PPC DIAGRAM : HANNABIS

**Explanation:**

* **Game Process:**
  + Manages the game session, deck, and tracks suits in construction.
  + Utilizes a message queue for communication with player processes.
* **Player Processes:**
  + Interact with users, the game process, and other player processes.
  + Communicate with the game process via message queue.
  + Player processes may use sockets for inter-process communication.
* **Message Queues:**
  + Used for communication between game and player processes.
  + Messages exchanged include player actions, game state updates, etc.
* **Sockets:**
  + Used for communication between player processes.
  + Facilitates exchange of information such as player actions, game state updates, etc.
* **Shared Memory:**
  + Shared memory contains information on tokens and suits in construction.
  + Accessed by both game and player processes.
* **Signals:**
  + Signals are exchanged between processes to notify end-of-game events or other critical situations.
  + Handlers for signals are defined in each process to respond appropriately.
* **Synchronization Primitives:**
  + Utilized to protect access to shared resources (e.g., shared memory) or count resources.
  + Includes locks, semaphores, or other synchronization mechanisms.

### Exchange Protocol for Hannabis Message Queue:

#### Message Types:

1. \*\*InfoMessage:\*\*

- Structure: `{"type": "info", "content": "The card is your only red card."}`

- Description: Player provides information about a card in another player's hand.

2. \*\*PlayCardMessage:\*\*

- Structure: `{"type": "play", "card": {"value": 3, "color": "blue"}}`

- Description: Player wants to play a card from their hand.

3. \*\*DrawReplacementCardMessage:\*\*

- Structure: `{"type": "draw\_replacement\_card"}`

- Description: Player requests to draw a replacement card after playing a card.

4. \*\*EndTurnMessage:\*\*

- Structure: `{"type": "end\_turn"}`

- Description: Player signals the end of their turn.

#### Rules:

1. \*\*InfoMessage Rules:\*\*

- Can only be sent if the player has enough information tokens.

- Must specify the type of information (color or value) and the card affected.

2. \*\*PlayCardMessage Rules:\*\*

- Can only be sent if it's the player's turn.

- Must specify the card to be played, including its value and color.

- The game process will validate if the play is legal.

3. \*\*DrawReplacementCardMessage Rules:\*\*

- Can only be sent after successfully playing a 5.

4. \*\*EndTurnMessage Rules:\*\*

- Must be sent after the player has completed their turn.

#### Example Usage:

- Player A wants to inform Player B about their blue cards:

```json

{"type": "info", "content": "The card is your only blue card."}

```

- Player B wants to play a blue 3:

```json

{"type": "play", "card": {"value": 3, "color": "blue"}}

```

- Player A successfully plays a 5 and wants to draw a replacement card:

```json

{"type": "draw\_replacement\_card"}

```

- Player B signals the end of their turn:

```json

{"type": "end\_turn"}

```

**Socket Exchange Protocol for Hannabis:**

**Message Types:**

1. **InitiateConnection:**
   * Structure: {"type": "initiate\_connection", "player\_id": 1}
   * Description: Player initiates a connection with the game process.
2. **InfoMessage:**
   * Structure: {"type": "info", "content": "The card is your only red card."}
   * Description: Player provides information about a card in another player's hand.
3. **PlayCardMessage:**
   * Structure: {"type": "play", "card": {"value": 3, "color": "blue"}}
   * Description: Player wants to play a card from their hand.
4. **DrawReplacementCardMessage:**
   * Structure: {"type": "draw\_replacement\_card"}
   * Description: Player requests to draw a replacement card after playing a card.
5. **EndTurnMessage:**
   * Structure: {"type": "end\_turn"}
   * Description: Player signals the end of their turn.
6. **GameUpdateMessage:**
   * Structure: {"type": "update", "content": "It's now Player 2's turn."}
   * Description: Game process sends updates to players about the game state.

**Rules:**

1. **InitiateConnection Rules:**
   * Sent by the player to initiate the connection.
   * Must include a unique player\_id for identification.
2. **InfoMessage Rules:**
   * Can only be sent if the player has enough information tokens.
   * Must specify the type of information (color or value) and the card affected.
3. **PlayCardMessage Rules:**
   * Can only be sent if it's the player's turn.
   * Must specify the card to be played, including its value and color.
   * The game process will validate if the play is legal.
4. **DrawReplacementCardMessage Rules:**
   * Can only be sent after successfully playing a 5.
5. **EndTurnMessage Rules:**
   * Must be sent after the player has completed their turn.
6. **GameUpdateMessage Rules:**
   * Sent by the game process to inform players about the current game state.
   * Contains important updates or notifications.

**Example Usage:**

* Player initiates a connection:

json

 {"type": "initiate\_connection", "player\_id": 1}

 Player provides information about a red card:

json

 {"type": "info", "content": "The card is your only red card."}

 Player plays a blue 3:

json

 {"type": "play", "card": {"value": 3, "color": "blue"}}

 Player successfully plays a 5 and wants to draw a replacement card:

json

 {"type": "draw\_replacement\_card"}

 Player signals the end of their turn:

json

 {"type": "end\_turn"}

 Game process sends an update:

json

{"type": "update", "content": "It's now Player 2's turn."}

Interaction thread

**Message queue**

**socket**

**socket**

**socket**

**Player**

**Process**

**Player**

**Process**

**Player**

**Process**

**Game Process**