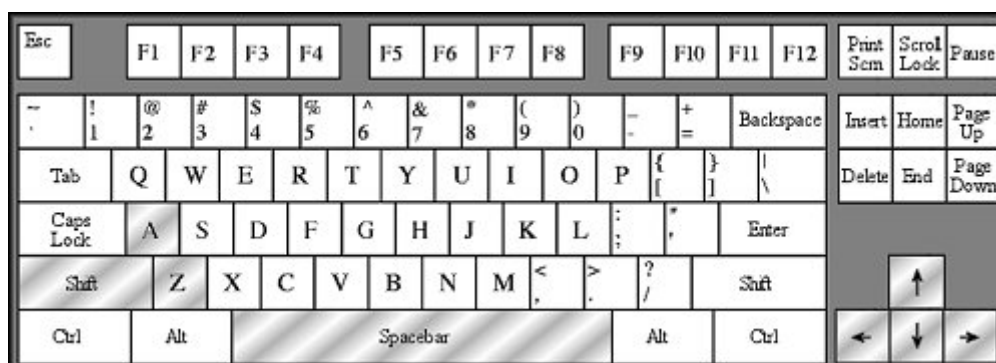
As the levels advance, power of turrets will be increased of course, not to get killed by turrets in early levels does not mean that you can survive later on. Also, laser scanners are level-sensitive too, their frequency and range to be scanned will differ from one level to another.

## 8.3 Controls

### 8.3.1 Controlling the game via keyboard

The keyboard can be used nicely with this game - below are shown the default controls for the keyboard. W-A-S-D quadruplet are used for movement activities while Shift, Spacebar and CTRL keys are used for any other activities needed. In addition to that, previously assigned macros for F keys, such as increasing/decreasing brightness or adjusting volumes, are also valid in game. Lastly, Esc button is used for pausing during the game and opens a new pop-up

screen where player can also quit the game as well as see the current gameplay activities – time, point, enemies eliminated and some stats etc.



### Movement Controls

You can go left or right on the map of course. In addition to that, up and down movement is needed for climbing up on the stairs.

A : Move Left	Left : Move Left	W: Climb up
D : Move Right	Right : Move Right	S: Climb down

### Other Controls

When enemies are patrolling, you cannot just run through them. You have to approach them silently, as ninjas do, so that you have to use Shift key to do this. In ventilation holes, you should crouch. Your interaction with environment is a necessity for both eliminating enemies and fulfilling any other required actions.

CTRL: Crouch	Spacebar: Jump	Mouse 1 : Fire / Eliminate
Shift: Silent walk	E: Use/Interact	Mouse 3: Change Weapon

And of course, who do not want to see what they have? We don't think so much, but in case of you should leave (when real world becomes more important than real-like world), you can use Esc key to perform this.

I: Inventory
Esc: Pause

### 8.3.2 Using A Joystick or Gamepad

In the Options->Controls menu you should be able to setup any joystick or gamepad to control the game. For Dual Analog Gamepads, the best way to to set them up is have one stick handle the movement and one stick handle the rotation. A stick can only handle two axis, so buttons have to be configured to handle forward and backward movement, and rotation around the Z-axis. This is the default setting for analog gamepads.

## 8.4 Tips

Tips are displayed in loading screens of the game, but you'd better read them here also. Tips give you important details and key points of the game, you'd better care them in order to finish the game with a good grade.

As a general rule, unless you activate the “Velocity” skill, you have to take it easy with enemies and lasers. In early levels and easy difficulties, they do not hurt much, but they can even kill you with one critical hit in after stages.

Try to be ungenerous as much as possible while using skills and items. If you spend your ammo before reaching the script, for example, it will be much harder to complete the level since all enemies are already alerted and looking for you.

Keep an eye constantly on the Nearest Target indicator - the red cross shown on the main window. It is hard to watch 6 directions at once, but this target can indicate to you where to look when you need it.

Skills are really important in game flow. You should really well identify what you have to use for specific levels and select wisely among many skill options. Do you need to stop



time? Select Chronos skill. Do you need to be fast so that any bullet can't catch you? Use Velocity. Do you want to be invisible so that any enemies can see you? Use Cap of Invisibility like Perseus. Don't forget, though everything is possible with skills, they have cool down and you can only use limited number of them. Over consumer found no place for repentance, though he sought it carefully with tears.

Do not forget: enemies are not idiots. When one of them sees you, they are all alerted and bear arms. So, try not to seen. Silence is golden.

Lasers and turrets work in droning manner. Do not rush, do not hurry up. Sometimes waiting and observing lasers and turrets can help you more than you expect. Be sure that you don't want to interact with them, they are cruel. But don't forget, if you gaze long into a turret, the turret also gazes into you.

You should find a secret door of a room that contains script and you should get it to finish a level. Do not overlook shadows, shadows contain much more than you can think. A computer that you can interact, a suitcase that you can carry and a script that lays on ground could be a key to finish the game. Details state process.

Time is an important input for pointing algorithm. The much time you spend, the less point you get. The desire of the slothful killeth players, for their hands refuse to labour.

## 8.5 Support

For online support, go to the Espionagé support page:

[www.espionage-game.github.io/support.html](http://www.espionage-game.github.io/support.html)

For any personal question, regarding development phase, process, project, progress or people, please mail to

[contact@espionagethegame.com](mailto:contact@espionagethegame.com)

## 9.0 References

- 1- Entertainment Software Association, 2015 Essential Facts About the Computer and Video Game Industry. Accessed via web, April 18, 2016. Available at <http://www.theesa.com/wp-content/uploads/2015/04/ESA-Essential-Facts-2015.pdf>
- 2- Missura, Olana. “Dynamic Difficulty Adjustment”, January, 2015, Bonn, Germany. Page 7.
- 3- Missura, Olana. “Dynamic Difficulty Adjustment”, January, 2015, Bonn, Germany. Page 7.