Networked Rummy

Brian Herman
Diana Carrillo
Spencer Cleveland
Rob Hayhurst
Nakul Saunshi

Software Design Document

Table of Contents

Purpose
Scope
System Overview
Architectural Design Diagrams
Decomposition Description
Data Dictionary

Purpose

The purpose of this project is to create a networked rummy card game with chat.

Scope

This project is to allow 2-6 players to play rummy together while using their own computers. They will communicate over the network using sockets. They will also have the ability to chat with each other as a group or send private messages.

System Overview

We have twelve classes. A class to store cards, a client JFrame class, a Deck class to store deck information, an Envelope class to send messages from the server to the client and vice versa, a Game class to store game information, a game JFrame class to display the game information, a Player class to store player information, a Server class to start the server thread class, accept incoming connections, and display a Server gui, a server thread class to accept each incoming connection, and a switcher class to switch between server and client.

3. System Architecture

3.1 Architectural Design

Card Class

Method	**Description**
constructor	Creates a new instance of the card class
getSuit	Returns the suit of the card
getRank	Returns the rank of the class.
toString	Returns the string representation of a card

Client Class

Method	**Description**
constructor	Creates a new instance of the client class
connect	Connects to the Server
leave	leaves the server

send(Envelope m)	Sends an envelope to the server
send(Game g)	Sends a gamestate to the server
listen	Creates a new thread to listen to the server.
updateGame	Updates the game state.
Subclass	
ClientActionListener	Handles events from the client
ClientThread	Handles server messages

Deck Class

Method	**Description**
constructor	Creates a new instance of the deck class
generateDeck	Shuffles the cards and generates a new deck.
add	Adds a card to the deck
remove	removes a card from the deck
deal	Deals cards from the deck

Envelope Class

Method	**Description**
constructor	Creates a new instance of the envelope class
sender	Returns the name of the sender
message	returns the message
recipients	returns an arraylist of recipient names

Game Class

Method	**Description**
constructor	Creates a new instance of the game class
discard	Returns the discard deck
addPlayer	Adds a player to the game
deal	deals cards to a player

getPlayer	Returns a player
getPlayers	Returns all the players
id	Returns the game's identifier

Server Class

Method	**Description**
constructor	Creates a new instance of the Server class and a gui for the server class
getUsers	Returns the users connected to the server.
send(Game g)	Sends a gamestate to each of the clients
send(Envelope m)	Sends an envelope to each of the clients based on the recipient arraylist.
remove	Removes a user from the threads Data Structure so you don't send messages to a closed I/O stream

4.1 Data Description

The GUI elements were implemented using the Java Swing libraries in order to facilitate the event-driven paradigm for this project. The Client JFrame object's GUI contains the appropriate eventhandlers for multiple user options stored in the form of menus(JMenuItems, JMenus, JMenuBar) allowing the user connect/leave/quit, also asks the user to enter in his/her handle name and the server they wish to connect to (IP address). The Server JFrame object's GUI is mainly used as a display (shows the port number, host name and host IP address), while the back-end processes sending/receiving messages to/from the appropriate parties.

4.2 Data Dictionary: See JavaDocs

5. Component Design: See JavaDocs