CS319

Object Oriented Software Engineering

System Design Report

Space Out

Group 3-I

Doruk Çakmakçı

Arda Atacan Ersoy

Anıl Erken

Umut Berk Bilgiç

Contents

1 - Introduction

Space Out is retro style sci-fi platformer game that anybody can use to "space out" from their boring life. It is implemented using Java because, Java is one of the most common programming languages in the world. Therefore, we can reach various reliable sources to ease our job. Nearly all computers with standard hardware and software specifications can run Space Out using JVM. Space Out's graphics are 2D and its environment is designed with respect to a retro style platformer game. Not to use high graphics provides the user to get more performance and quick response time and this makes Space Out more playable. To be able to entertain the user, Space Out has different power ups with special animation effects. Local and global leaderboard systems attract more people to Space Out. From another perspective, creating a level from fundamental blocks, structures and enemies help user to unleash their creativity.

1.1 - Purpose of The System

The main goal of Space Out is to entertain the users with rich contents(character skins, retro-space themed graphics) and gameplay. Gameplay is in accordance with ease of use principle(only 3 keys on keyboard are needed). With this aspects, Space Out aims to make the users space out from their daily stress and problems.

1.2 - Design Goals

Design goals includes many aspects such as graphical advantages and game performance that makes the game more qualified and effective. These design goals come from non-functional requirements that are mentioned in the analysis report. Design goals explained elaborately in the below.

Graphical Advantages

Space Out is a 2D platformer game and that's why game does not have high-quality graphics but this case increases Space Out's game performance and in the game, user does not encounter any delays. By this performance advantage, lots of animation will be used in the game.Because of three distinct world concept, Space Out includes different materials, terrains, bricks and so on. All of these serve user to play a more attractive game.

Game Performance

Since middle quality graphics and 2D platformer game concept is used, every standard computer can run Space Out easily. Thus, in terms of performance, when user made a request in the game, response time will be very quick and user will not be exposed to delay. All of the game components have been chosen with respect to get more performance but graphics and performance relation causes a tradeoff. By considering both of their advantages and disadvantages, we try to maximize user's comfort and joy by our decision.

Interface

User interface is designed according to facilitate player's game experience and ease of learning game. In order to play Space Out, user must login or signup, but we hold these pages simple as much as we can. In the menu we put only necessary informations and buttons to make the interface simpler. But this situation does not mean that Space Out has no features. We only add massive features to the system such as level maker and leaderboard system, and these features are very clear and understandable in the game. Additionally, gameplay and character control buttons are arranged to make user more comfort in the game. For instance, W, A, D buttons are selected and these buttons are used bulk of platformer game. All of designing procedures are done to comfort user in the game.

Extendibility and Maintainability

One of the most important goal in the software programming is the life of program. Hence, program's subsystems, implementation and features should be appropriate for extendibility and maintainability. Java, IntelliJ, MVC pattern and other popular tools are used in Space Out. Because popularity and wide usage of these tools provide us various easiness and options while we are implementing game. Without affecting and changing many subsystems, we could add new features to the game. Wide usage of them makes Space Out more reachable and everyone can run and play it easily. Since these tools are compatible with every standard computer, mobile phone and tablet.

Modifiability, Flexibility and Adaptability

To be able to satisfy extendibility and maintainability, Space Out must satisfy modifiability, flexibility and adaptability first. As long as program is adaptable and flexible, it is also extendible and maintainable. Thus, subsystem coupling is tried to be minimized. This leads us to change unwanted features of the game easily or adding new functionalities as well. Besides using level-maker, Space Out will be flexible for each user demand. On the other hand, implementing low coupling subsystems is difficult for us but it has great advantages for users.

High Availability, Reusability and Productivity

Leaderboard system and the usage of Java increase the availability of Space Out. And also by having custom level maker, our blocks, terrains, enemies and other objects are reused and they will be more effective in the game. When user creates their own level, they will be thinking like programmers and by that their productivity and reasonable thinking skill will be increased.

1.3 - Definitions, Accronyms and Abbreviations

MVC(Model View Controller): The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller.[1]

JDK(Java Development Kit): The Java Development Kit (JDK) is an implementation of either one of the Java Platform, Standard Edition, Java Platform, Enterprise Edition, or Java Platform, Micro Edition platforms[1] released by Oracle

Corporation in the form of a binary product aimed at Java developers on Solaris, Linux, macOS or Windows.[2]

JVM(Java Virtual Machine): A Java virtual machine (JVM) is an abstract computing machine that enables a computer to run a Java program.[3]

JRE(Java Runtime Environment): Java Runtime Environment (JRE) is a software package that contains what is required to run a Java program.[4]

VPC(Virtual Private Cloud): A virtual private cloud (VPC) is an on-demand configurable pool of shared computing resources allocated within a public cloud environment, providing a certain level of isolation between the different organizations (denoted as users hereafter) using the resources.[5]

GCloud(Google Cloud): Gcloud is a tool that provides the primary command-line interface to Google Cloud Platform.[6]

HDD (Hard Disk Drive): A hard disk drive (HDD), hard disk, hard drive or fixed disk is a data storage device that uses magnetic storage to store and retrieve digital information using one or more rigid rapidly rotating disks (platters) coated with magnetic material.[7]

2 - Software Architecture

2.1 - Subsystem Decomposition

From several architectures taught in context of CS319, the most appropriate architecture for our project is four-tier architecture. Layer 1-4 (increasing from top the bottom) are named as View, Model, Controller-1, Controller-2 respectively. The figure 1 below represents the subsystem decomposition for our design.

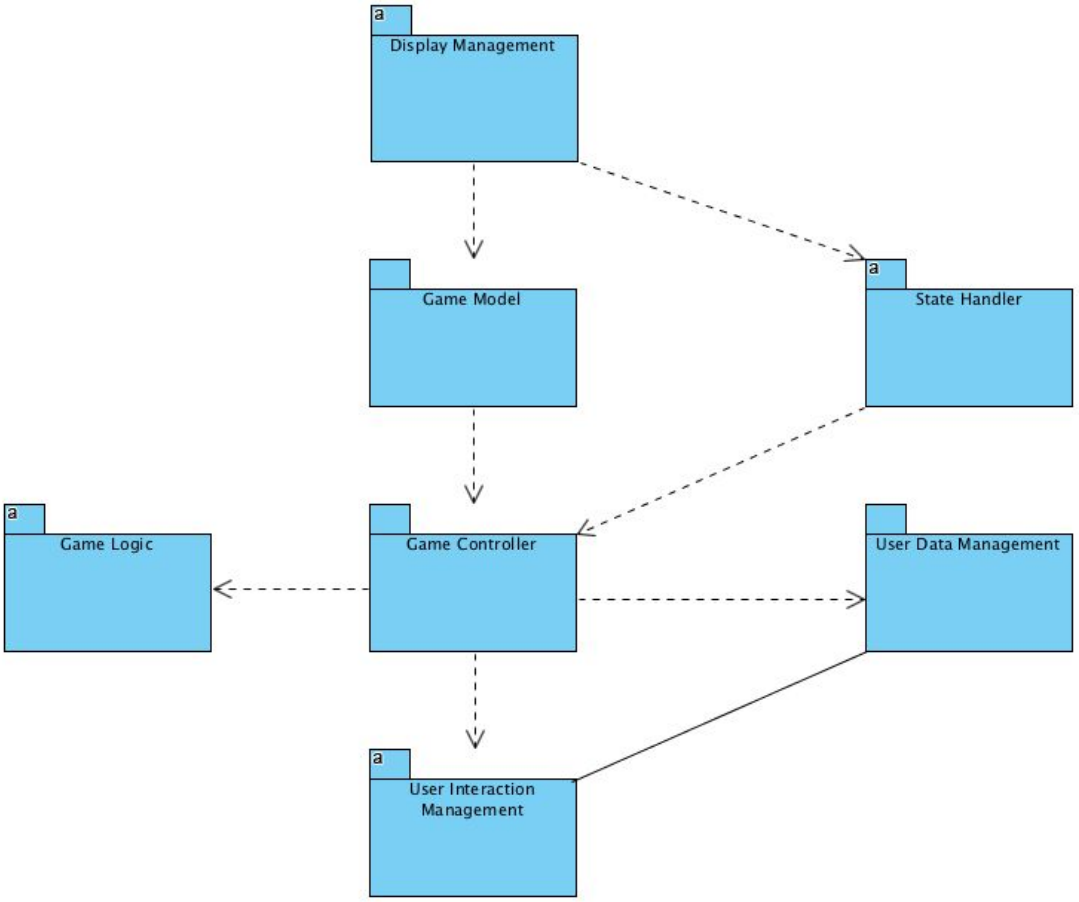

Figure 1: Subsystem Decomposition for Space Out

Layer 1, View Layer, consists of classes that are related with user interface (Menu related classes, in-game sprites, classes related to display etc.) All of the classes mentioned are included in DisplayManagement subsystem. However, user inputs(keystrokes, mouse events .etc.) and sign-in,sign-up use cases are handled in Layers 3 and 4 by interaction between UserInteractionManagement and UserDataManagement subsystems.

Layer 2, Model Layer, is composed of classes that form a model for in-game elements (character position, enemy position, positions of blocks), in-game states(a deterministic finite automaton that shows interactions between in-game(while player

plays the game) and external states (Whether the user is in main menu or settings or any other activity)) and also classes that form a FSM for the application itself (including the attributes of the user and the application itself). In-game part is handled by classes contained in Game Model subsystem. On the other hand, external part which is composed of menu navigation, achievements, leaderboards, settings is handled by classes contained in State Handler subsystem.

Layer 3, Control 1 Layer, contains the subsystems that have ability to change the in-game state elements and external state elements contained in layer 2. In general, this subsystem is the controller for the model. Game Logic subsystem contains the game engine which controls collisions (collision detection) and applies physics laws to inputs from Game Controller(gets inputs from User Interaction Management) in context of the parameters provided also by Game Controller (gravity varies among different worlds). User Data Management holds the user data(username, password, current skin, skins list, achievement list etc.) and provides a security layer in collaboration with User Interaction Management subsystem.

Layer 4, Controller 2 Layer, is an one way interface(only gets input (mouse click or keystroke) from the user for which the Display Management subsystem prompts the user) between the user and the application. The controller subsystems are divided to 2 different layers(Layer 3-4) to create a boundary between some of the controller elements(game logic, user data and game controller) and the user itself.

2.2 Hardware/Software Mapping

Space Out runs on JVM. Therefore, as a software requirement, Java must be installed on the computer where Space Out is going to be played. When java is

initialized successfully, the user needs to open the .jar file to play the game. Internet connection is essential for Space Out due to cloud sync for factory levels, achievements, updating global leaderboard and in essence, to log in or sign up to play the game.

From hardware perspective, a standard computer with average processing power, average CPU(for computations in context of the logic of the game and java requirements) and GPU(for graphics processing and display). From another point of view, keyboard and mouse is needed to play and pause the game and select menu items and create levels with the drag and drop service of the level maker respectively. Gameplay of Space Out will be better if a stronger GPU is present in the corresponding computer due to GPU acceleration present in newer and more powerful GPU's.

2.3 Persistent Data Management

Game Data will be partitioned to cloud (Google Cloud's VPC service will be used) and client HDD. Some portion of data's are dynamic such as client-created levels, leaderboard entities and rankings, character skins that are unlocked with earning the corresponding achievement and achievements itself, log in and log on username and passwords etc.. The dynamic data will be stored on cloud. On the other hand, the data for default game maps and their components, main character's default skin, sound effects, main menu and gameplay sounds, images, graphics and sprites etc. will be held locally. All levels will be files with .lvl extension but, in order not to take more disk space from the client's machine, user will need to download the client-made levels to play.

2.4 Access Control and Security

To be able to play Space Out, internet connection is needed because each user must login or sign up to the system. User's information and their datas(leaderboard scores) are kept in Google Cloud Service in a safe way. By doing that each user's scores, achievements, opened power ups, skins and custom levels will be private for him/her. Google Cloud Service has various precautions for any security issue, thus user can sign up/login and play the game in a secure way.

2.5 Boundary Condititons

Initialization of The Game is the first boundary condition to play Space Out. Game will be extended (.jar) file. Therefore, user does not need to install the game because .jar files are executable, when user has an JVM. However, some levels and stuffs will be installed to reduce cloud database spacing.

Termination of The Game can be done 2 different ways. User can click the "Exit Game" button when he/she is in the main menu. While user is playing Space Out, user can push the "Esc" button from the keyboard after that user can click the "Exit Game" option from the game.

Error of The Game could occur many different ways when user play or try to open Space Out. For instance, sounds, images or Java could not be executable in that computer and user must install the required programs to run Space Out. And when user lost their internet connection his/her score will not be included in the leaderboard system and his/her game will be functionless.

3 - Subsystem Services and Low-Level Design

3.1 Game Model

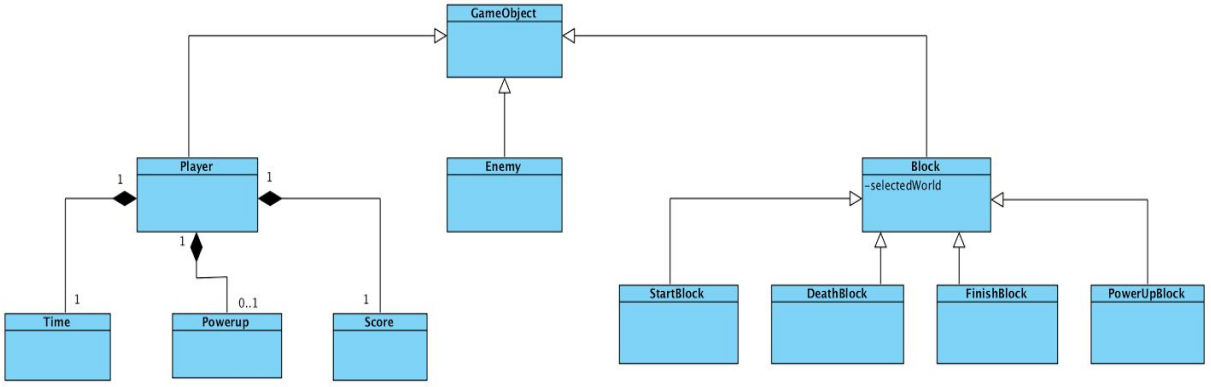

Figure 2: Class diagram for Game Model

Game Model subsystem contains in-game entity classes. Player, Enemy and Block are GameObjects, so they extend GameObject class. Also, Player is composed of Time, Powerup and Score classes. On the other hand, Block is the superclass of StarBlock, DeathBlock, FinishBlock and PowerUpBlock. These classes will help us to detect player's in-game state and performance, and responsible classes in Controller part of our MVC design will trigger the necessary actions accordingly.

3.2 Game Controller

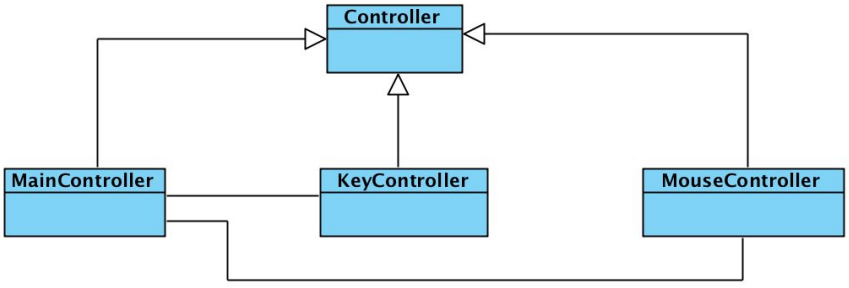

Figure 3: Class diagram for Game Controller

Game Controller subsystem contains user input related classes. KeyController, MouseController and MainController are subclasses of Controller class. Those controller classes will recieve inputs from User Interaction Management subsystem and will send them to necessary subsystems to provide user interaction.

3.3 User Interaction Management

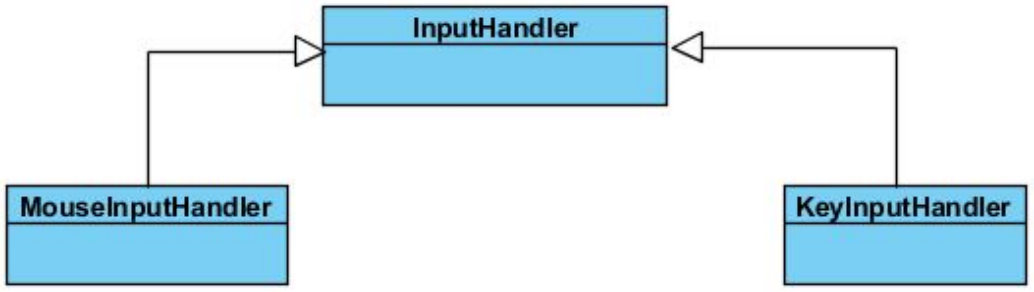

Figure 4: Class diagram for User Interaction Management

User Interaction Management subsystem has classes that listen for mouse or keyboard input and sends the input to corresponding controller. MouseInputHandler and KeyInputHandler are subclasses of InputHandler.

3.4 Game Logic

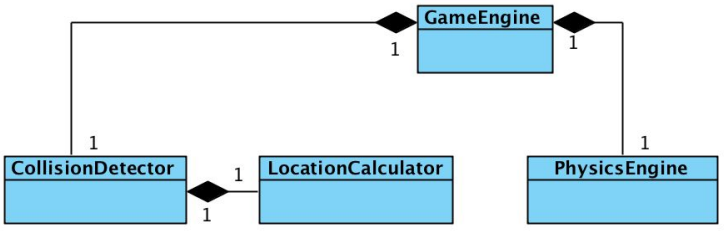

Figure 5: Class diagram for Game Logic

Game Logic subsystem has the GameEngine class. It is composed of CollisonDetector and PhyicsEngine classes. Also CollisonDetector is composed of LocationCalculater. These classes do the required physical calculations. Using the location of main character, enemies and blocks, those classes detect the collisons, hande game objects' movements by using gravitational constant of the current game level and other factors that affect character speed, acceleration etc.

3.5 Data Management

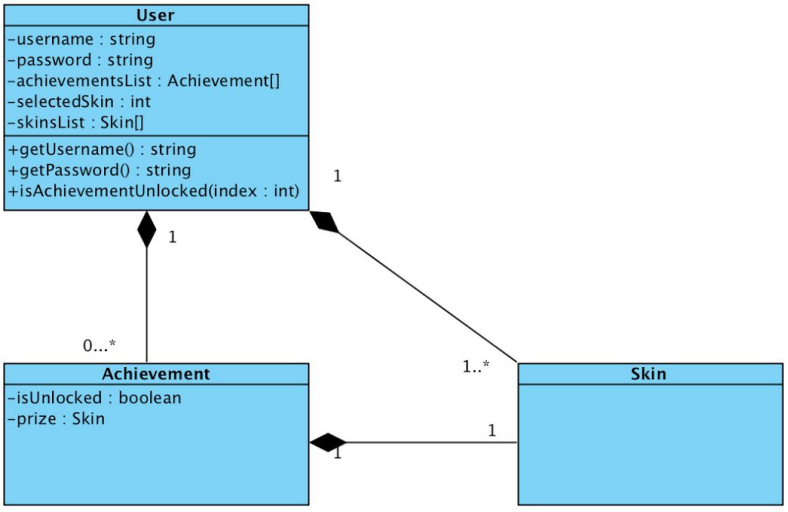

Figure 6: Class diagram for User Data Management

User Data Management has classes that authenticate the user and provide the corresponding content for them. User class will be composed of Skin and Achievement classes, because they are different for each user. Username, password attributes and their getter methods will be used to authenticate the user. isAchievementUnlocked method will run in each gameplay and it will edit the

achievementsList array, if necessary. By this, we will detect the achievements that user won. Since winning achievements unlocks different character skins, Achievement class is composed of Skin class. So that we will detect the skin user won by any achievement and update selectedSkin attribute of the User accordingly.

3.6 State Handler

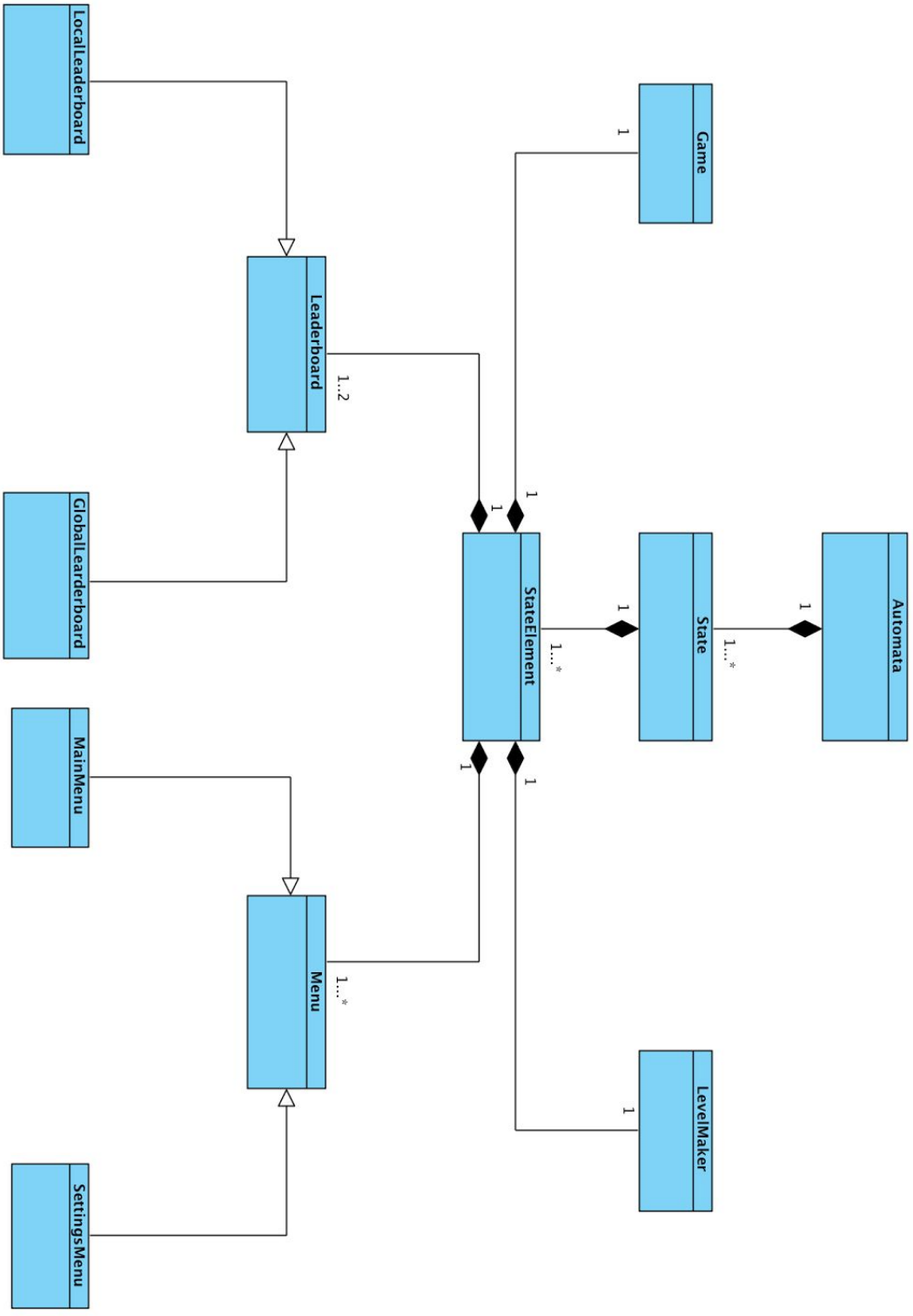

Figure 7: Class diagram for State Handler

State Handler subsystem contains classes that provide navigation in game menu. This is a finite automata that handles the current state, the input and provides the corresponding content to user as its output. All menu options - Game, LevelMaker, LeaderBoard(with two subclasses for local and global), Menu(with two subclasses for main menu and settings menu) - are classes that compose StateElement class, and it composes the current State class, and this class composes the Automata, which also receives the input and give the output.

5 - Glossary and References

[1] https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm

[2] https://en.wikipedia.org/wiki/Java_Development_Kit

[3] https://en.wikipedia.org/wiki/Java_virtual_machine

[4]https://en.wikipedia.org/wiki/Java_virtual_machine#Java_Runtime_Environment_from_Oracle

[5] https://en.wikipedia.org/wiki/Virtual_private_cloud

[6] https://cloud.google.com/sdk/gcloud/

[7] https://en.wikipedia.org/wiki/Hard_disk_drive