TUT02_Group08_Deliv02

SFWRENG 3A04: Software Design III - Large System Design

Anando Zaman
Brian Kibazohi
David Yao
Qiushi Xu
Tamas Leung

1 Introduction

1.1 Purpose

The purpose of this document is to describe the software architecture of the game-library application, ”Ready-player-one”. This document will outline the components of the application for both the client and developer based on the business events developed in Deliverable 1. In addition, this document will contain the architecture decisions that may be of benefit for any future development teams that want to refer back and understand specific architectural choices.

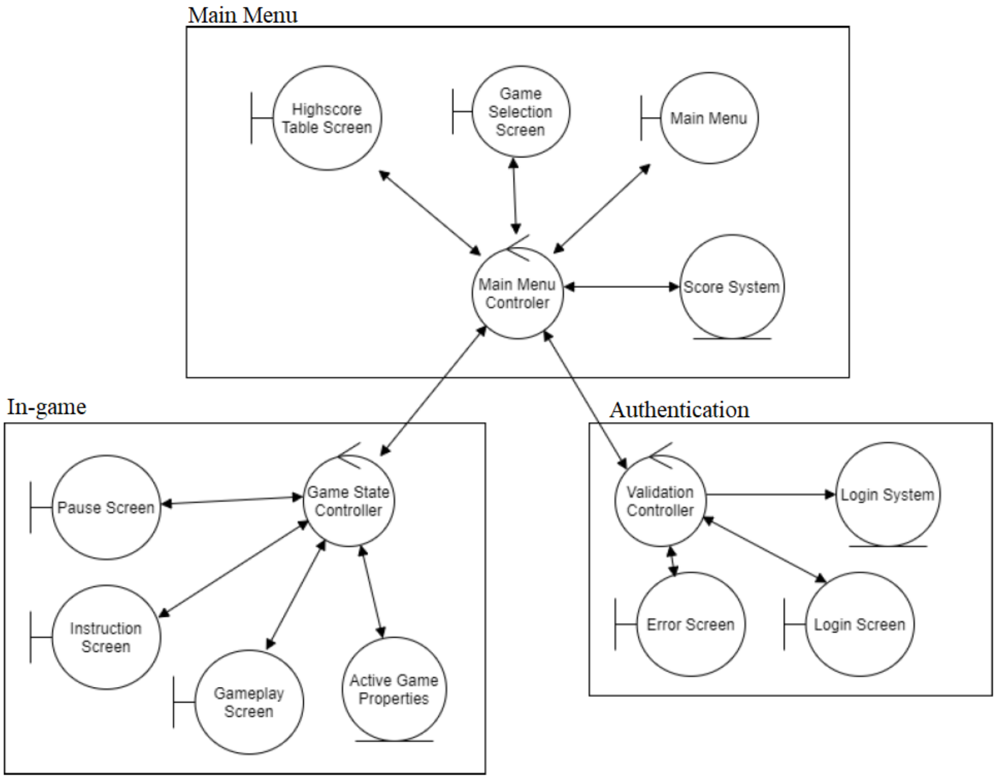
1.2 System Description

The ”Ready-player-one” system is an interactive game library that will allow the user to play a series of arcade-style games while maintaining high-scores amongst their peers. It is comprised of three subsystems namely ”main menu”, ”In-game”, and ”Authentication”. The ”main menu” subsystem will be used to link the other two subsystems. This module will allow the user to view the five available games and choose the one that they want to play. Additionally, it will be responsible for displaying the available high-scores and communicate with the ”Authentication”, and ”In-game” subsystems. Next, the ”In-game” subsystem will be used to track the players activity when playing a game. It will also keep track of their score for a particular game. Finally, the ”Authentication” subsystem will be used to authenticate the user. This ensures that any game played by the user will have the new score reflected only on their account.

1.3 Overview

This document details the design of the software architecture of our system, with the aid of several class diagrams. The document is divided into three sections - the analysis class diagram, architectural design details, and class responsibility collaboration cards.

2 Analysis Class Diagram



3 Architectural Design

3.1 System Architecture

The architecture style that the team believes is best suited for our system is that of the the Presentation-Abstraction-Control (PAC) architecture. The team has chosen this type of architecture because it is suitable when there is a need to separate several user interactions. Our use cases would benefit from having their own controllers for functionality rather than relying on a larger, more general controller. The rationale behind this is that it helps reduce complexity and ensures high-cohesion & low-coupling amongst the modules. Thus, the code structure becomes more modular and prevents a single controller from doing several activities. Additionally, it allows for "design for change" as additional changes in the future can more effectively be made without having unwanted effects on the whole system.

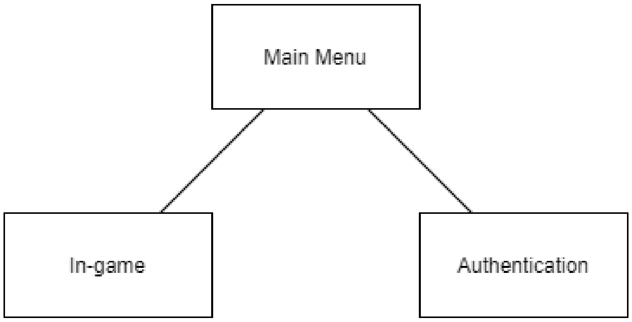


Figure 1: PAC Structure Architecture Diagram

3.2 Subsystems

The following is a brief explanation of the subsystems for this project that outlines their purpose and relationship with other subsystems.

a) Main Menu

- Purpose: This agent is the highest-level agent that transitions the user between the various in-game agents. This agent also interacts with the score-system to access high-score information.
- Direct Relationships: Authentication Agent and In-game agent.

b) Authentication

- Purpose: This agent is a low-level agent that authenticates the user. It interacts with the login system to validate user credentials.
- Direct Relationships: Main-menu agent.

c) In-Game

- Purpose: This agent is a low level agent that handles the game activity for each game. It interacts with the "active game properties" entity to maintain the local game score.
- Direct Relationships: Main-menu agent.

4 Class Responsibility Collaboration (CRC) Cards

4.0.1 Entity Class CRC Cards

Class Name: Score System	
Responsibility:	**Collaborators:**
Record and store the high score of each game	Main Menu Controller
Add high scores of all 5 games as overall score	
Save the overall score score	Main Menu Controller

Class Name: Login System	
Responsibility:	**Collaborators:**
Save users' usernames	Validation Controller
Save users' passwords	Validation Controller
Authenticate users' usernames	Validation Controller
Justify users' passwords with the according usernames	Validation Controller

Class Name: Active Game Properties System	
Responsibility:	**Collaborators:**
Save the content for each mini game	
Load the content for selected mini game	Game State Controller
Execute the user inputs and instructions for the selected game(receive new game property)	Game State Controller
Update the game state for the selected game	Game State Controller
Load instructions for selected mini game	Game State Controller

4.0.2 Boundary Class CRC Cards

Class Name: Main Menu	
Responsibility:	**Collaborators:**
Display game selection button	Main Menu Controller
Display high scores button	Main Menu Controller
Display exit button	Main Menu Controller

Class Name: Game Selection Screen	
Responsibility:	**Collaborators:**
Display button for game 1	Main Menu Controller
Display button for game 2	Main Menu Controller
Display button for game 3	Main Menu Controller
Display button for game 4	Main Menu Controller
Display button for game 5	Main Menu Controller
Display button to return to main menu	Main Menu Controller

Class Name: Highscore Table Screen	
Responsibility:	**Collaborators:**
Present list of games	Main Menu Controller
Present the users’ score	Main Menu Controller
Present username and high-score associated to a game	Main Menu Controller

Class Name: Login Screen	
Responsibility:	**Collaborators:**
Present login screen window	Validation controller
Receive username input	Validation controller
Receive password input	Validation controller
Allow submission of credentials	Validation controller

Class Name: Error Screen	
Responsibility:	**Collaborators:**
Prompt error screen window	Validation controller
Present list of errors associated with user	Validation controller

Class Name: Gameplay Screen	
Responsibility:	**Collaborators:**
Actively displays the game state	Game State Controller
Display user’s current score	Game State Controller
Accept User input to interact with the Game State	Game State Controller
Accept User input to switch to instruction/pause screen	Game State Controller

Class Name: Instruction Screen	
Responsibility:	**Collaborators:**
Display instructions based on current game selection	Game State Controller
Display button to allow user to switch to gameplay screen	Game State Controller
Display graphics assisting in instructions of gameplay	Game State Controller

Class Name: Pause Screen	
Responsibility:	**Collaborators:**
Display button and accept input to unpause game	Game State Controller
Display button and accept input to exit game and return to game selection screen	Game State Controller

4.0.3 Controller Class CRC Cards

Class Name: Main Menu Controller	
Responsibility:	**Collaborators:**
Display and populate high score screen	Highscore Table Screen
Initiate display of game selection screen	Game Selection Screen
Display main menu screen	Main Menu
Display high scores based on user	Score System

Class Name: Validation Controller	
Responsibility:	**Collaborators:**
Display error screen	Error Screen
Display login window	Login Screen
Prompt authentication steps for login	Login System

Class Name: Game State Controller	
Responsibility:	**Collaborators:**
Prompt gameplay screen when a game is active	Gameplay Screen
Prompt user with instructions to play game	Instruction Screen
Display grew filter over game after game is paused	Pause Screen
Display properties of active game onto the screen	Active Game Properties

A Division of Labour

Table 1: Contributions

Name	**Contribution**
Tamas	Analysis Diagram, Some CRC
Anando	Introduction section, Architectural Design, Some CRC
David	Analysis Diagram, Some CRC
Brian	Architectural Design, Some CRC
Quishi	Architectural Design, Some CRC

T.L

Tamas Leung

A.Z

Anando Zaman

D.Y

David Yao

B.K

Brian Kibazohi

Q.X

Quishi Xu