Bilkent University

Department of Computer Engineering

Object-Oriented Software Engineering Term Project

CS319 Term Project: Disc-OH-4

Design Report

Mehmet Emre Arıoğlu, Güneş Batmaz, Abdurrezak Efe, Enes Kavak

Supervisor: Bora Güngören

Progress Report

November 13, 2016

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Object Oriented Software Engineering Project for course CS319.

Contents

[1 Introduction 1](#_Toc466845582)

[1.1. Purpose of the system 1](#_Toc466845583)

[1.2. Design Goals 1](#_Toc466845584)

[2 Software Architecture 2](#_Toc466845585)

[2.1 Subsystem Decomposition 2](#_Toc466845586)

[3 Subsystem Services 4](#_Toc466845587)

Design Report

CS319 Term Project: Disc-OH-4

# 1 Introduction

## Purpose of the system

The main purpose of this system, Disc-OH-4, is to provide joy to Connect-Four lovers by giving a different gameplay that the usual version of the game. In Connect-Four, the flow of the game is pretty stable, but in Disk-OH-4, the gameplay is more dynamic that the usual one as the Disc-OH-4 is it provides bonuses and time limitation to make the game more enjoyable. Disc-OH-4 aims to improve the players’ capability of thinking and responding fast.

## Design Goals

**Adaptability:** In this system, we decided to use Java programming language to make the system more adaptable for different platforms, such as Windows OS, Linux, Mac OS.

**Efficiency:** This system is needs to be very responsive since there exist a time limitation in the game which is very important for the gameplay.

**Reliability:** The system need to run properly, in other words, should not include any bugs since the bugs will lead the game to crash, and this will harass the gameplay as well as it will disturb the user.

**Usability:** Usability is very important as it will designate the user’s main opinion. In this system, usability is one of the most important features of the system since a joyful game should be simple and understandable.

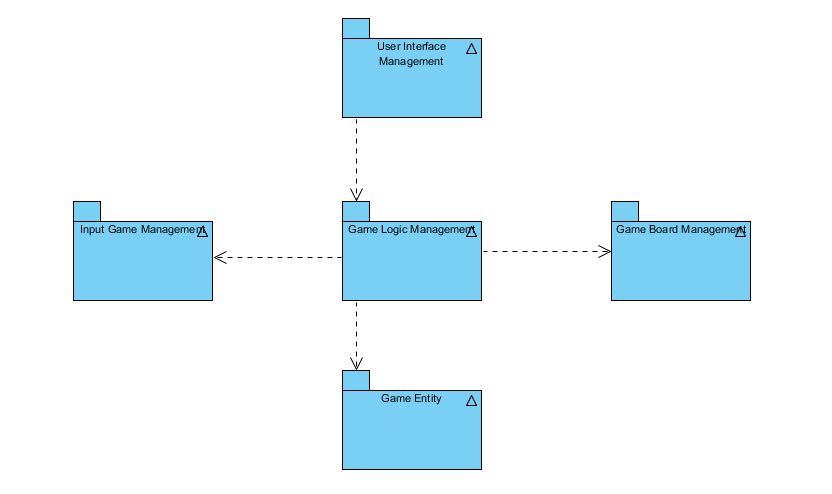
**Extensibility:** By the terms of Object Oriented Software Engineering, the system is design to have an architecture that is capable to be extendable for the future needs.

# 2 Software Architecture

## 2.1 Subsystem Decomposition

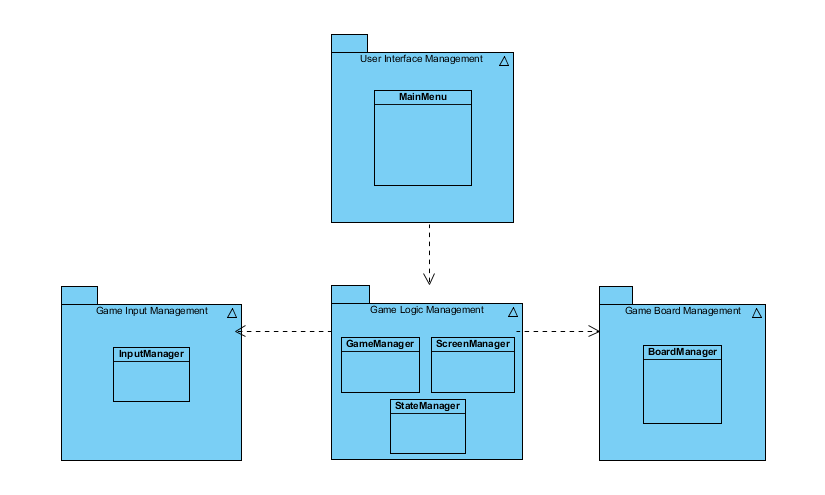
In the following sections we address how the system is broken into parts so that it is much more easier to understand and maintain the organization therefore program easily. Our software is divided into smaller parts which are related to different classes of the solution domain. It is essential during the development since it is a good way to reduce the complexity of our problem.

We break our system into three subsystems namely "User Interface Management", "Game Logic Management" and "Game Entity". We decided to use three level subsystem design since it is best fitting for our system. These subsystems are composed of different solution domain classes. The aim is to reduce coupling and increase the cohesion considering subsystems. We tried to comply with MVC pattern while implementing our interfaces at the decomposition stage of the system.



**Figure 2.1A – Basic Representation of Subsystem Decomposition**

In the first layer there are classes that are hold for presenting an interface between users and system. “Main Menu” class is responsible where the user interact begins so it is located inside “User Interface Management”.

In the “Game Logic Management” subsystem the user’s selection will be interpreted. If the users have chosen “Play”, game screen layout will be created by “GameManager” with handling of package “Game Board Management” and “Screen Manager” class. At each stage of the game “GameManager” will interact with “Input Manager” package to check for player inputs. The “Game Board Management” package contains the classes that are responsible for encapsulating the game layout and the board – disc placement pattern.

**Figure 2.1B – Demonstration of Interactions between Layer 1 and 2**

The current game information updates will be done concurrently in line with the result of the current turn in the game which is checked by “StateManager” class in “Game Logic Management” package.

# 3 Subsystem Services

In this part, detalied information about Disc-O Four’s subsytems will be given.

**SoundManager Class:**

playEffect(effect ID int): it provides us choosing effect sound.

**Game Manager Class:**

gameLoop(): it controls whose turn.

updateTime(): it changes players’ time limit if there is a change due to bonuses.

checkGameState():it controls board’s situation such as draw, win of first and second player and game maintenance.

detectBonus(): it is used for whether a bonus will be applied.

**Board Manager Class:**

backgroundImage(): it is the image which appears as a background during the game.

loadBoard(): before a player’s move, the board will be ready with the help of this function.

drawBoard(): it draws board’s current situation on the screen.

**Screen Manager:**

getScreen(): it gets board’s states.

getGraphics(): it brings proper graphics according to board’s state.

**Board Class:**

stateMatix: it will be a 2D array and hold board’s current situation.

updateBoard():before a player’s move, the board will be updated.

**Disc:**

visibility: in the beginning, the game decides some bonuses and their visibilities. So, it keeps bonuses visibilities.

colour: it is for to determine a disc’s side. It can be first and second players’ or impartial disc.

effectSoundID(): with the help of this function, some sounds will come up according to move.

getType(): It shows us what is the disc’ for.

getColur(): it is for determine the colour of disc.