Module Interface Specification for UnoFlip3D

Team24 unomaster

Mingyang Xu
Kevin Ishak
Jianhao Wei
Zain-Alabedeen Garada
Zheng Beng Liang

January 15th, 2025

1. Revision History

Date	Developer	Changes
January 12th, 2025	Kevin Ishak	Initialize template, add rough draft of section 3,4 and 5
January 13th, 2025	Jianhao Wei	Add modules into section 6
January 13th, 2025	Zain-Alabedeen Garada	Add modules into section 6

2. Symbols and Acronyms

Contents

3. Introduction

The following document details the Module Interface Specifications for UNO Flip. “ADD PROJECT DESCRIPTION”
Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at https://github.com/simon-0215/UNO-Flip-3D/tree/main .

4. Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol := is used for a multiple assignment statement and conditional rules follow the form ($c1 \Rightarrow r1|c2 \Rightarrow r2|...|cn \Rightarrow rn$). The following table summarizes the primitive data types used by UnoFlip.

Data Type	Notation	Description
Character	char	a single symbol or digit
Integer	Z	a number without a fractional component in (-∞, ∞)
Natural Number	N	a number without a fractional component in [1, ∞)
Real	R	any number in (-∞, ∞)

5. Module Decomposition

Level 1	Level 2
Hardware-hiding	Device Input Audio Output Screen Rendering
Behaviour-hiding	Game Rules Logic Turn Management Card Management Player Interaction
Software Decision	Networking Protocol Multiplayer System

Table 1: Module Hierarchy

**The following are the rough draft of the MIS

6. MIS of GameLogic Module

6.1 Module
GameLogic

6.2 Uses
Multiplayer, UI, AssetManagement, Server

6.3 Syntax

6.3.1 Exported Constants
None

6.3.2 Exported Access Programs
`validateMove(playerId, card)`
`endTurn(playerId)`
`shuffleDeck()`
`drawCard(playerId)`

6.4 Semantics

6.4.1 State Variables
`currentPlayer`: Tracks the player whose turn it is
`deck`: Represents the stack of remaining cards in the game.
`discardPile`: Stores played cards
`playerHands`: Stores each player's cards

6.4.2 Environment Variables
`maxPlayers`: Maximum number of players allowed in a game
`flipEnabled`: Boolean to toggle the flip functionality

6.4.3 Assumptions
The number of players, game rules, player restrictions are preloaded
Game Environment are known

6.4.4 Access Routine Semantics

- `validateMove(playerId, card)`
  Transition: Checks if a move is valid
- `endTurn(playerId)`
  Transition: Ends the current player's turn and starts the next
- `shuffleDeck()`
  Transition: Randomizes the card deck
- `drawCard(playerId)`

Transition: Adds a card to the specified player's hand

6.4.5 Local Functions

`shuffleProcess()`:Contain the random algorithm to shuffle the deck

`CardModifier()`: Contain algorithm to draw different card to screen

7. MIS of Multiplayer Module

7.1 Module

Multiplayer

7.2 Uses

GameLogic, Server

7.3 Syntax

7.3.1 Exported Constants

None

7.3.2 Exported Access Programs

`createGameRoom(playerId, roomSettings)`

`joinGameRoom(playerId, roomId)`

`broadcastUpdate(gameId, update)`

7.4 Semantics

7.4.1 State Variables

`activeGames`: Tracks all ongoing game sessions.

`connectedPlayers`: List of currently connected players.

7.4.2 Environment Variables

`serverIP`: IP address of the game server.

`timeoutLimit`: Time limit for a player to respond during their turn.

7.4.3 Assumptions

The connection to other machines can be established successfully

The encryption and decryption methods are known

7.4.4 Access Routine Semantics

- `createGameRoom(playerId, roomSettings)`
  Creates a new game room
- `joinGameRoom(playerId, roomId)`
  Adds a player to an existing room
- `broadcastUpdate(gameId, update)`
  Sends game state updates to all players in a room.

7.4.5 Local Functions

`encryption(information)`: Contain encryption algorithm to encrypt data before sending `decryption(information)`: Contain decryption algorithm to decrypt Data after receiving

8. MIS

References