Foundations of Software Engineering

Software Requirements Specification

EN.605.601.81

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this Software Requirements Specification (SRS) is to outline the requirements needed to complete the development of Clue-Less for the Foundations of Software Engineering (605.601.81.FA19) course. This requirements document shall be submitted to the course instructors and grader to receive a grade for the course as well as guide the design and development of the Clue-Less project.

The software requirements detail the requirements specification of what our Clue-Less software package will do. Our objective is to describe what our systems do, their structure, information, and behavior. The flow of events displays the sequence of events which include inputs from the users and outputs from the system to the users.

1.2. Scope

Clue-Less is a simplified version of the popular board game, Clue®. Clue-Less shall be a digital game which allows users to connect to the game and play with friends through a web browser. Clue-Less consists of a grid with nine-rooms, six weapons, and 2-6 players. Clue-Less shall be preferably played through a GUI but shall also support a text-based interface for users with minimal systems.

Clue-Less shall be develop by The Collective group project team with the goal of receiving a high grade in the Foundations of Software Engineering course while also providing a sound learning experience for the software engineering process. Clue-Less is not intended to be sold to retailers post-release and shall be decommissioned upon course completion to ensure no incurred resource costs.

1.3. Glossary

- API (Application Programming Interface) - defines a set of methods and protocols for interfacing between multiple software components
- Backend - refers to the components at the data access layer (layer 2 of the OSI model)
- Frontend - refers to the components at the application layer (layer 7 of the OSI model) which interact directly with the user
- GUI (Graphical User Interface) - provides a visual interface for the user to interact with
- UI (User Interface) - the components that allow the user to interact with the system
- User - a person who interacts directly with the product
- Stakeholder - a person or group with interest or concern in the software system

2. Software Architecture

2.1. Subsystems and Functions

The software product is comprised of six subsystems which are defined by their purposes, namely GUI, API, service, and database. Figure 1 provides a high-level overview of how these subsystems interact with each other. The six subsystems are grouped into the two larger subsystems GUI (frontend) and Clue-Less Server (backend).

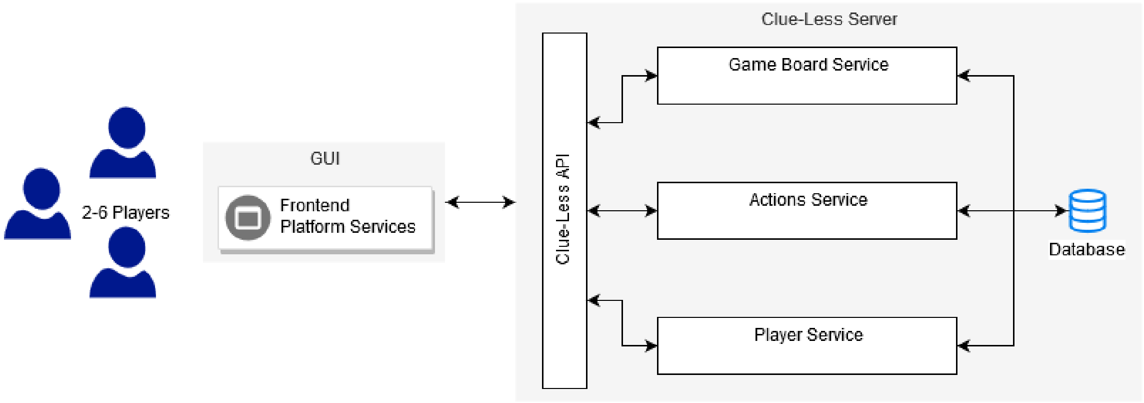

Figure 1: System Architecture Diagram

The GUI is the client-facing subsystem and its purpose is to provide an aesthetically pleasing interface for the end user. The API provides a set of routines used by the UI to interact with the Clue-Less server. Each of the service subsystems handles data processing and game logic for a specific aspect of the game. There is a service for the game board, game actions, and players. The database subsystem stores all the data.

The GUI displays all information relevant to the player and receives information from the player to send to the backend for processing. This will include a live snapshot of the game as it’s being played.

The game board service controls the data and logic for the game board. It maintains the game board state. When the game first starts this service initializes the game board with the appropriate players and items (e.g. case file).

The action service controls the actions that can be taken during the game. A player can move between rooms and hallways, make a suggestion, prove a suggestion, or make an accusation. This service also handles the start game and end game actions.

The player service controls the state of the players throughout the game and notifications to the players. At the beginning of a game, each player is assigned at least one character and weapon. The player service maintains these attributes.

The database simply holds all the system's data. The services interact with it to read, write, update, and delete data as necessary.

2.2. Information and Encapsulation

The UML class diagram in Figure 2 models the classes of information in the system.

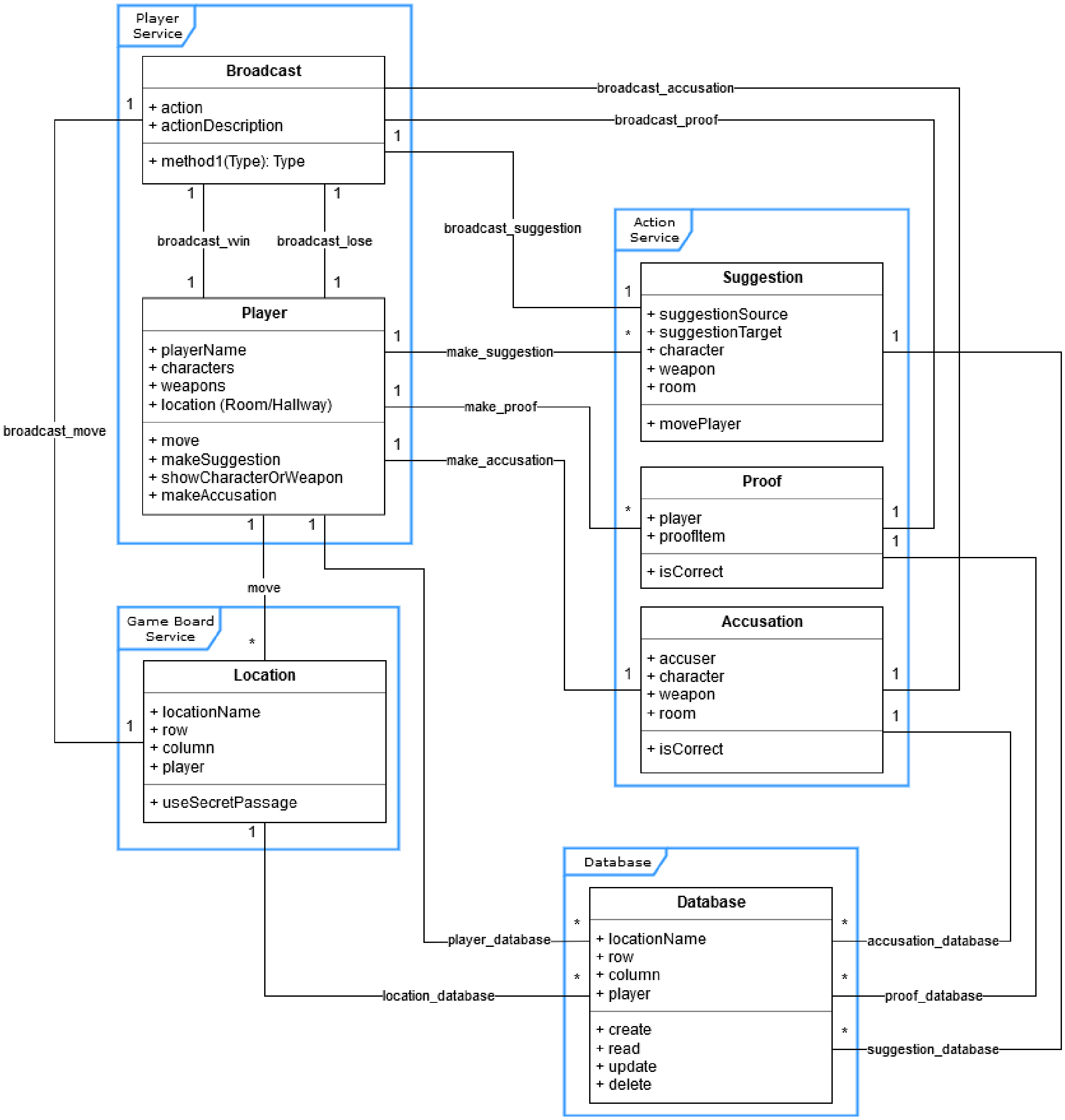

Figure 2: Clue-Less UML Class Diagram

2.3. Interfaces

This system interfaces with the user, the hardware hosting it, and with its contained subsystems. The system interacts with the user through the GUI. The GUI is displayed on the user's computer screen. The user can interact with the GUI through their keyboard and pointing device (e.g. mouse, finger, stylus). User interactions through the GUI are sent to and processed by other subsystems. The API touches all data that is sent to or sent from the GUI. Clue-Less shall be preferably played through a GUI but shall also support a text-based interface for users with minimal systems.

2.4. Information Domain

Information relevant to the GUI:

- Layout of the rooms
- Location of characters and weapons
- The current user's dealt hand of rooms, characters and weapons
- A list of the rooms, characters, and weapons, with the ability for each user to be able to privately mark off as information comes in from other players

Information maintained by the game service includes:

- Current state of the board, including the location of characters and weapons
- The character, weapon, and room of the solution to the current game

The action service encapsulates no data.

The player service encapsulates the information relevant to the players:

- The cards dealt to each player
- The location of the player
- Whether or not each player can currently make a guess
- Player names and characters associated with each player

3. Functional Requirements

3.1. Specifications

The system follows the ensuing instructions for gameplay operations:

- The player doesn't need to roll a die in its turn.
- The player may make an accusation at any time during its turn.
- All player's locations are shown.

Player options of moving around the grid are guided by the following instructions:

- The player's first move must be to the hallway adjacent to its home square. The inactive players stay in their home squares until they are moved to a room by someone making a suggestion.
- Whenever a suggestion is made, the room must be the room the one making the suggestion is currently in. The suspect in the suggestion is moved to the room in the suggestion.
- When a player is in a room (s)he may:
    - Move through one of the doors to the hallway (if it's not blocked).
    - Take a secret passage to a diagonally opposite room (if there is one) and make a suggestion.

  - If the player was moved to the room by another player making a suggestion, the player may stay in that room and make a suggestion, otherwise, the player may move through a doorway or take a secret passage.
- When a player is in a hallway (s)he may:
  - Move to one of the two rooms accessible from that hallway and make a suggestion.
- When a player is in a hallway, another player may not move there, there will only one player per hallway at a time.
- If all of the exits are blocked (i.e., there are players in all of the hallways) and the player is not in one of the corner rooms (with a secret passage), and the player wasn't moved to the room by another player making a suggestion, the player loses its turn (except for maybe making an accusation).

3.2. Requirements

Following are the functions the system must perform and the behavior it must possess:

- Each player shall access the game from a separate computer with a GUI.
- The game rules shall be the same as in the regular (physical) Clue® game except for the navigation.
- All players shall be notified each time the game state changes.
- The message interfaces between the clients and server (client to server, server to client, and broadcast) shall be documented.
- Messages shall be considered as triggers for use cases at the subsystem level.
- A text-based client for the minimal system and a GUI version in the target shall be presented.
- The set of names and graphics shall be stored in the client.
- Messages to and from the server should use standard identifiers of people, weapons and rooms and the mapping from the standard names to the user selected names shall be done in the client.

3.3. Use Cases

An example of a use case scenario for the below gameplay is shown in Figure 3:

1. Player Ana, Ben, Chuck, and Dave join the game remotely and are assigned the users Miss Scarlet, Col Mustard, Mr. Green, and Prof Plum respectively.
2. All players start at their respective locations on the grid.
3. Miss Scarlet moves to the Lounge and makes a suggestion with Mr. Green. Miss Scarlet's suggestion is found to be incorrect and the game moves on.
4. Col Mustard moves to the Dining Room and makes a suggestion with Miss Scarlet. Col Mustard's suggestion is found to be incorrect and the game moves on.
5. Mr. Green moves to the hallway between the Lounge and the Dining Room.
6. Prof Plum moves to the Library and makes a suggestion with Col Mustard. Prof Plum's suggestion is found to be incorrect and the game moves on.
7. Miss Scarlet moves to the hallway between the Dining Room and the Billiard Room.

8. Col Mustard moves to the hallway in between the Conservatory and the Library.
9. Mr. Green moves to the Lounge and makes a suggestion towards Miss Scarlet. Mr. Green’s suggestion is found to be incorrect and the game moves on.
10. Prof Plum moves to the hallway between the Library and the Study.
11. Miss Scarlet moves to the Conservatory through the secret passage and makes an accusation towards Mr. Green. Miss Scarlet’s accusation is incorrect, she can’t win any longer and is skipped, the game moves on.
12. Col Mustard moves to the Library and makes a suggestion towards Mr. Plum. Col Mustard’s suggestion is found to be incorrect and the game moves on.
13. Mr. Green moves to the hallway between the Conservatory and the Library.
14. Prof Plum mover to the hallway between Library and the Study.
15. Col Mustard moves to the hallway between the Library and the Billiard Room.
16. Mr. Green moves to the Library and makes an accusation towards Prof Plum. Mr. Green’s accusation is correct, he wins, and the game is over.

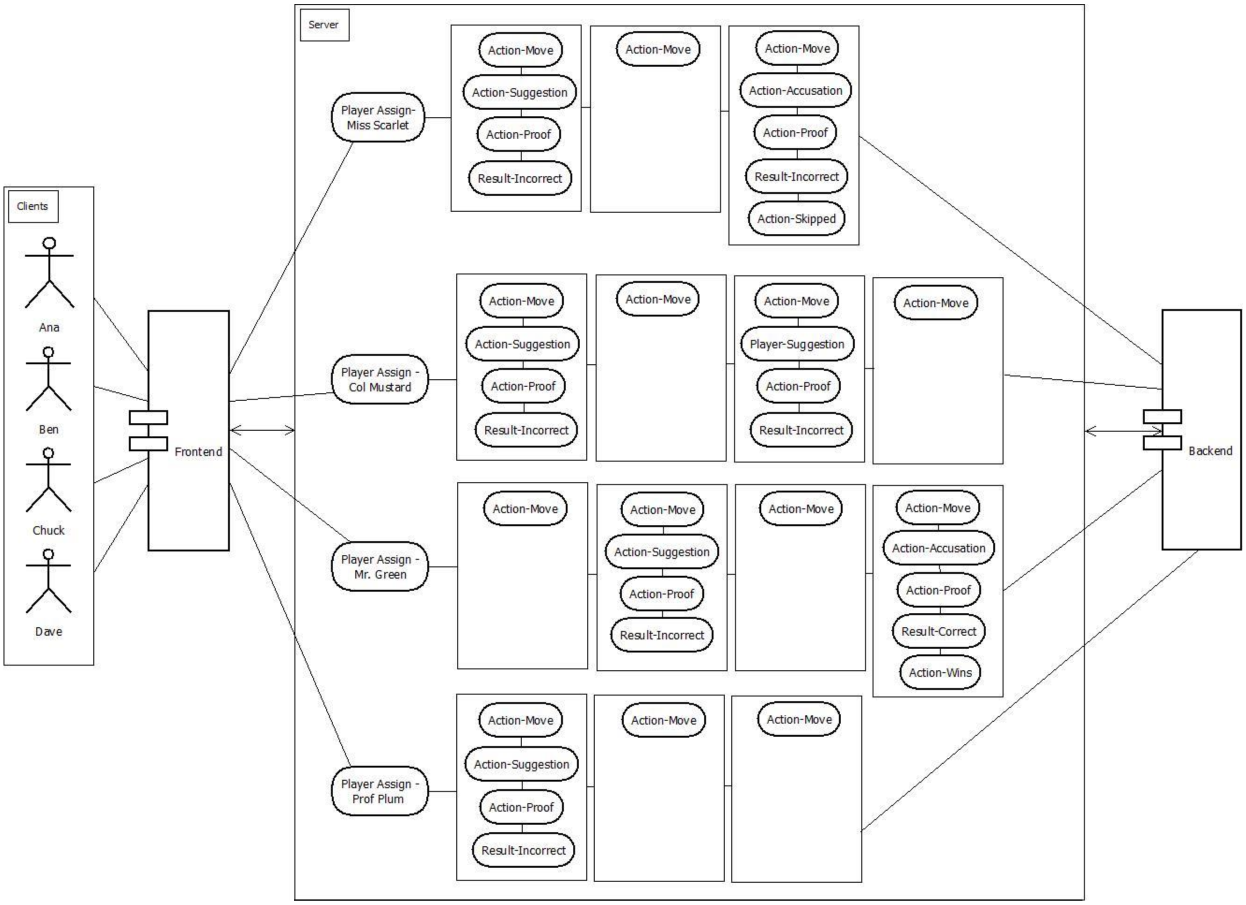

Figure 3 – Use Case Scenario

4. Non-Functional Requirements

4.1. Reliability

- Client connections shall be displayed to the user in three levels
  - Red: client connection falls is less than 15 FPS
  - Yellow: client connection falls between 16 FPS and 30 FPS
  - Green: client connection is greater than 30 FPS
- Clue-Less shall have a 99% uptime.
- Server maintenance shall be announced through the website.
- Maintenance period shall not exceed 2 hours.

4.2. Usability

- Clue-Less shall be played with a maximum of 6 players in one game.
- Clue-Less shall be played with a minimum of 2 players in one game.
- Clue-Less shall start a game with the minimum amount of players if no additional players are found for 5 minutes.
- Clue-Less rules shall be made readily available to the user.
- Game turns shall not exceed 2 minutes.

4.3. Performance

- The server shall be able to scale to a maximum of 3 instances based on the number of concurrent users.
- Each server instance shall be able to host a minimum of 10 games of Clue-Less.
- Each server instance shall be able to hold 100 concurrent users waiting to play a game of Clue-Less.
- Players that are dropped from the game shall receive a notification stating that they have been dropped from the game.
- Players that are dropped form the game shall automatically attempt to reconnect unless the reconnection is cancelled.

4.4. Security

- No user data shall be stored during gameplay.
- Users shall be able to create a unique alias upon connection to the server.
- Data transmission shall be limited to game logic.

4.5. Supportability

- Software and documentation shall be stored in a GitHub repository.
- Clue-Less web page shall work with Firefox and Edge.
- Clue-Less releases shall be maintained until 12/31/2019.

5. Design/Implementation Constraints

5.1. Resources

- Preferably no money needed for hosting solution
- Free or low-cost version control system
- Low latency server

5.2. Time

- Time limit on project is constrained to length of class
- Schedule must follow incremental release date

5.3. Usability

- Intuitive user interface
- Help menu for options

5.4. Miscellaneous

- All user moves must be validated by server
- GUI for client
- Time limit for each move to help speed up games
- Time-out when a player disconnects from the session
- User can create account or use a generic logon
- Easy to upgrade software

5.5. Testing of solution