ALGONQUIN COLLEGE

SCHOOL OF ADVANCED TECHNOLOGY

ICT - Applications & Programming
Computer Engineering Technology – Computing Science

A31

Game C/S Model – Collaboration Diagram

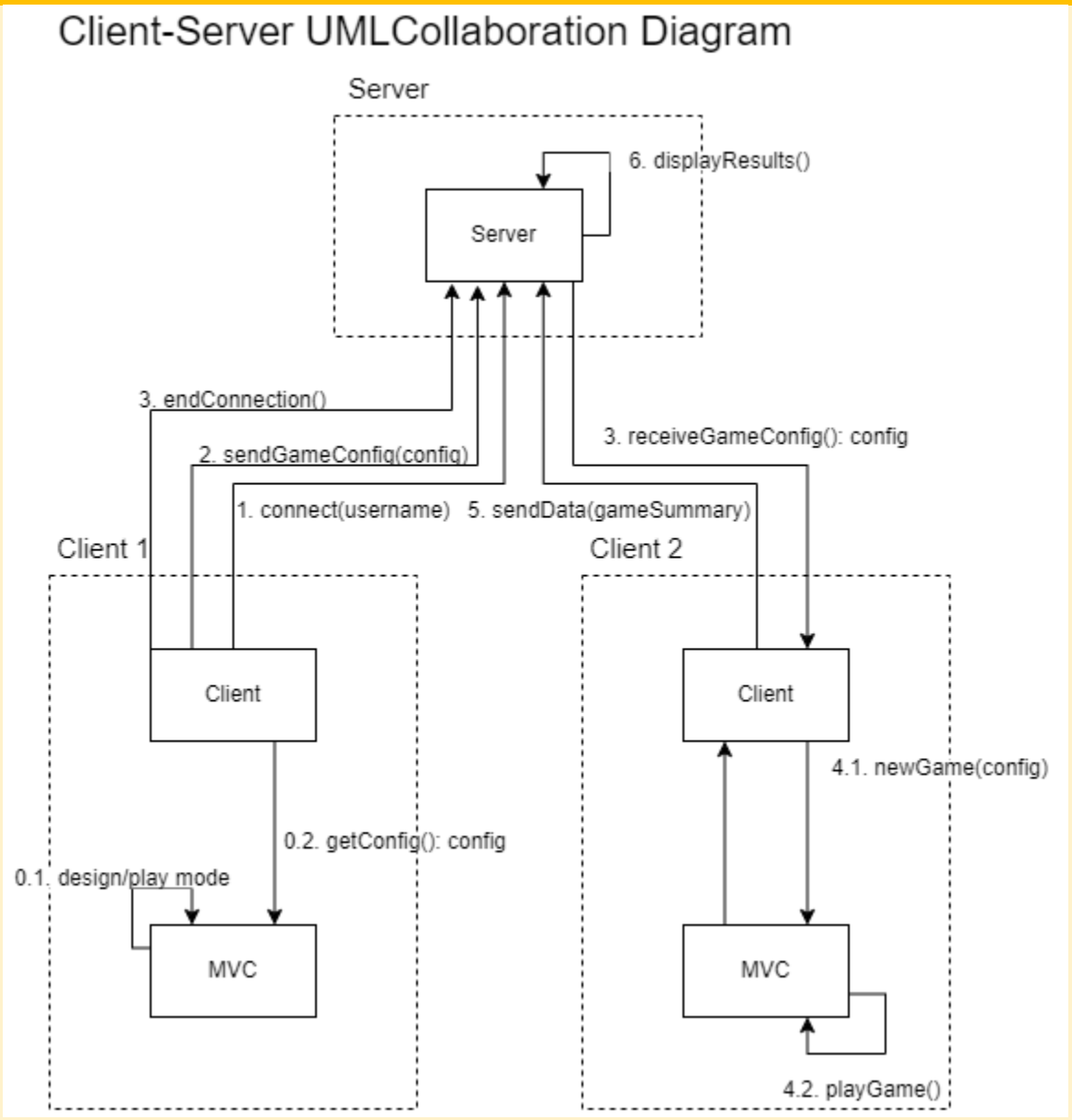
Team:
Dylan Boyling - ID: 041042484

Battleship Proposal

C/S Architecture

1. Client-Server Model (UML Collaboration Diagram)

Collaboration Diagram (Client-Server)



1. Protocol Proposal

Example (using the string definition mentioned in the A21 specification)

<Separator> = comma (,)

Protocol P0: Client is connecting to server.

Format for request:<username>

<username> = string to identify the user

Protocol P1: Client has connected to the server

Format for response:<userId>

<userId> = unique integer to identify the user when sending requests in the future, i.e. game data or configurations

Protocol P1: Client is sending game configuration.

Format for request: <userId><separator><dim><separator><dataConfig>

<userId> = client's unique identification ID on the server

<dim> = integer from 2 to 20, in increments of 2

<dataConfig> = <ship><separator><ship><separator>…etc, list of ship configurations for the player's board

<ship> = <x><separator><y><separator><length><separator><isHorizontal> which contains the x, y coordinates for a ship of a given length, and 1/0 to indicate if the ship is horizontal/vertical.

NOTE: Might change to just outputting grid to a string e.g. 0002233300100.. etc and create the "Ship" object by using the pattern. Such as, if "22" is a sequence then it's a size 2 ship that is horizontal. Ship object is primarily used for displaying feedback to the user in the form of "Size of X sunk by player!" when all of it's squares are sunk. Current configuration may be a bit long so will look into different strategies.

Protocol P2: Server is sending game configuration after a request.

Format for response: <dim><separator><dataConfig>

<dim> = integer from 2 to 20, in increments of 2

<dataConfig> = <ship><separator><ship><separator>…etc, list of ship configurations for the player's board

<ship> = <x><separator><y><separator><length><separator><isHorizontal> which contains the x, y coordinates for a ship of a given length, and 1/0 to indicate if the ship is horizontal/vertical.

Game configuration will randomly select one stored on the server and send it but may allow selection of stored game servers if it is feasible.

Protocol P3: Client is sending game data to server (username, points, time) to server

Format for request: <userId> <separator><username><separator><playerPoints><separator><systemPoints>

<userId> = client's unique identification ID on the server

<username> = string to identify the user

<playerPoints> = Points scored by player

<systemPoints> = Points scored by system

NOTE: My Battleship configuration does not currently have a timing mechanism. May include one and if so, <separator><timePlayed> will be appended to the request format. timePlayed will be in ms.

2. Database Integration (Bonus)

Usernames can be stored in one table, a unique ID given for the user in the form of an integer and a username can be stored as a VARCHAR[10]. Game configurations can be stored in a table with unique IDs for each configuration. Game configurations can also store the user ID it belongs to as an integer (read: foreign key to username table) as well as a VARCHAR[100] containing the full configuration string. Additionally, on competition of a game the user may INSERT data into database after the completion of a game which will store the game summary with a unique ID for that summary, the user's unique ID, the player's points (integer), the system's points (integer), and the time played (integer).

After the player's username has been entered into the database, if the same user sends another username request and the user already has an entry in the database, then it may be UPDATED to the new value. The database should not allow duplicate configurations to be stored by the same user to prevent one user from flooding the database with useless configurations.

Algonquin College
Summer, 2023