Project on

Break the Wall

Roshan Raj Thapaliya

NCC ID: 00163095

Computing Project

Level 5 Diploma in Computing

Softwarica College of IT & E-Commerce

Kathmandu, Nepal

April 23, 2017

Table of Contents

[Analysis specification 3](#_Toc480893692)

[Introduction 3](#_Toc480893693)

[Requirements 3](#_Toc480893694)

[Functional Requirement 3](#_Toc480893695)

[Non-Functional Requirement 5](#_Toc480893696)

[Prioritization 6](#_Toc480893697)

[Use Case 7](#_Toc480893698)

[Architecture 8](#_Toc480893699)

[System Architecture 8](#_Toc480893700)

[Initial Class Diagram 9](#_Toc480893701)

[Conclusion 10](#_Toc480893702)

[References 10](#_Toc480893703)

# Analysis specification

## Introduction

Analysis is the process of determining requirement. It is the first and important steps of software development life cycle. In this process, complex task is break down into small partition to obtain clear idea of the whole project. It is essential to identify about the project, requirement, planning and components to gain certain goal. Breaking down of the whole project into small division helps in unravelling the problems and gather necessary resources for development of the project. Through analysis specification, functional as well as non-functional requirement can be identified which supports to prioritize the requirement. Requirements, Use Cases and Architecture is an important factor to complete analysis process. Requirement are collected at first than Use Case is developed based on collected requirement. At last, system architecture is planned and designed.

## Requirements

Requirements is feature which is included in system. Requirement is important factor in analysis requirement. The main working features is known as Requirements. It helps to determine the system needs and plays important part during the advance of the application. (SearchSoftwareQuality, 2017)

Requirement are gather from different ways. By asking questions about the game such as graphic, levels, sounds in general people. It helps to determine the users concept, like and dislike about game. With the help of gathered information, it is easy to know what types of features users want in the game.

Requirements are categorized into following two categories:

### Functional Requirement

Functional requirement is the logical and functionality of the system. It describes the main work of the system.

Functional Requirement of the system are as follow:

**ID: R1**

Title: Player Registration

Description: Player must register themselves to play the game. They must provide some essentials information.

Rational: To gain user information for login authentication.

Dependencies: R2

**ID: R2**

Title: Player Login

Description: Existing player can login into system with information provided while registration.

Rational: To maintain security and access control in the system.

Dependencies: R1

**ID: R3**

Title: Edit Profile

Description: Player can be able to edit their information or profile.

Rational: To modify the players details.

Dependencies: R1, R2

**ID: R4**

Title: Select Level

Description: Player must be able to select levels before playing game.

Rational: To provide player to change map.

Dependencies: R2

**ID: R5**

Title: View Control Button

Description: Player can view detail information about button for playing game.

Rational: To view detail about controller.

Dependencies: R2

**ID: R6**

Title: View Highest Score

Description: Player can view highest Score in the game scored by all players.

Rational: To view highest Score.

Dependencies: R1, R4

**ID: R7**

Title: View Collected Coins

Description: Player can view collected coins on the screen while playing game.

Rational: To view coins collected.

Dependencies: R4

**ID: R8**

Title: View Live

Description: Player can view lives on the screen while playing game.

Rational: To view lives.

Dependencies: R4

### Non-Functional Requirement

**ID: Q1**

Title: Performance

Description: The game need to response fast to players. It must provide result accurate with high performance.

Rational: To provide better performance.

Dependencies: Q2, Q3

**ID: Q2**

Title: security

Description: Players’ data must be secured and encrypted.

Rational: To provide security and minimize unauthorizes access.

Dependencies: N/A

**ID: Q3**

Title: Reliability

Description: The system should be reliable. Data should be retained without change.

Rational: To make system reliable with accurate result.

Dependencies: Q2

**ID: Q4**

Title: Availability

Description: System must be available all the time without any errors.

Rational: To make system available.

Dependencies: Q3, Q2

**ID: Q5**

Title: Usability

Description: The system should be easy to operate.

Rational: To minimize complexity.

Dependencies: Q1, Q3

### Prioritization

Prioritization is an important factor which helps to prioritize the requirement to develop the better system. It helps to arrange the both functional and non-function requirement according to importance. With proper prioritization, it will be easy to develop system. It helps in distribution of resources and time and help to determine less important requirement and very important requirement. It permits understanding of system.

To understand prioritizes, MoSCoW prioritization technique is used. This technique is used to choose the prioritization order of the requirement. The letter of MoSCoW stand for:

* + Must Have: The compulsory requirement of the system is listed in must have. Lacking must have requirement system will be unusable.
  + Should Have: The important requirement of the system is listed in should have. Should have requirement will expand scope of the system.
  + Could Have: Those requirements which add benefit are listed in could have. Could have requirement will boost the system.
  + Won’t Have: Those requirements which are not included in the system are won’t have. (Agile Business Consortium, 2015)

The prioritization of the requirement of the system using MoSCoW prioritization are listed in below table:

Functional Requirement

|  |  |  |
| --- | --- | --- |
| ID | Functional Requirements | MoSCoW |
| R1 | User Registration | Should Have |
| R2 | User Login | Should Have |
| R3 | Edit Profile | Could Have |
| R4 | Select Level | Must Have |
| R5 | View Control Button | Should Have |
| R6 | View Highest Score | Should Have |
| R7 | View Collected coins | Must Have |
| R8 | View Lives | Must Have |

Non-Functional Requirement

|  |  |  |
| --- | --- | --- |
| ID | Non-functional Requirements | MoSCoW |
| Q1 | Performance | Must Have |
| Q2 | Security | Should Have |
| Q3 | Reliability | Should Have |
| Q4 | Availability | Must Have |
| Q5 | Usability | Should Have |

## Use Case

Use case is a procedure used in system analysis to classify, clarify and unite system requirement. It is the pictorial representation of relating interaction of users with system. It is made up of a set of probable sequences of collaboration of users with system in related to goal. It shows functionality of the system. It is used for gathering necessary requirements of the system. In use case diagram ovals are used to represent use cases and stickman for actor. Use case diagram is drawn on the base on actual situation of the system. (SearchSoftwareQuality, 2015)

Reason for using use case diagram are as follows:

* It holds Functional requirement to read and tracking format.
* They represent the goal of interaction between an actor and the system.
* It can be used to identified dependency and constraint in multiple use cases.

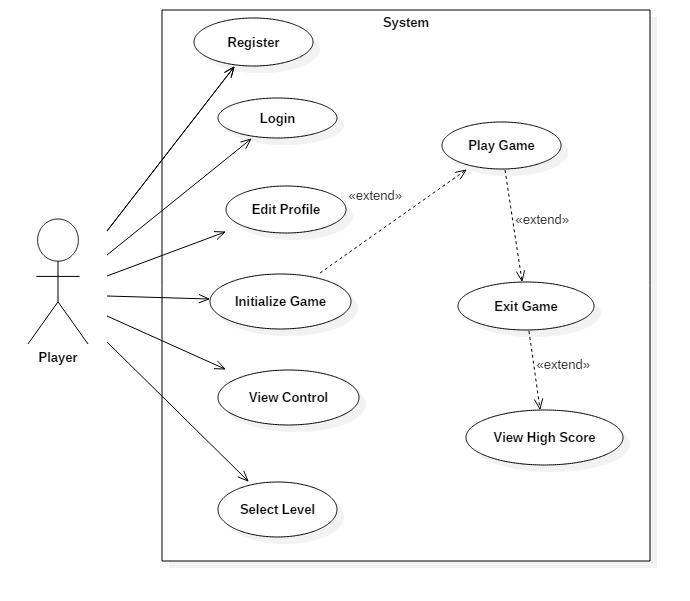


Figure: Use Case Diagram

Description of use case diagram:

* Player can register into the system.
* Player can login into system with legal information.
* Player can edit their profile as necessary after login into the system.
* Player can view all the details information about control buttons of the game.
* Levels can be selected by the player before playing game.
* Game is initialized after selecting level.
* Player can exit game at any time during playing game.
* After exiting game player can view their score with highest score/

## Architecture

Architecture is the method of defining a structured solution that meets all the operational and technical requirement. It acts as a blueprint for the system and provides an abstraction to achieve the system complexity and create a communication and coordination mechanism between components. Architecture involves a set of signification decisions that can have considerable impact on maintainability, quality and performance. (Msdn.microsoft.com, 2014)

### System Architecture

System architecture is known as the method of defining a structure solution that meets all the technical and operation requirement. It involves series of decision based on a wide range of factor. It determines the structure, behavior and design of the system.

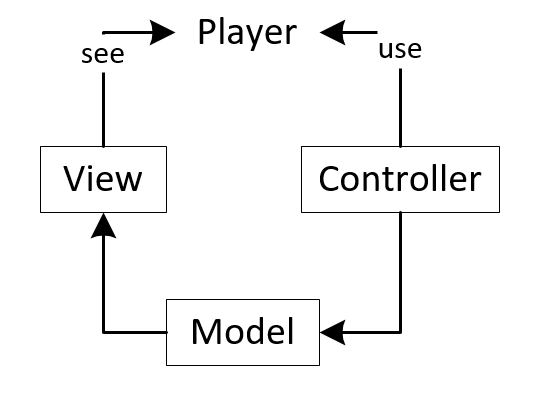
Model View Controller (MVC) design pattern will be used for developing the system. It is a system architecture pattern and made up of three parts i.e. Model, View and Controller. MVC pattern isolates the application logic from user interface layer. Java will be used as programming language with MySQL as database. 

Figure: MVC design pattern

### Initial Class Diagram

A class diagram is an illustration of the relationship and dependencies among classes. It is static diagram. Class diagrams describes the attributes and operations of the classes. They are arranged in group that share common characteristics. Class diagram are extensively used in modelling of object oriented systems. Class diagram demonstrations a collection of classes, interface, collaborations, constraints and association. (www.tutorialspoint.com, 2015)

Possible candidate classes, attributes and functionality are acquired with the help of Natural Language analysis.

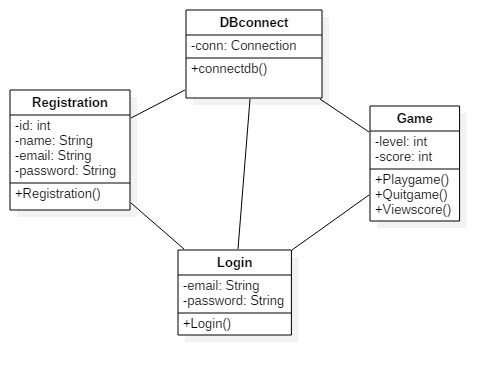


Figure: Initial Class Diagram

## Conclusion

An appropriate documentation on Analysis specification of Break the Wall game is accomplished. At first, all the system requirement is identified and classified into functional and non-functional requirement. All the requirements are gather and explain as necessary. All non-functional and functional requirement are prioritized with the help of MoSCoW technique. From using this technique, all the important and less important requirement are known. The interaction of player with the system is shown with Use Case. Design pattern was described for designing the system in system architecture. MVC design pattern is picked for system design. At last, initial class diagram was prepared. Using NLA potential class with their attributes and functions initial class diagram was organized.

# References

Agile Business Consortium, 2015. *MoSCoW Prioritisation.* [Online]   
Available at: https://www.agilebusiness.org/content/moscow-prioritisation-0  
[Accessed 17 04 2017].

Msdn.microsoft.com, 2014. *Chapter 1: What is Software Architecture?.* [Online]   
Available at: https://msdn.microsoft.com/en-us/library/ee658098.aspx  
[Accessed 22 04 2017].

SearchSoftwareQuality, 2015. *What is use case? - Definition from WhatIs.com.* [Online]   
Available at: http://searchsoftwarequality.techtarget.com/definition/use-case  
[Accessed 21 04 2021].

SearchSoftwareQuality, 2017. *What is requirements analysis (requirements engineering) ? - Definition from WhatIs.com.* [Online]   
Available at: http://searchsoftwarequality.techtarget.com/definition/requirements-analysis  
[Accessed 16 04 2017].

www.tutorialspoint.com, 2015. *UML - Class Diagram.* [Online]   
Available at: https://www.tutorialspoint.com/uml/uml\_class\_diagram.htm  
[Accessed 22 04 2017].