Use Case 3: Play Network Game

Author: Yasin Garip

Characteristic Information

Goal in Context: The user can successfully play a network game with remote players

Scope: Network and Game

Level: Summary

Primary Actor: Player

Stakeholders & Interests:

- Player: Wants to play Scrabble according to the game's rulebook with remote players

Preconditions: Server and client are working and running

Success Guarantee (Postconditions): User was able to play network game with other remote human players

Trigger: User clicks on Button 'Play with Friends'

Main Success Scenario

1. User chooses to host game *Include UC 3.1: Host Game*
2. User waits for remote players to join game
3. User plays Scrabble: *Extends UC 2: Play Scrabble*
4. User gets game results *UC 5: Display statistics*

Extensions

*a. At any time, System fails:

1. System reopens system, logs in, and requests recovery of prior state.
2. System reconstructs prior state.
   2a. System detects anomalies preventing recovery: System signals error and records the error.
3. Warning sign that the game crashed appears.

*b. Server/Client fails:

1. Application returns to main menu

Sub-Variations

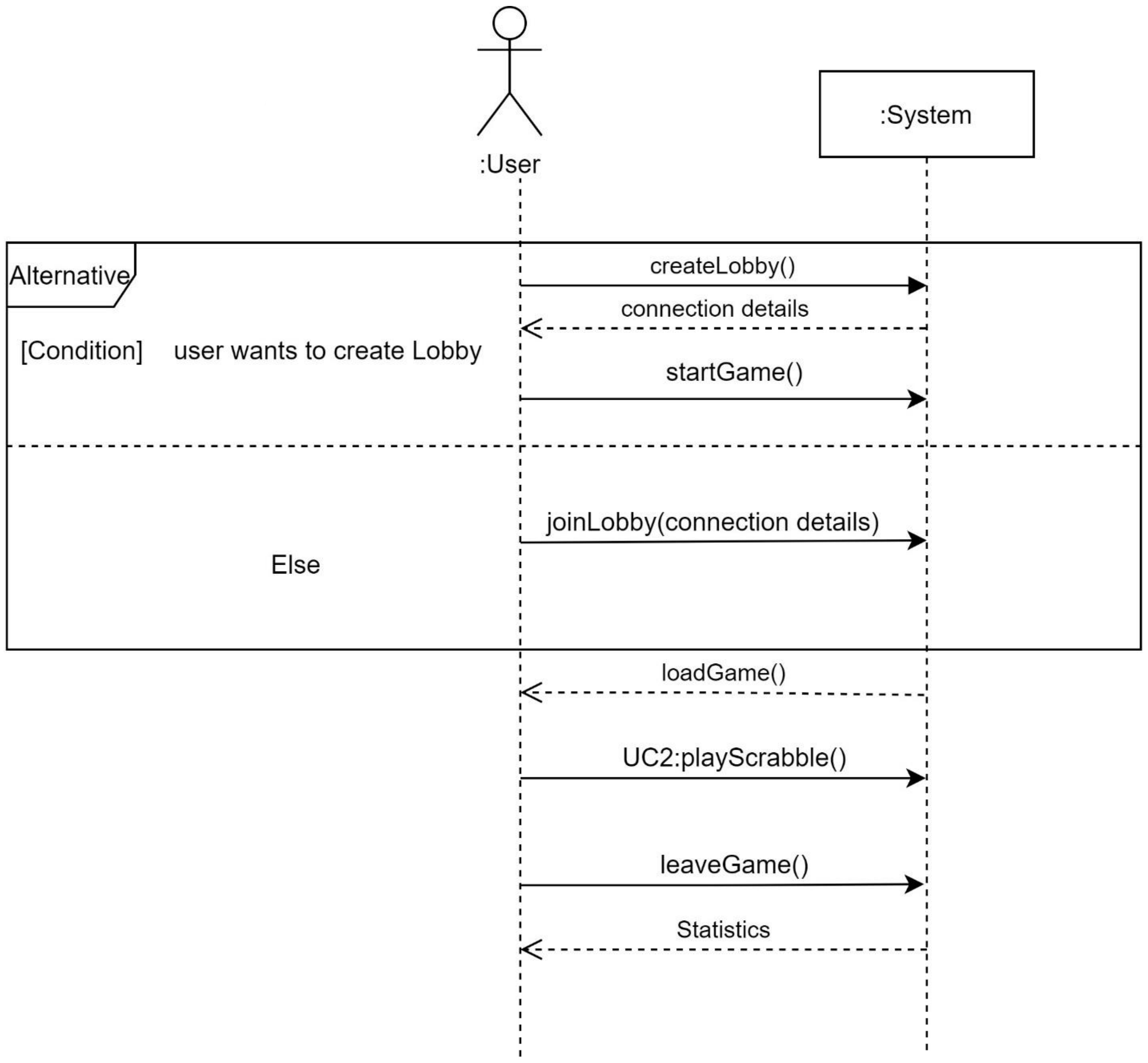
1. User can use join a game instead of creating game lobby: *Include UC 3.2: Join Game*
2. Users can send messages over chat application: *Include UC 3.3: Chat*
3. User leaves game lobby while game still goes on: *Include UC 3.4: Leave Game*

Due Date

10.05.2021

System Sequence Diagram

Author: Yasin Garip



Operation Contracts

Author: Yasin Garip

Contract 3.1: createLobby

Operation: createLobby() : void

Cross References: UC 3.1: Host Game

Preconditions:

- User was in game lobby

Postconditions:

- Server instance s was created
- Player p was created and associated with server s
- Connection details were sent back to user

Contract 3.2: joinLobby

Operation: joinLobby(connection details) : void

Cross References: UC 3.2: Join Game

Preconditions:

- User u1 created a lobby and started a server s
- User u2 knew the connection details of u1
- u1 was associated with server s

Postconditions:

- u2 was associated with s

Contract 3.3: LeaveGame

Operation: leaveGame() : void

Cross References: UC 3.4: Leave Game

Preconditions:

- Server s was running and had association g with instance of Game
- Player p was associated with s and g

Postconditions:

- p was disassociated with s and g