1. Comunicação TCP no Jogo de Truco

Optamos por utilizar o protocolo TCP para viabilizar a comunicação multiplayer em nosso jogo de truco. Ao iniciar o jogo, oferecemos duas opções: Host e Cliente.

Os clientes atuam como ouvintes (listeners) dos pacotes, executando ações na interface do usuário (UI) com base nas informações recebidas. No entanto, ao optar por iniciar o jogo como Host, implementamos uma abordagem em duas threads.

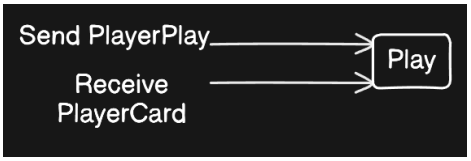
Iniciamos uma thread chamada "Server" que abrange toda a lógica do truco e o gerenciamento do envio de pacotes entre os clientes. Simultaneamente, iniciamos outra thread como cliente, que se conecta em loopback com o servidor em uma thread separada. Essa abordagem permite centralizar toda a lógica do jogo na thread do servidor, garantindo que apenas uma instância execute a lógica do truco. Todas as janelas estão vinculadas à thread do cliente para receber pacotes e executar ações na UI com base nas informações recebidas.

1.1 Definindo a Estrutura do Jogo: Round, Turn e Play

No contexto do nosso jogo de truco, é crucial compreender a estrutura de Round, Turn e Play. Essa abordagem modular de Play, Turn e Round contribui para uma experiência de jogo clara e estruturada, tornando o acompanhamento do progresso e das ações dos jogadores mais intuitivo.

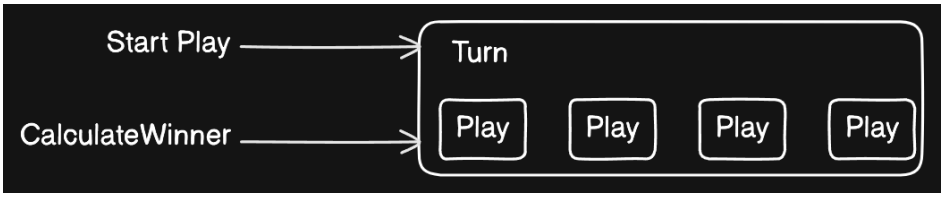
1.1.1 Play

Refere-se à ação específica de um jogador durante um Turn. Pode ser uma jogada de carta, um pedido de truco, ou qualquer ação relevante. Cada Turn é composto por uma Play de cada jogador.



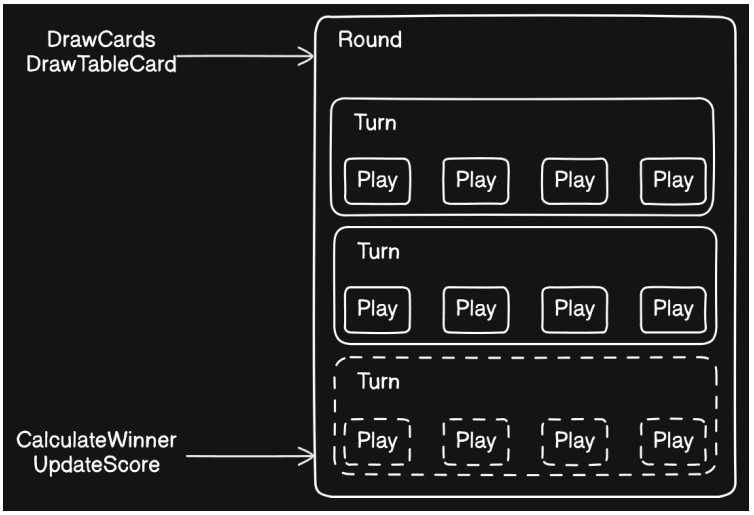
1.1.2 Turn

Representa um ciclo de jogadas, onde cada jogador realiza uma Play. Durante um Turn, os participantes executam ações, como jogar cartas ou tomar decisões estratégicas. Cada conjunto de 3 ou 2 Turns culmina em um Round.



1.1.3 Round

Constitui um agrupamento de 3 ou 2 Turns, dependendo da dinâmica do jogo. Após cada Round, é possível determinar qual foi a dupla vencedora.



Tipos de pacotes

Em nosso jogo, temos 8 tipos de pacotes para definir a comunicação:

2.1 StartGame:

- Descrição: Enviado no início do servidor para todos os clientes, informando o ID do jogador e a qual time pertence.
- Campos: int playerId, int teamId.

2.2 StartRound:

- Descrição: Enviado no início de cada round para todos os jogadores, revelando a carta que inicia a mesa e as cartas na mão de cada jogador.
- Campos: Card tableCard, std::vector<Card> handCards.

2.3 EndRound:

- Descrição: Enviado ao final de um round, contendo o ID do time vencedor, o valor da rodada e os scores atualizados de ambos os times.
- Campos: int winnerTeamId, int stakes, int team0Score, int team1Score.

2.4 EndTurn:

- Descrição: Enviado ao final de cada turn, indicando o ID do time vencedor e o ID do jogador responsável pela carta vencedora.
- Campos: int winnerTeamId, int winnerPlayerId.

2.5 PlayerPlay:

- Descrição: Enviado quando é a vez de um jogador, contendo o ID do jogador e um booleano indicando se é possível solicitar truco naquele momento.
- Campos: int playerId, int canRequestTruco.

2.6 PlayerCard:

- Descrição: Enviado pelo jogador indicando qual carta deseja jogar e se ela está coberta ou não.
- Campos: Card card, int playerId, int isCovered.

2.7 Truco:

- Descrição: Gerencia a lógica do truco, incluindo o ID do jogador solicitante, o ID do time que deve responder e um enum indicando se o pedido foi aceito, negado ou se a aposta deve ser aumentada.
- Campos: int requesterId, int responseTeamId, TrucoResult result.

2.8 ElevenHand:

- Descrição: Enviado quando um time atinge 11 pontos, apresentando a carta inicial da mesa, as cartas na mão do jogador e as cartas na mão do aliado.
- Campos: Card tableCard, std::vector<Card> handCards, std::vector<Card> partnerHand.

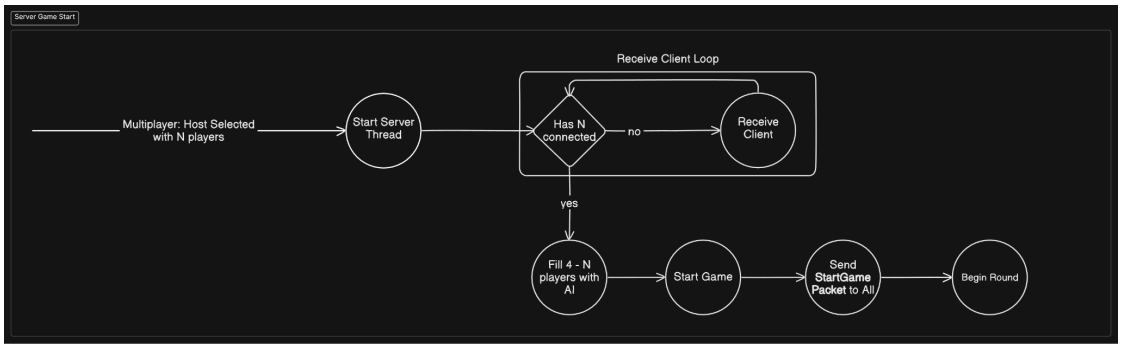
2.9 ElevenHandResponse:

- Descrição: Resposta do time com 11 pontos indicando se irá aceitar ou recusar jogar a rodada subsequente.
- Campos: int response;

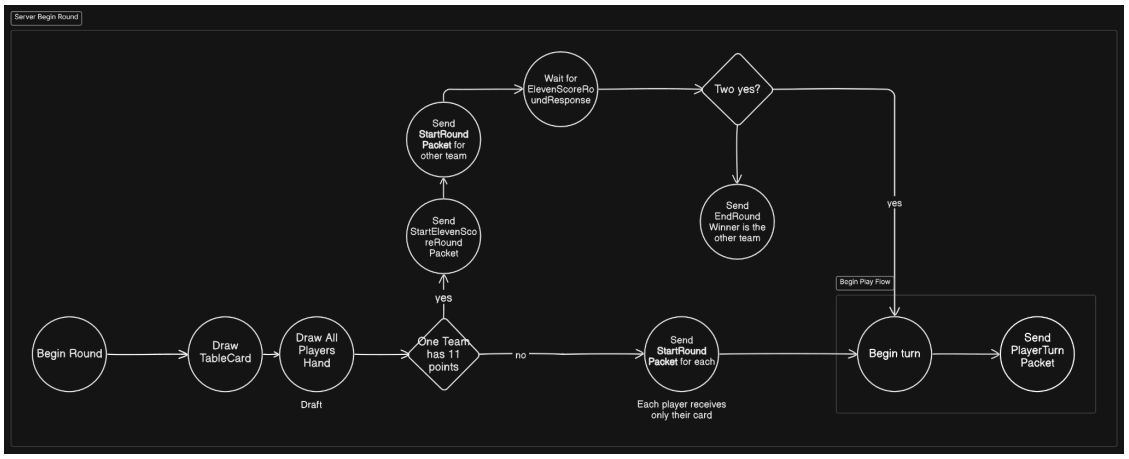
3. Servidor

Conforme mencionado anteriormente, o servidor é responsável por abrigar a lógica do truco. A seguir, apresentamos diagramas de fluxo que ilustram a implementação do código.

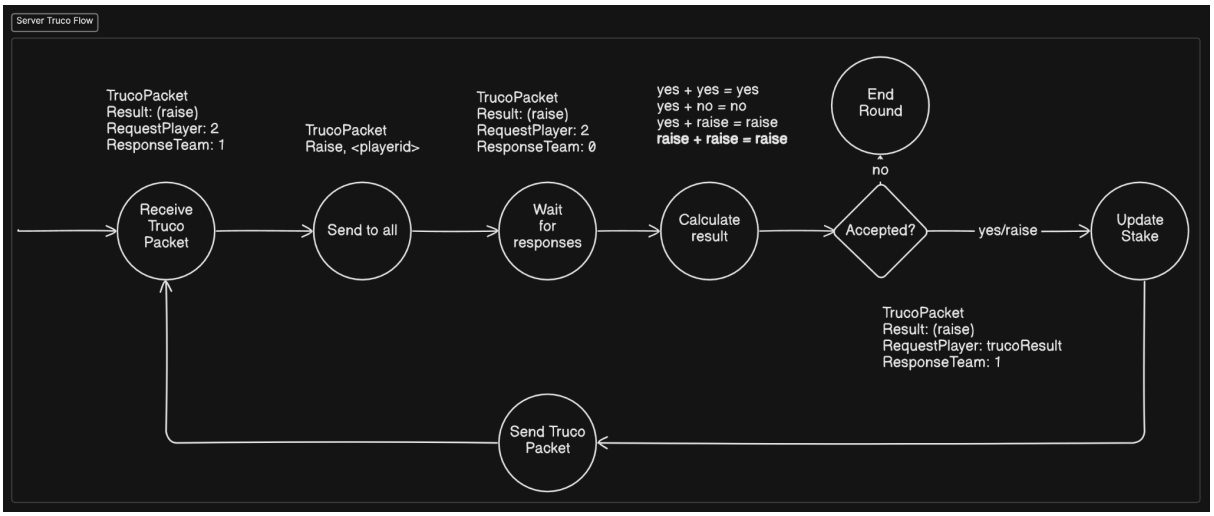
3.1 Fluxo do servidor para início de jogo



3.2 Fluxo do servidor para início de rodada



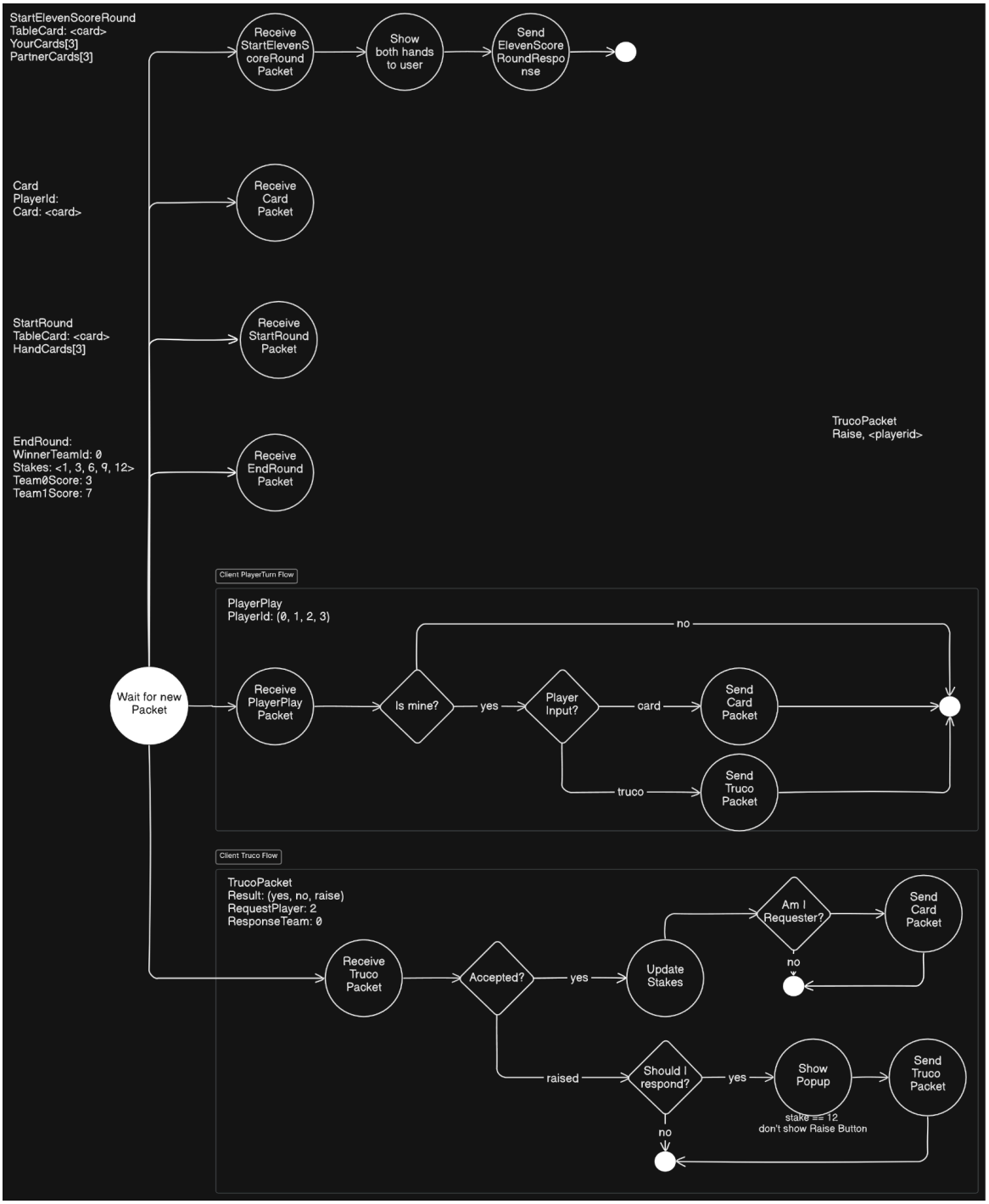
3.3 Fluxo do servidor para pedidos de truco



4. Cliente

O fluxo do cliente está intrinsecamente ligado à atualização e recebimento de entradas pela interface do usuário (UI). Sua lógica fundamental consiste em um listener que aguarda a chegada de pacotes, representados pelos círculos brancos abaixo (todos os demais círculos pequenos indicam o retorno ao estado de "Wait for new Packet").

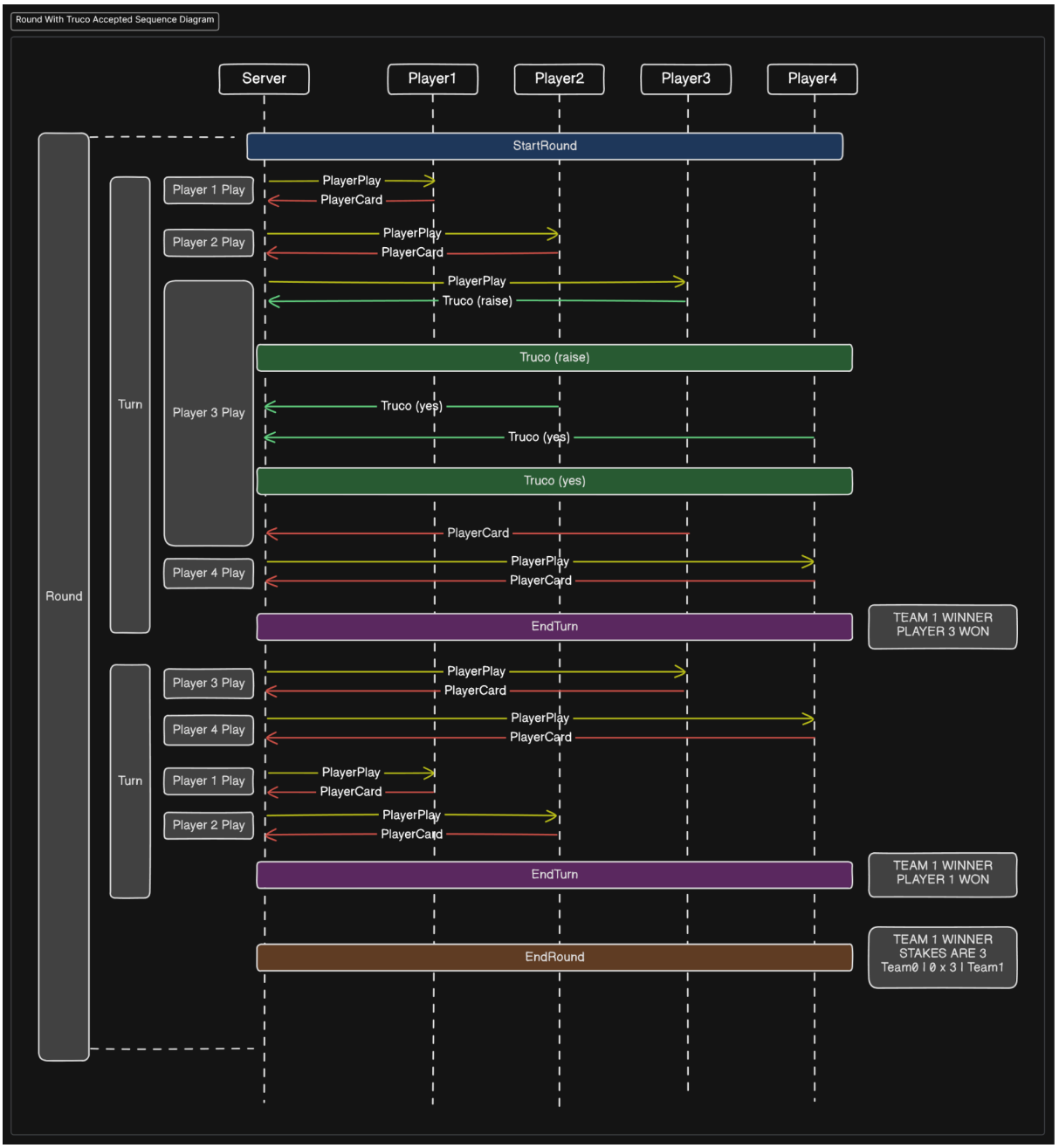
Embora o foco desse fluxo destaque predominantemente o papel do servidor, é crucial ressaltar que a partir dessas interações, podemos efetuar atualizações na UI. Por exemplo, ao receber um pacote de carta de outro jogador, identificado como "Receive Card Packet", a subsequente ação seria a atualização da UI, exibindo a carta jogada na mesa.



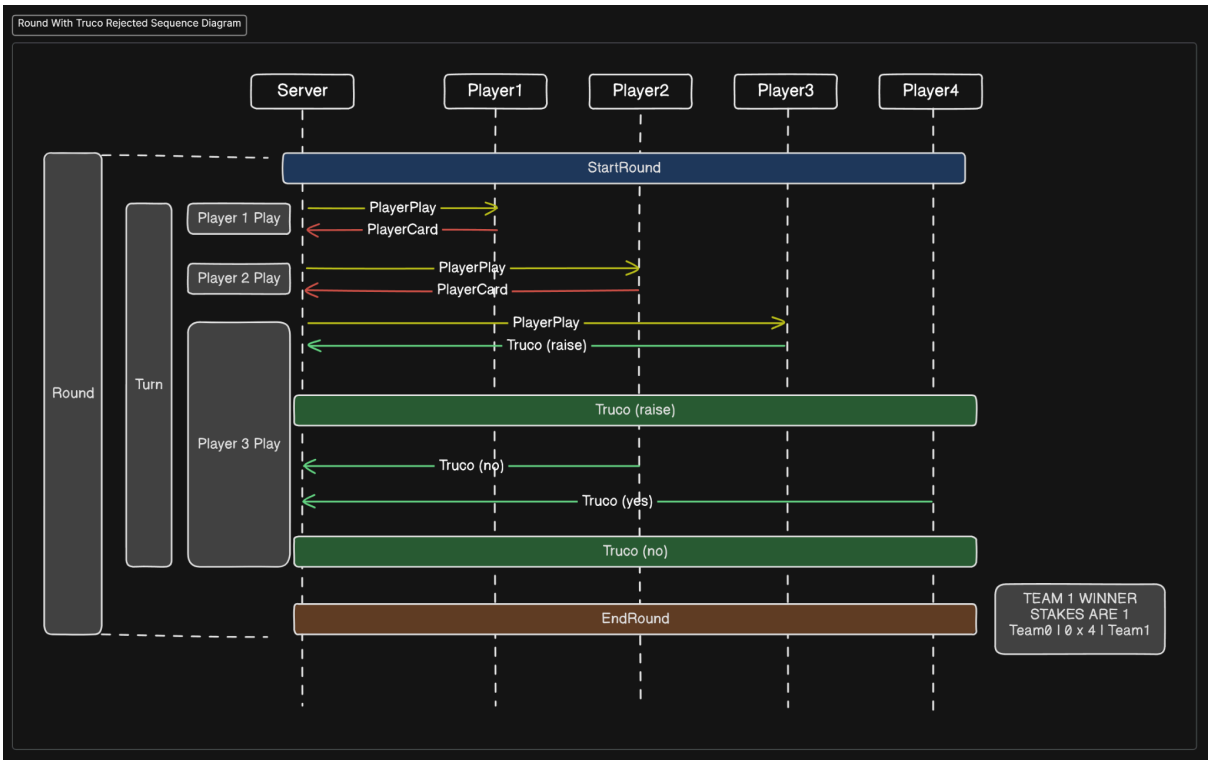
5. Diagrama de sequência

A seguir, apresentam-se alguns diagramas de sequência que exemplificam a distribuição de pacotes no contexto do jogo de truco, abordando diversas situações. É relevante observar que as barras coloridas de cor sólida representam a distribuição de pacotes para todos os clientes, servindo como uma forma mais concisa de indicar um broadcast.

5.1 Diagrama de fluxo com um caso de truco aceito



5.2 Diagrama de fluxo com um caso de truco recusado



5.3 Diagrama de fluxo com um caso de truco aumentado