# Chapter: 2 Analysis

## 2.1 Introduction to Analysis

2D Game analysis. A systematic examination and evaluation of data or information, by breaking it into its component parts to uncover their interrelationships. Opposite of synthesis. Analysis is the very first process to construct a 2D platform Game. We can also say that analysis is the first step to build or construct any system and to understand what the system could be. Without any clear vision of how the system would be it will be difficult to work later so analysis is necessary. In most of the cases analysis usually starts from problem statements, being an analyst I am going to understand the problem first and do something meaningful in my project.

## 2.2 Analysis Methodology

Analysis methodology is mainly used for gaining the depth knowledge about 2D game and applying data analysis system functionality. There are two type of Analysis Methodology Hard approach and soft approach methodology. Among which I am taking hard approach methodology for the creation of my 2D platform game. In order to take hard approach to a 2D game there are some steps that is needed to take. Steps are listed below

**Problem definition**: Problem definition is mainly about focusing more on answering then on the questions “what is the problem or opportunity?” In systems terms we are saying that there exists a system whose output is demonstrating an unwelcome deviation from an expected performance. This is a problem and the solution involves the restoration of the existing, satisfactory performance. The aim, therefore, of the initial step is to identify and describe the problem or opportunity and obtain agreement from any interested parties that this is what will be addressed.

**Analysis of situation**: It is about defining the current “as is” state and performance level. It is at this point the system boundary is defined in order to decide on “what’s in and what’s out”. The analysis of situation may also require the collection of data and information to quantify the current state and performance level. Iteration with the first step often occurs because the analysis of the existing system nearly always means a redefinition or refinement of the problem or opportunity.

**Identification of objectives and constraints**: It is about defining where we would like to be and the constraints that make affect our ability to achieve the new state. It is an important step because it forces stakeholders to clarify what they hope to achieve, but also to understand the external factors and constraints that will restrict our change choices and therefore the level of change.

**Measures of performance** is about defining measurable means of assessing the efficacy of any definite possibility. It’s really asking and answering the question “how will we know if the change has occurred?”

**Develop Options** is about developing the definite possibilities to the position where they could be implemented. This involves doing sufficient work on each option for technical and other details to be defined, and for costs and benefits to be assessed, while at the same time minimizing the time and resources devoted to the task.

**Evaluation of options** is about evaluating how well each option will work. The objective of this step is to determine whether:

• The option will meet the operational objectives.

• It is technically feasible.

Hard approach is mainly used when working on large business, Complex information

System and other highly bigger-scale business industry. In this project I’ve described about SSADM little and shown Data Flow Diagram. Below table shows how SSADM Works.

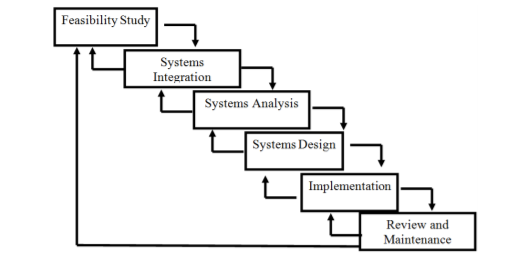


Figure 1: SSADM (Waterfall model)

### **Data flow Diagram (DFD)**

Data Flow Diagrams mainly used for illustrate the functions that an information system performs. It identifies, models and documents how data moves around the system of information. Data flow diagram can be explained in different ways like current physical DFD, Current logical DFD, and Required logical DFD and required physical DFD.

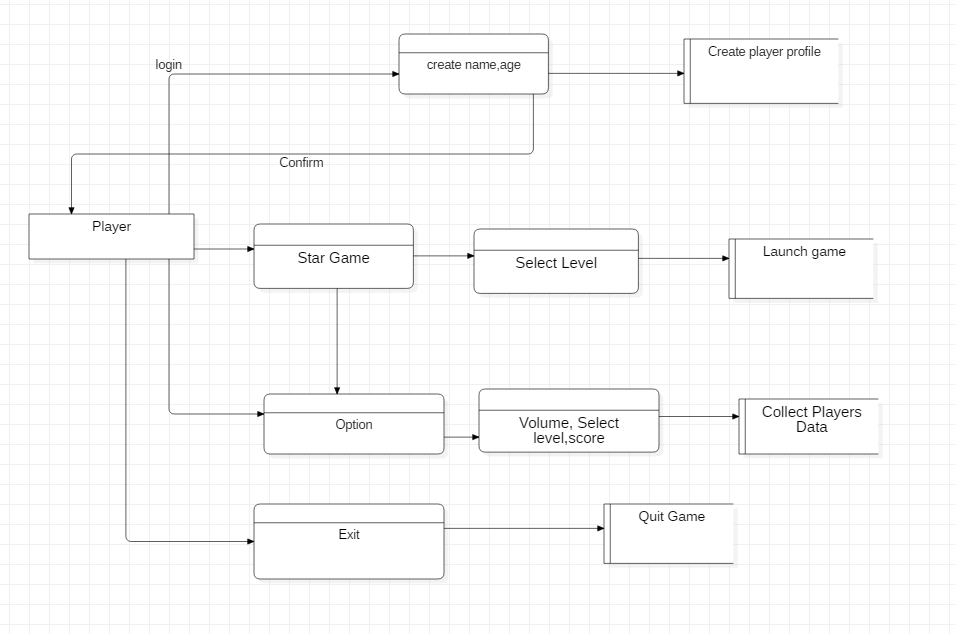


Figure 2: Data Flow Diagram

## 2.3 Feasibility Study

|  |  |  |  |
| --- | --- | --- | --- |
| S.NO | Feasibility Study | What does this study finds out? | How it is related to my project. |
| 1 | Schedule Feasibility | Is there a sufficient or limited time for Project? | WBS, Gantt Chart, Mile Stone. |
| 2 | Economic Feasibility/Cost Benefits | Is my project Cheaper or low of cost? | In Order to Design the game it may require some high graphics and RAM. |
| 3 | Technical/Resources Feasibility | Is there Every Technical and sufficient Resources? | I’ve got my (Dell’s inspiron1500) with 2 GB graphics cards which helps me technically. |
| 4 | Marketing Feasibility | Is there any market value | This types of game have lots of market values for so many countries. |
| 5 | Cultural Feasibility | Does my project effect on a cultural aspect. | Its helps in cultural aspects. |
| 6 | Operational Feasibility | How it will solve 2D game problems. | It is operational |
| 7 | Ethical feasibility | Is this legal for my project. | In ethical aspect it is good. |
| 8 | Comprehensive Feasibility | In which way project may effect on different aspects like cultural, ethical and so on |  |

## 2.4 Requirement Analysis

Requirement analysis is very necessary for the proper development of 2D game

### 2.4.1 Function Requirements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.N | Functional Requirement | Data | Rational | Description | Dependencies |
| FR001 | Menu Bar | New game,  Levels, Load game, Score. | To enter inside a game. | Other game options can only be selected from menu bar. | **N/A** |
| FR002 | Start button | Start, new game | To run a game | Game can only be start from this button. | FR001 |
| FR003 | Sound options | Volume up,  Volume down | To listen the sound of shooting | We can have a sound experience from many game objects. | **N/A** |
| FR004 | Level options | Easy level,  Medium level,  Hard level | To select a different levels | Different levels can be selected with different graphics. | FR001 |
| FR005 | Score board | High score, | To see the shooting score. | We can see our game score data. | FR006 |
| FR006 | High Score board | High score | To see the highest score. | We can see our high score. | FR005 |
| FR007 | Credits | Developer name, Copyright, policies. | To see the game description. |  | **N/A** |
| FR008 | Exit Button | Yes button,  No button. | To close the game. | We can exit the game at any time. | **N/A** |
| FR009 | Save Game | Save game,  Load Game | To save a games, | To save game. | FR010 |
| FR010 | Load Game | Save game | To load a previous game. | To open previous game at any time. | FR009 |

### 2.4.2 Non-Function Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.N | Non-Functional Requirements | Description | Rational | Dependencies |
| NFR01 | Security | System should require a data security. To run game better. | To protect data from different virus and bugs. | FR001 |
| NFR02 | Reliability | System should run smoothly and without any lags. | To make sure that the system is reliable in every aspect. | **N/A** |
| NFR03 | Documentation | System should provide guidelines or tutorials to allow users to know the system better. | To ensure that the user knows about the system. | **N/A** |
| NFR04 | Maintainability | System should be maintainable to any error or bugs of a system | To make sure that system can handle any data’s and informational | **N/A** |
| NFR05 | Capacity | System should run same in any type of situations. | To make sure that game follows concepts of physics and graphic. | **N/A** |
| NFR06 | Availability | System should run very effectively | To make sure game maintain its availability. | **N/A** |

### 2.4.3 Moscow prioritization SRS

Moscow prioritization is a very popular method in order to analyze or managing requirements. In this prioritization method we’ve got four category of initiatives Must-haves, should-haves, could-haves, and will not have at this time. Before going through this method I’ve analyzed and collected all the necessary requirements like functional and non-functional requirements and then worked on it.

**-Functional Requirements (MOSCOW prioritization)**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Title | MOSCOW | Description |
| FR001 | Menu Bar | Must have | Other game options can only be selected from menu bar. |
| FR002 | Start Button | Must have | Game can only be run from this button. |
| FR003 | Sound option | Should Have | We can have a sound experience from many game objects. |
| FR004 | Level Option | Must Have | Different levels can be selected with different graphics. |
| FR005 | Score board | Must Have | We can see our game score data. |
| FR006 | High Score Board | Should Be | We can see our high score. |
| FR007 | Credits | Could Have | It’s not that necessary it’s usually |
| FR008 | Exit Button | Must Have | We can exit the game at any time. |
| FR009 | Save Game | Should Have | To save game. |
| FR010 | Load Game | Should Have | To open previous game at any time. |

**Non Functional Requirements (MOSCOW prioritization)**

|  |  |  |  |
| --- | --- | --- | --- |
| S.N | Non Functional Requirements | Priority | Reasons Based on projects. |
| 1 | Security | Must Have | The project mainly deals with user transactions which needs to be secured. |
| 2 | Reliability | Must Have | Project performance need to be at peak to provide service to more users. |
| 3 | Documentation | Must Have | The project must be legally accepted by the rules and regulations of the country and its people. |
| 4 | Maintainability | Should Have | It can be helpful but does not have to be compulsory. |
| 5 | Capacity | Should Have | Every project will always have bugs, errors and non-functioning features in it. |
| 6 | Availability | Won’t Have | The project is based on game for now but can also be expanded in the future. |

### 2.4.4 Software and hardware requirement analysis.

In order to build a game we need a certain requirements of different hardware and software. In order to make 2D game we may need some powerful PC and good working software. So in the below table I’ve described the requirement.

Pre-project requirements.

|  |  |
| --- | --- |
| **Hardware** | **Software** |
| --Laptop (DELL)  --Core i5(7th Generation)  --RAM 4 GB  --Hard Disk(1024 GB) | --Windows 10 OS(updated)  -- Unity  -- MS office  --SQL Server  --Star UML  --Project liber |

Post project requirements.

|  |  |
| --- | --- |
| **Hardware** | **Software** |
| --Android Devices  --PC, laptop  --RAM 2 GB  --HD 500 GB | --OS windows 7,8 and 10  --Android version 5 or above. |

## 2.5 Use-case Diagram

A use case diagram can be said as a diagrammatic representation of the different entities of a system. In this 2D Game it helps to clarify the role of different external parties on the functionality of the system commonly called actors. The advantage of creating this diagram on my project are as follows.

* It is an easy and understandable method of representing a system to the local people since it doesn’t have any technicality.
* Use cases evolve with each iteration and change in requirements can be traced easily.
* It shows the relationship of these entities with different functionalities.

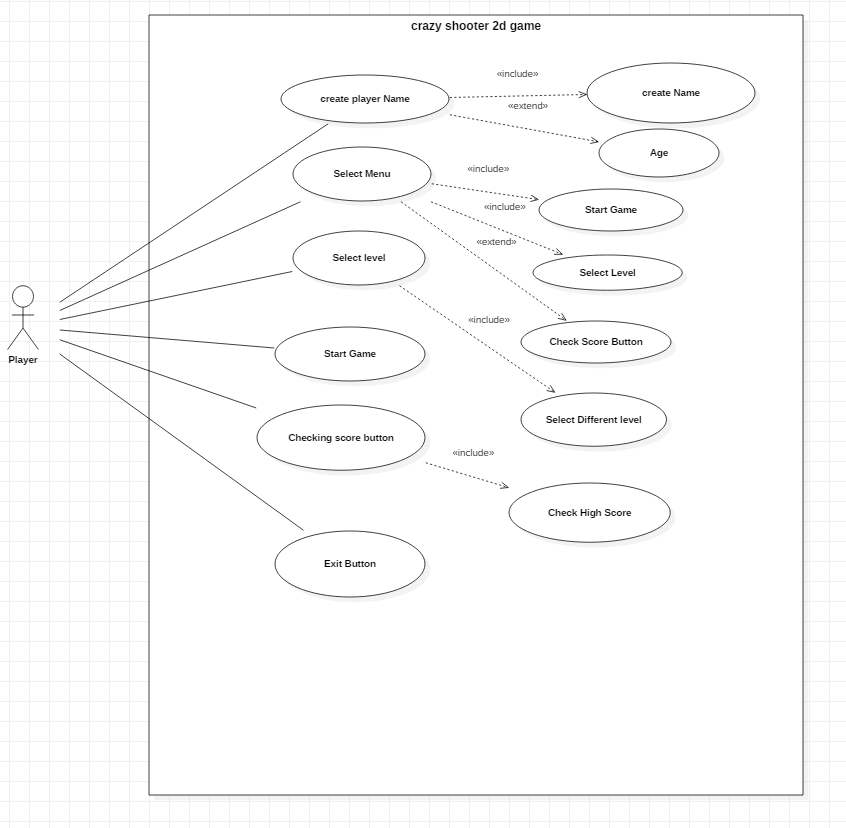


Figure 3: Use case diagram

.

Explanation of a use case diagram.

1. Player Login System.

Before launching the player needs to enter their name and age in order to make an identity of a player. Only after inserting the information of own player can play the game with their name.

1. Menu option.

From this option player will be able to Execute different other option like Select level, Score Checking and credits

1. Select level Option.

From this select level option player will be able to select different levels like easy, medium or hard levels according to their wish.

1. Checking Score.

From this Checking score option player will be able to view how much score they have collected in their game and they can also view their highest score.

1. Exit Button.

This option is used to close every action of the game and go to main windows.

## 2.6 NLA and Initial Diagram

## NLA (Natural Language Analysis).

NLA is the process of static analysis and design. Usually Natural language analysis allows or make us to identify a candidate class and also their attributes. NLA process also helps to clearly identify the system that are incomplete or ambiguous. It is the process of knowing the verbs, adjectives and nouns in the descriptive text. In the case of NLA we have:

--Nouns relates to a potential classes.

--Adjective relates to a potential attributes

--Verbs are related to potential functionality that must be represented.

The candidate class, attributes and methods of my game have been showed below

|  |  |  |
| --- | --- | --- |
| Candidate Classes | Candidate Attributes | Candidate Methods |
| Player  Menu  Score  Volume | Name  Age  Score  Credits | Add player  Add Score  Save Game  Change Level |

## Initial Class Diagram.

A class Diagram is simple graphical representation or the prototype that shows the relationship between different classes, attributes and methods.

Here I’ve used star UML in order to make a class diagram. From the class diagram I’ve shown the classes, attributes and operations in a very precious way. With the help of above Candidate Class, Attributes and methods I’ve created a most efficient class diagram which helps in implementation of a system.

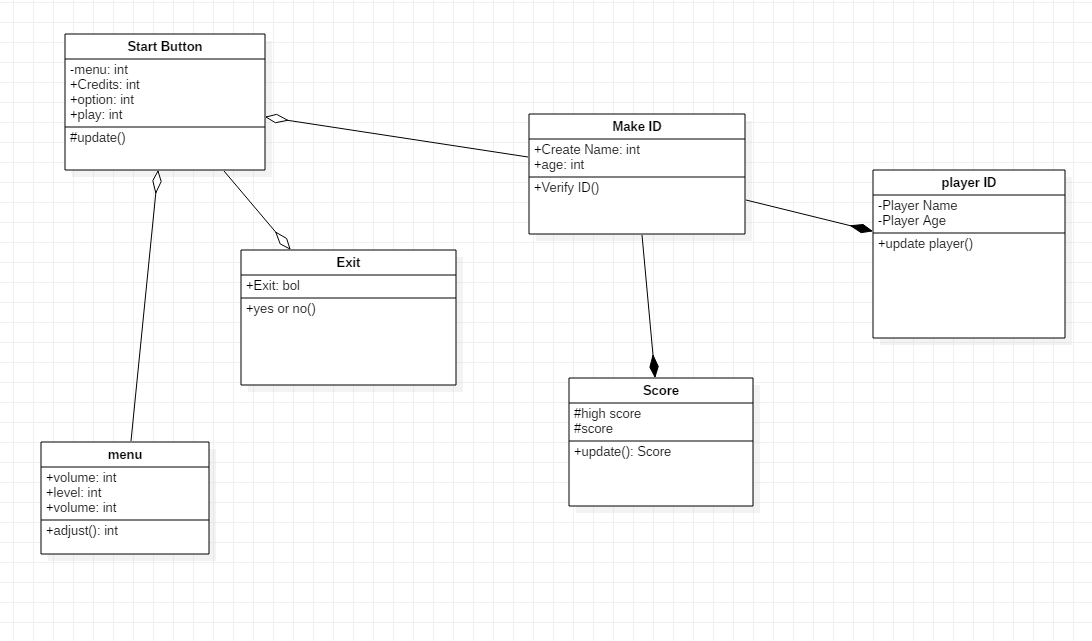


Figure 4:class diagram.

.