Przykładowa klasa:

```
public class MakaoProtocol implements Serializable {
    public static final int LOGIN = 1;
    public static final int PLAY_CARD = 2;
    public static final int DRAW_CARD = 3;
    public static final int END_TURN = 4;

    private int operationCode;

    private String username;
    private Card playedCard;

    public MakaoProtocol(int operationCode) {
        this.operationCode = operationCode;
    }

    public int getOperationCode() {
        return operationCode;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public Card getPlayedCard() {
        return playedCard;
    }

    public void setPlayedCard(Card playedCard) {
        this.playedCard = playedCard;
    }
}
```

Opis wymiany informacji klient-serwer

Logowanie (LOGIN):

Klient wysyła żądanie logowania, zawierające kod operacji LOGIN oraz nazwę użytkownika. Serwer sprawdza poprawność danych logowania, w tym unikalność nazwy użytkownika. Serwer odpowiada klientowi, informując go o sukcesie lub niepowodzeniu logowania, a także przekazuje informacje o aktualnym stanie gry.

Zagrana karty (PLAY_CARD):

Klient wysyła żądanie zagranej karty, zawierające kod operacji PLAY_CARD oraz informacje o zagranej karcie.
Serwer sprawdza, czy ruch jest zgodny z zasadami gry (np. czy karta może zostać zagrana) i informuje o sukcesie lub ewentualnym błędzie.
Jeśli ruch jest poprawny, serwer przesyła informacje o wykonanym ruchu do wszystkich pozostałych klientów, aktualizując ich widok gry.

Pobranie karty (DRAW_CARD):

Klient wysyła żądanie pobrania karty z talii, zawierające kod operacji DRAW_CARD.
Serwer sprawdza dostępność kart w talii, losuje kartę i przesyła ją do klienta.
Jeśli talia jest pusta, serwer informuje klienta o braku dostępnych kart i może przetasować stos odrzuconych kart, aby utworzyć nową talie.

Zakończenie tury (END_TURN):

Klient wysyła żądanie zakończenia tury, zawierające kod operacji END_TURN.
Serwer sprawdza, czy klient ma prawo zakończyć turę (np. czy poprzedni ruch był poprawny), a następnie przekazuje turę następnemu graczowi.
Serwer informuje wszystkich klientów o zakończeniu tury, aktualizując ich widok gry i podając informacje o nowym graczu.

Serwer -> Klient:

Informacje o grze:

Serwer regularnie przesyła informacje o aktualnym stanie gry do wszystkich klientów, np. karty na stosie, aktualny gracz, liczba kart przeciwników.
Informacje te są przesyłane w ramach kodu operacyjnego UPDATE_GAME_STATE lub podobnego.
W przypadku zmiany stanu gry (np. kiedy ktoś zagra kartę), serwer natychmiast przekazuje te informacje wszystkim klientom, aby utrzymać synchronizację.

Ruch przeciwnika:

Po otrzymaniu poprawnego ruchu od jednego z klientów, serwer przekazuje informacje o tym ruchu do pozostałych klientów.
Informacje zawierają wykonany ruch, np. nazwę gracza, zagranej karty, a także aktualny stan gry po wykonanym ruchu.
Dzięki temu wszyscy gracze utrzymują wspólny widok gry.

Informacje o dołączeniu/opuszczeniu gry:

Serwer informuje wszystkich klientów o dołączeniu nowego gracza lub opuszczeniu gry przez jednego z uczestników.

Te informacje mogą zawierać nazwę gracza, który dołączył lub opuścił grę, oraz aktualny stan gry.

Zakończenie gry:

Gdy gra kończy się (np. jeden z graczy wygrywa), serwer wysyła powiadomienie o zakończeniu gry do wszystkich klientów.
Informacje te zawierają wyniki gry, takie jak pozycje graczy i liczba punktów, oraz mogą składać się z kodu operacyjnego GAME_OVER.