# **CSC 431** Strategy Game System Architecture Specification (SAS)

**<Team number>**

|  |  |
| --- | --- |
| Andrew Gonzalez | Project Manager, Developer |
| Salahuddin Bawaney | Developer |
| Micheal Bowermaster | Developer |
| Micheal Taylor | Developer |
| Giuseppe Cannavo | Lead Game Designer, Artist, Developer |
| Renel Desir | Developer |

# Version History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author(s) | Change Comments |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Table of Contents

[1. System Analysis 6](#_Toc412746586)

[1.1 System Overview 6](#_Toc412746587)

[1.2 System Diagram 6](#_Toc412746588)

[1.3 Actor Identification 6](#_Toc412746589)

[1.4 Design Rationale 6](#_Toc412746590)

[1.4.1 Architectural Style 6](#_Toc412746591)

[1.4.2 Design Pattern(s) 6](#_Toc412746592)

[1.4.3 Framework 6](#_Toc412746593)

[2. Functional Design 7](#_Toc412746594)

[2.1 Diagram Title 7](#_Toc412746595)

[3. Structural Design 8](#_Toc412746596)

[4. Behavioral Design 9](#_Toc412746597)

# Table of Tables

<Generate table here>

# Table of Figures

<Generate table here>

### System Analysis

#### System Overview

<Describe the system in brief and your architecture choice>

#### System Diagram

<Insert System Diagram>

#### Actor Identification

<Identify all actors interacting with the system>

#### Design Rationale

##### Architectural Style

<Identify and briefly explain the architectural style e.g. 3-tier architecture>

##### Design Pattern(s)

<Identify the design pattern(s) you have deemed applicable to this architecture>

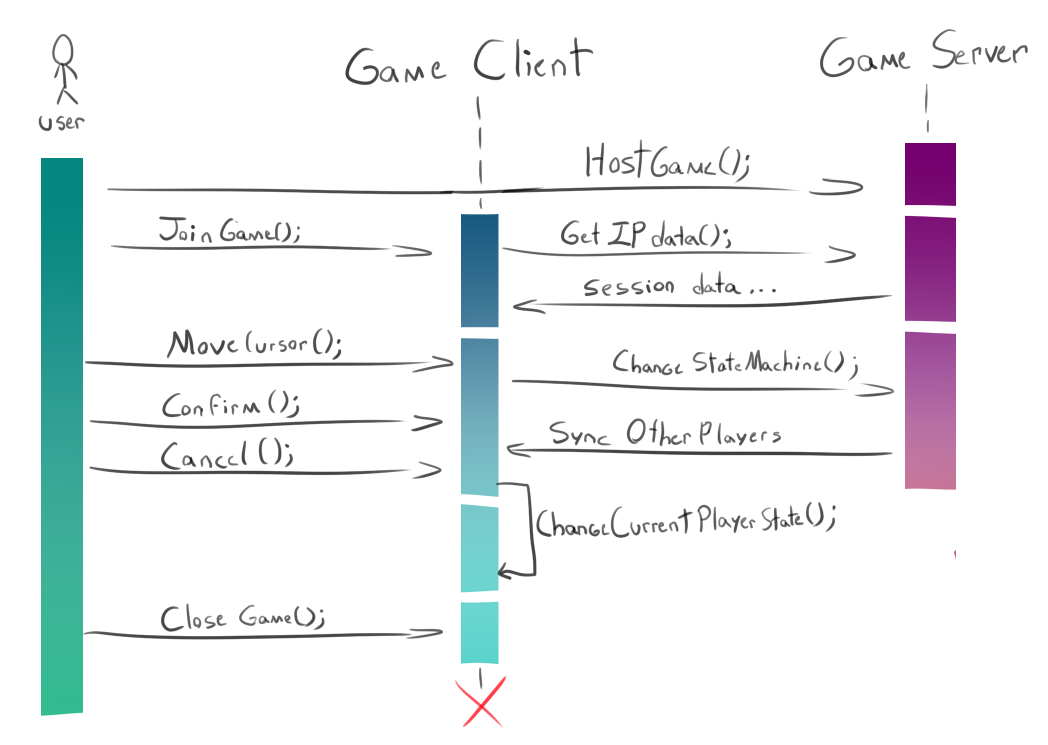
##### Framework

<Identify and briefly explain the frameworks you are using, if any. Also specify the rationale behind selecting this framework>

### Functional Design

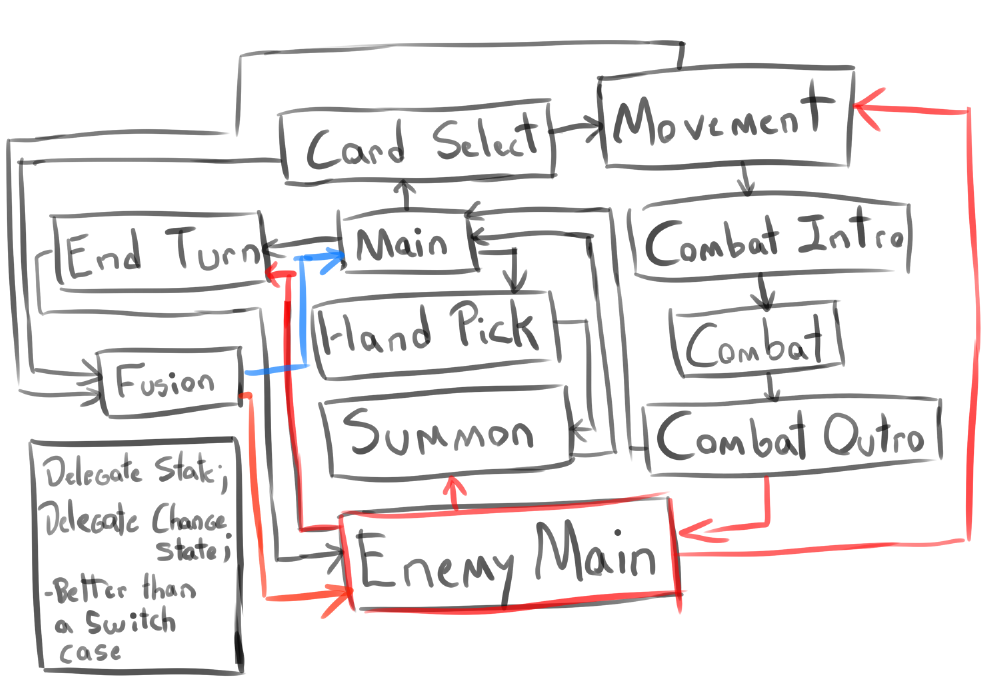
#### Functional Design

**Sequence Diagram**



* When the user Hosts the game, the game client will become the host and responsible for syncing every player that subsequently joins.
* When the player chooses to join the game, they must input the IP address of the host to join them, then the server retrieves the session data and gives it to the client.
* The user can Move their cursor, confirm or cancel actions in their turn and that updates the game server with a change on the state machine and then syncs every player to the current state. They could also just change their current state without the other players knowing (Such as when picking cards from their hand.) Winning and Losing is also handled via state machines.
* Finally the player can close the game when they are done with the game.

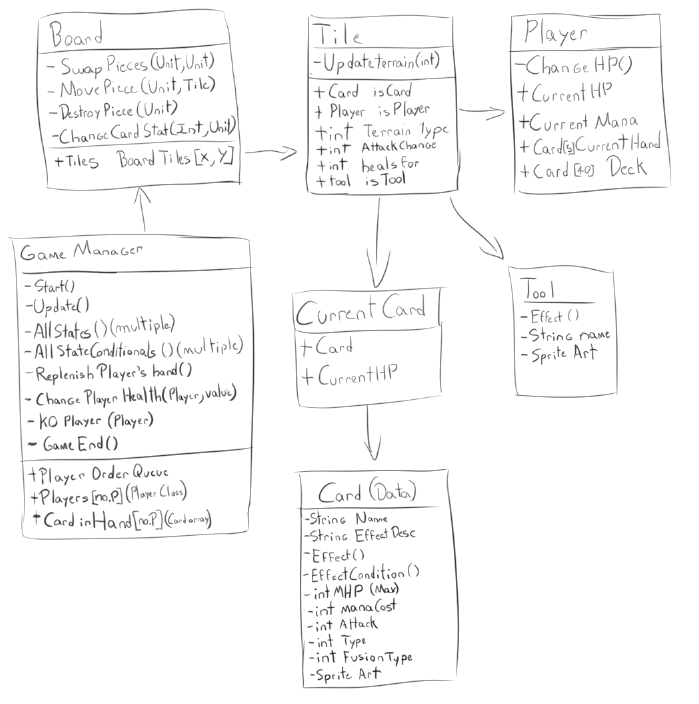
**Game States (Finite State Machine)**



Game States are the states the game will go through to streamline the game flow, it will use delegates to achieve this effect on a Gameplay manager object. It makes the game do one thing and one thing only per state and optimizes runtime play. (Check out Finite State Machines for a better idea)

* -Main: The “Default” state on the current active player, this will allow you to do most things in your turn.
* -Hand Pick: this state is only visible by the active player, as it shows your current hand and the cards that you will play. You can fuse in this menu.
* -Summon: after Picking a card, the player will choose a surrounding space to summon the card.
* -Card Selection: When a player selects a card, it gets highlighted, and the player is able to select a space to where to move the card.
* -Movement: Shows the animation of the card moving towards its destination.
* -Combat intro: when a card overlaps an opponent’s, it starts this phase, it may show effects that trigger before combat starts as well.
* -Combat: Shows animation and calculates the outcome of the combat with the two cards.
* -Combat Outro: Shows the cards moving after the combat is done (if any was defeated) and any of the affected player’s health decreasing.
* -End Turn: Changes the active player to the next one in the queue.
* -Enemy Main: Shows any (non-controllable) players, you can move around the board in this phase.
* Combat, movement, and Summon state should work for all players. Regardless if it’s their turn or not.

### Structural Design



The Game manager changes the board, which holds a number of tiles and these tiles hold one of three: A Card (Current Card), a Tool, or a Player. The current card is there so the card can hold it’s current HP stat and gather its data from the rest of the card’s data. The game manager manages everything in the game by its states, going in some sort of hierarchy as shown in the diagram.