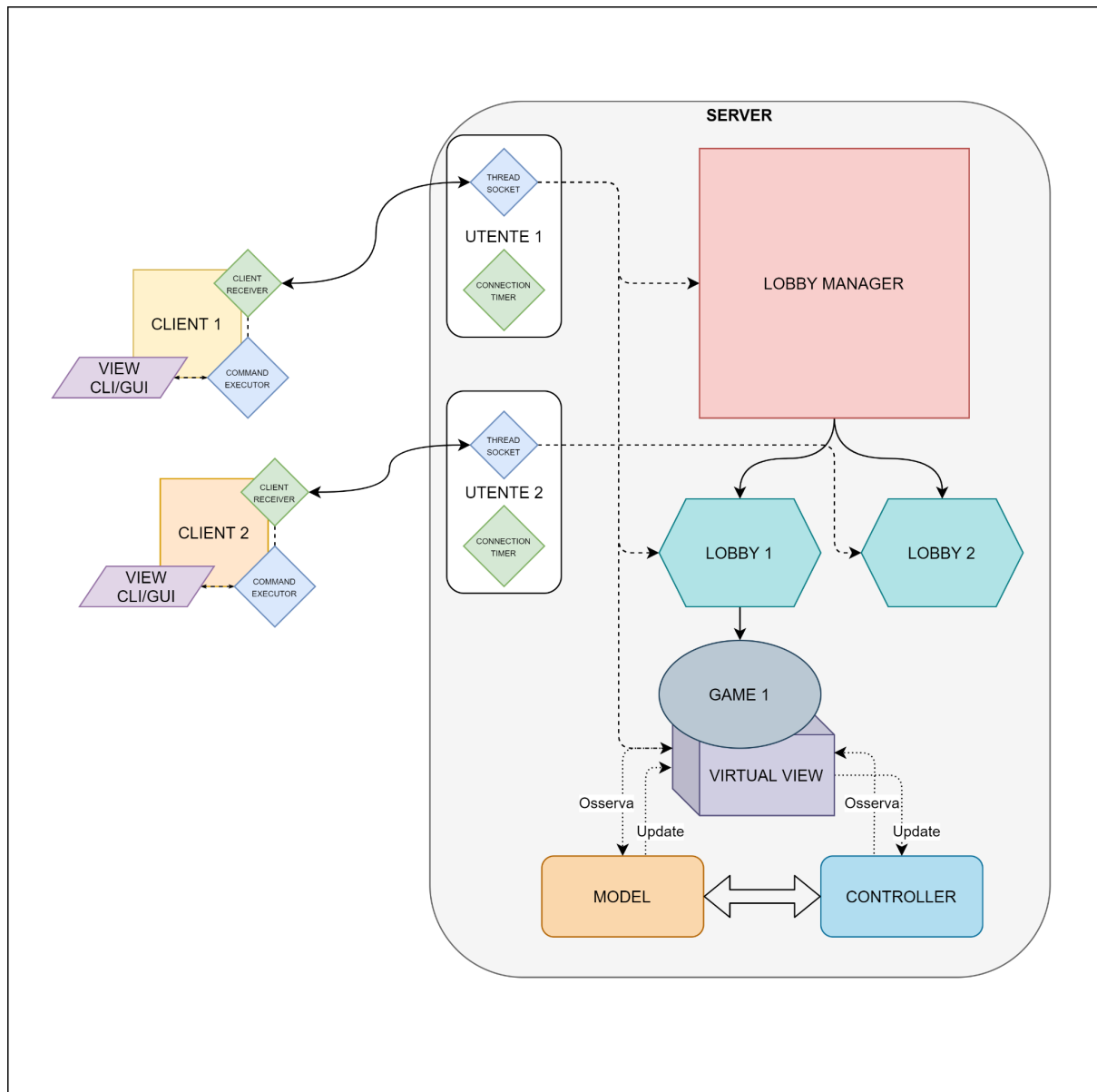# NETWORK STRUCTURE

Connection Timer:

      After *T*ms sends a Ping message to the user.

      Disconnects the user after *N* unreceived Pongs (disc. time = $T * N$).

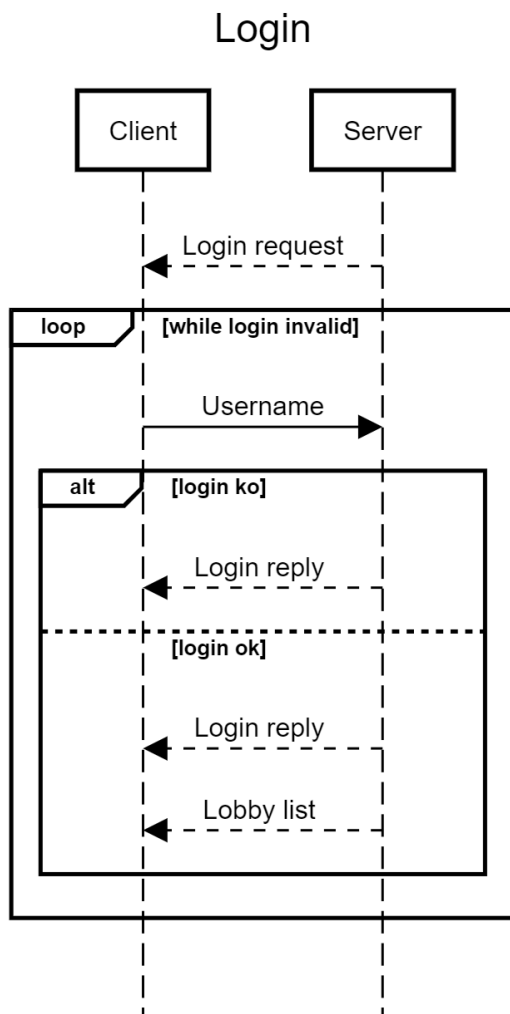   Every correctly received command also counts as a Pong command.

Command Executor:

      Is an ExecutorService that creates a SingleThreadExecutor()
      in order to split the network from the view's execution that could interrupt
      the system.

JSON - SEQUENCE DIAGRAM

GAME INITIALIZATION

Login

```
   Client              Server
  ┌──────┐            ┌──────┐
  │Client│            │Server│
  └──────┘            └──────┘
     │                   │
     │◄ ─ ─ Login request ─ ─│
     │                   │
 ┌loop│  [while login invalid]│─────────┐
 │   │                   │              │
 │   │────── Username ──────►│           │
 │   │                   │              │
 │ ┌alt│  [login ko]       │──────────┐ │
 │ │  │                   │          │ │
 │ │  │◄ ─ ─ Login reply ─ ─│          │ │
 │ │  │                   │          │ │
 │ ├─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ┤ │
 │ │   [login ok]        │          │ │
 │ │  │                   │          │ │
 │ │  │◄ ─ ─ Login reply ─ ─│          │ │
 │ │  │                   │          │ │
 │ │  │◄ ─ ─ Lobby list ─ ─ │          │ │
 │ └──│───────────────────┘──────────┘ │
 └────│───────────────────┴────────────┘
     │                   │
     │                   │
```

[EchoServerClientHandler.java]

Login request (server)
Command.REPLY\n
info: "Inserire username"

Username (client)
Command.LOGIN\n
JSON:{ String nickname }
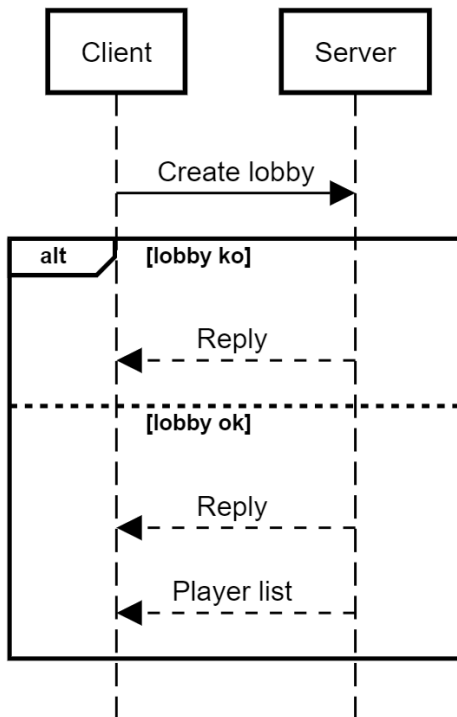
Login reply (server)
-On Error:
Command.REPLY\n
info: "Username already in use, insert a valid nickname: "

-On Ok:
Command.LOGIN\n
info: "Welcome to the server"

lobby list (server)
Command.Lobby_List\n
LobbyListMessage.java
    List<LobbyInfo>:
        { [lobbyName, Owner,MaxPlayer,
        NumConnected, isFullisClosed],
    [...], [...] }

Lobby Creation

[LobbyManager.java]

Create Lobby (client)
Command.CREATE_LOBBY\n
JSON:{   String lobbyName
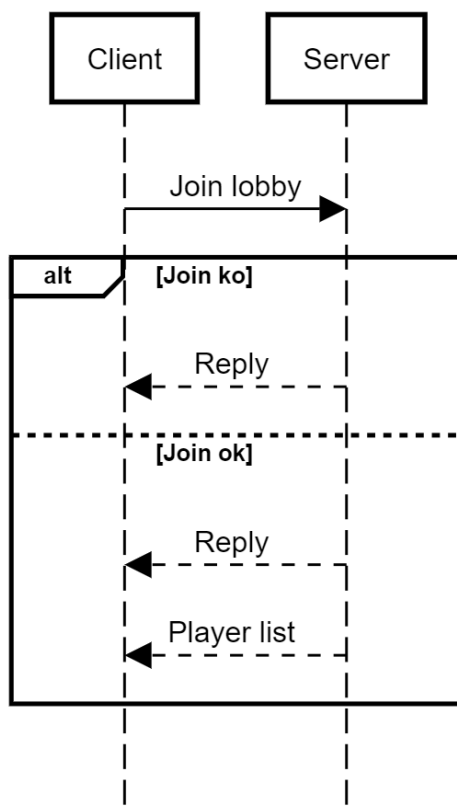   int numOfPlayer }

Reply (server)
-OnError:
Command.REPLY\n
info: "The lobby already exists"

-OnOk:
Command.JOIN_LOBBY\n

Player list(server)
Command.PLAYER_LIST\n
StringMessage.java
data: JSON{ nick1, nick2, ..}

Join lobby

[Lobby.java]

Join lobby(client)
Command.JoinLobby\n JoinLobbyMessage.java
JSON:{ "LobbyName" : string }

Reply (server)
-OnError:
Command.REPLY\n
info: Error

Error 1: The lobby is full
Error 2: The selected lobby doesn't exist
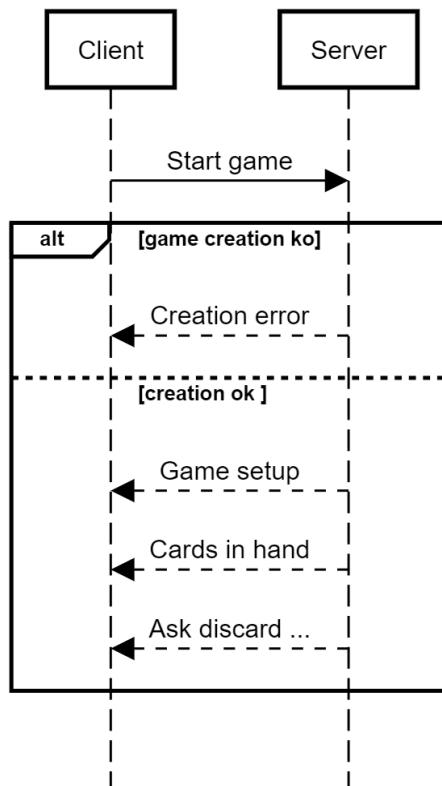Error 3: The game is already started
ecc.

-OnOk:
Command.JOIN_LOBBY\n

Player list(server)
Command.PLAYER_LIST\n
JSON:{ nick1, nick2,..}

GAME PHASE

Before playing

```
┌────────┐        ┌────────┐
│ Client │        │ Server │
└────────┘        └────────┘
    │                  │
    │   Start game     │
    │─────────────────>│
┌───────────────────────────┐
│ alt  [game creation ko]   │
│   │                  │    │
│   │  Creation error  │    │
│   │<─ ─ ─ ─ ─ ─ ─ ─ ─│    │
│ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─│
│      [creation ok ]       │
│   │                  │    │
│   │   Game setup     │    │
│   │<─ ─ ─ ─ ─ ─ ─ ─ ─│    │
│   │   Cards in hand  │    │
│   │<─ ─ ─ ─ ─ ─ ─ ─ ─│    │
│   │  Ask discard ... │    │
│   │<─ ─ ─ ─ ─ ─ ─ ─ ─│    │
└───────────────────────────┘
    │                  │
```

[Lobby.java]

Start game(client)
Command.START_GAME\n

Creation error(server)
Command.GAME_CREATION_ERROR\n
info: Error

[VirtualView.java]

Game Setup(server)
Command.GAME_SETUP\n
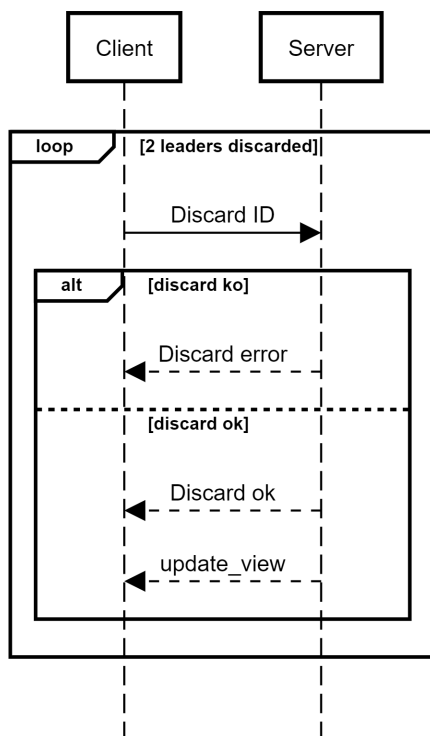JSON:{ List<Integer> cardGridIds,
          List<String> nicknames,
       List<ResourceContainer> marketSetup }

Cards in hand(server)
JSON: { List<Integer> cards_iDs }

...

Ask discard

```
┌────────┐        ┌────────┐
│ Client │        │ Server │
└────────┘        └────────┘
    │                  │
┌───────────────────────────────┐
│ loop  [2 leaders discarded]│   │
│   │   Discard ID     │      │
│   │─────────────────>│      │
│ ┌─────────────────────────┐ │
│ │ alt  [discard ko]    │   │ │
│ │  │                │   │   │ │
│ │  │ Discard error  │   │   │ │
│ │  │<─ ─ ─ ─ ─ ─ ─ ─│   │   │ │
│ │ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─│ │
│ │     [discard ok]     │   │ │
│ │  │  Discard ok    │   │   │ │
│ │  │<─ ─ ─ ─ ─ ─ ─ ─│   │   │ │
│ │  │  update_view   │   │   │ │
│ │  │<─ ─ ─ ─ ─ ─ ─ ─│   │   │ │
│ └─────────────────────────┘ │
└───────────────────────────────┘
    │                  │
```

Discard ID (client)
Command.DISCARD_LEADER\n
JSON:{Integer ID}

Discard error(server)
Command.REPLY\n
info: "You selected an invalid id"/"You already
discarded 2 leaders"

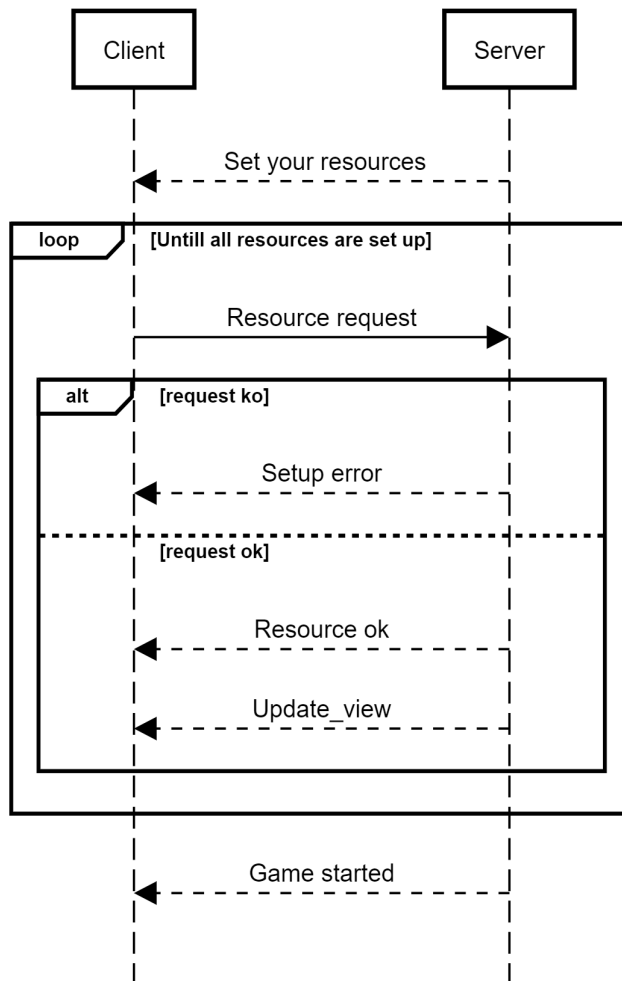**Discard ok(server)**
Command.DISCARD_OK\n
JSON:{Integer ID}
info:"You discarded ID

Update_view
View_Update: the leader card is removed

Resources Setup

Client   **Server**

Set your resources

loop [Untill all resources are set up]

Resource request

alt [request ko]

Setup error

[request ok]

Resource ok

Update_view

Game started

Set your resources
Command.REPLY\n
Info:

info1: "Wait for others to select their resources"
info2: "You can select n resources"

Resource request
Command.SETUP_CONTAINER\n
JSON{ResourceContainer container,
String destination,
Integer destinationID,
Boolean added }

Setup error
Command.REPLY\n
Info:

info1:"You already selected your bonus resources"
info2:"you selected an invalid deposit id"
info3:"you are the first you cannot select extra resources

Resource ok
Command.REPLY\n
info1: "Successfully selected the resources"
info2: "You can select one more resource"

Update_view
Updates all views resource has been added to a player

Game started
Command.REPLY
info: "The game has started"

Turn phase
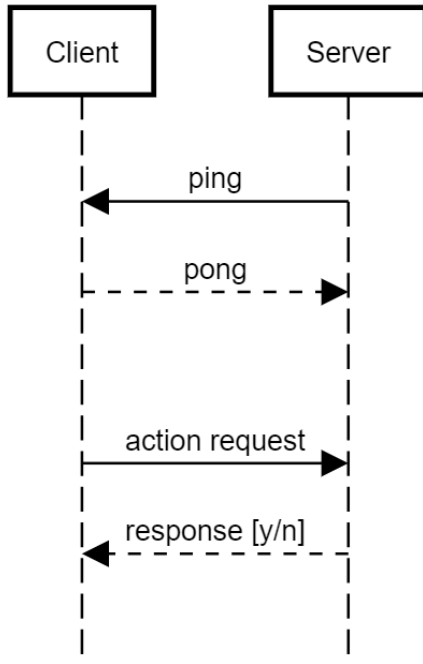
Actions:
  (1) BuyDevelopmentCard
  (2) Produce
  (3) SelectResourcesFromMarket
  (4) ActivateLeader
  (5) ManageResources
  (6) EndTurn

action request (client)
ActionRequest(n)\n

response (server)
Command.Reply\n
response: "You already used your main action"/"Ok"

The next part of the conversation depends on the chosen action.

Buy Development Card
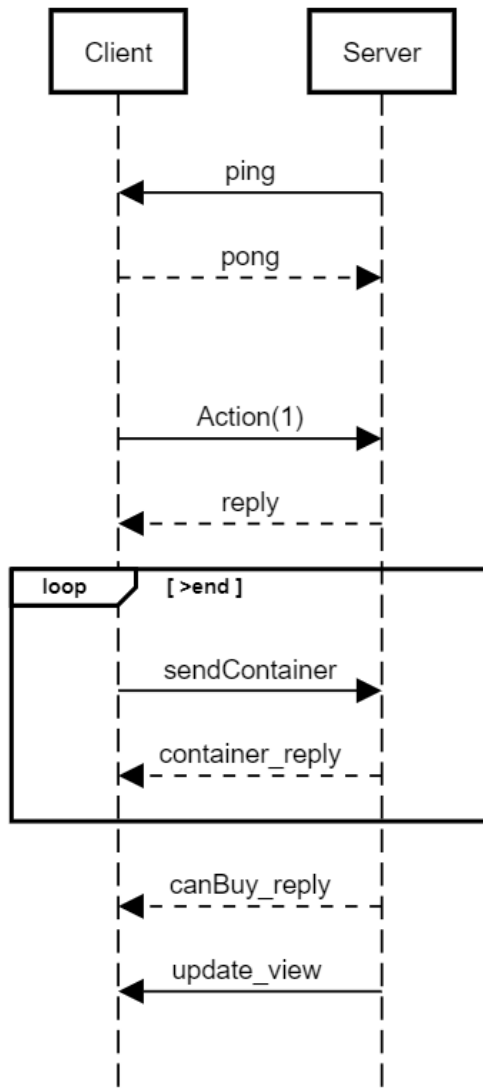
(1) Action (client)
  BuyDevelopmentCard\n
  JSON:{ int id,
           int productionSlotId }

reply (server)
Command.Reply\n
info: Errors/"ok"
Error 1: Invalid ID number
Error 2: The Deck is empty
Error 3: You don't have enough resources
Error 4: You can't insert that card into the selected Slot

sendContainer(client)
Command.SendContainer\n
JSON: {   ResourceContainer resCont,
           String source ("Deposit/Vault"),
           Int sourceID (x/null)
           Boolean added }

container_reply(server)
-On Error
Command.REMOVE_CONT_ERROR\n
info: Errors
Error1: NotEnoughResources
Error2: DifferentResourceType

-On Ok
Command.REMOVE_CONT_OK\n

 canBuy_Reply(server)
-On Error
Command.BUY_ERROR
info: Errors
Error1: NotEnoughResources
Error2: DifferentResourceType

-On Ok
Command.BUY_OK\n
info: "You bought the card!"

update_view(server)
the card appears in the player's board

Produce
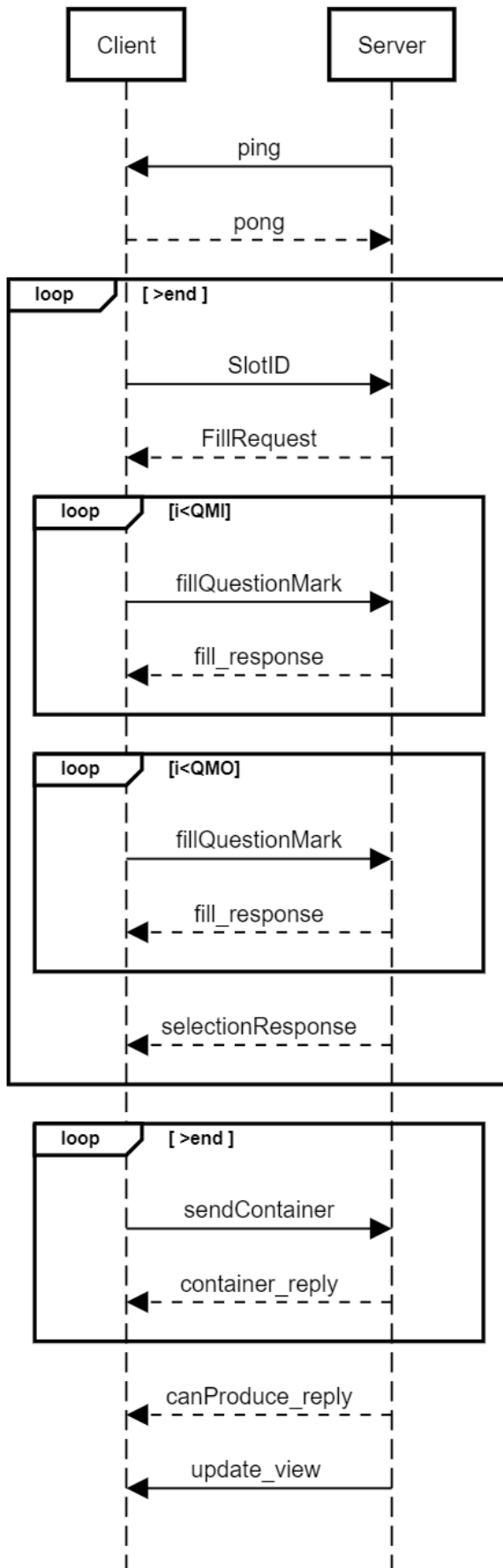
(2) Action

SlotID (client)
Command.PRODUCE\n
JSON: { List<Integer> SlotIDs }

Fill_request (server)
Command.START_FILL\n
JSON: { int QuestionMarkInput,
        int QuestionMarkOutput,
        int ProdSlotID }

fillQuestionMark (client)
Command.FILL_QM\n
JSON: { ResourceType resType}

fill_response (server)
Command.REPLY\n
info: Error/"ok"

Error: The type doesn't exist

selectionResponse (server)
-On Error
Command.PRODUCE_ERROR\n
info: You don't have enough resources

-On Ok
Command.REPLY\n
info: "Now you can select the resource payment"

sendContainer(client) see prev. page
container_reply(server) see prev. page

canProduce_Reply(server)
-On Error
Command.PRODUCE_ERROR

Error 1: You didn't select the right amount of resources
Error 2: You selected too many resources

-On Ok
Command.PRODUCE_OK\n
info: "Prod. executed correctly"

Market Request

(3) Action

Market_request (client):
Command.PICK_FROM_MARKET\n
JSON: { String selection ("ROW/COLUMN")
          int number }

reply (server):
Command.REPLY\n
info: Errors/"ok"/

Error 1: Invalid Row number
Error 2: Invalid Column number

ConversionDecisionRequired (server):
Command.ASK_MULTIPLE_CONVERSION\n
info: "Please select an available conversion"

ConversionChosen (client):
Command.CONVERSION\n
JSON: { ResourceType resourceType }

reply(1)
Command.REPLY\n
info: Error/"ok"
Error: invalid conversion chosen

ChooseDepositSlot (server):
Command.ASK_MARKET_DESTINATION\n

DepositSlotChosen(client):
Command.SEND_DEPOSIT_ID\n
JSON: { int depositID }

reply(2)
Command.REPLY\n
info: Errors/"ok"/
Error1:DifferentResourceType
Error2: ResourceTypeAlreadyStored
Error3: "You must discard this resource"

update_view
All the resources now are visible in the deposit
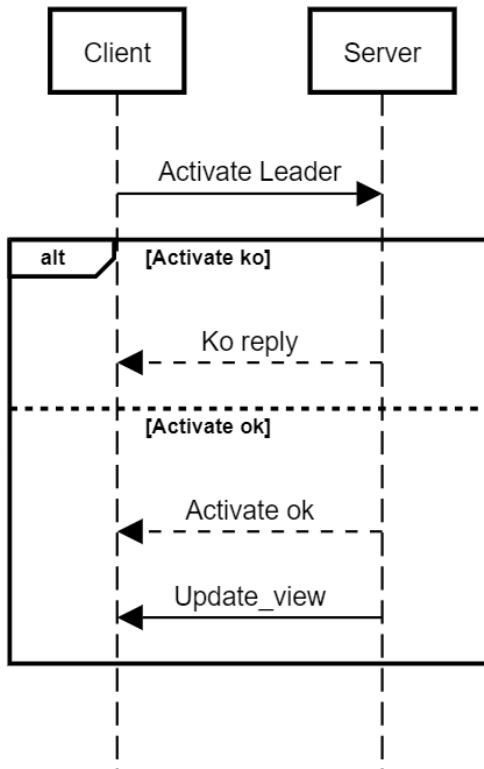
Activate Leader

(4) Action

Activate Leader (client):
Command.ACTIVATE_LEADER\n
JSON:{int leaderCardID}

Ko reply (server):
Command.REPLY\n
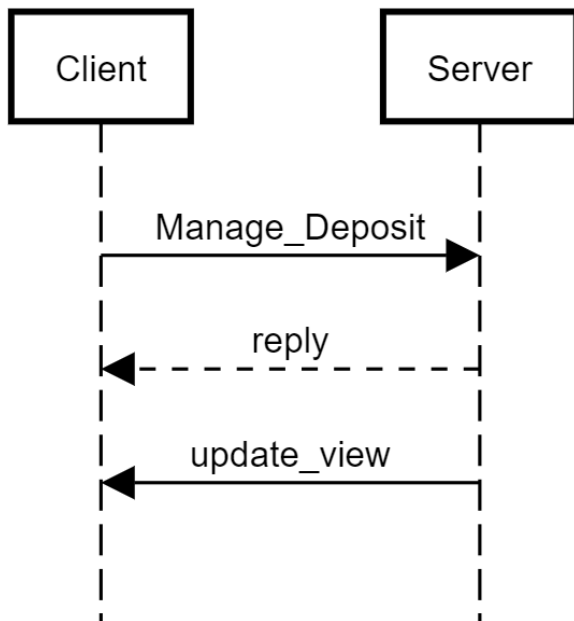info:"You don't meet the requirements to activate this leader"

Activate ok (server):
Command.ACTIVATE_OK\n
JSON: { int leaderCardID }
info:"Correctly activated ID leader"

Update_view (server):
updates all views that the player has activated the ID leader

Manage Deposit

(5) Action

Manage_deposit (client):
Command.SWITCH_DEPOSIT\n
JSON:{ int sourceDepositID,
        int destinationDepositID }
or
Command.MANAGE_DEPOSIT\n
JSON:{ int sourceDepositID,
        int qty,
        int destinationDepositID }
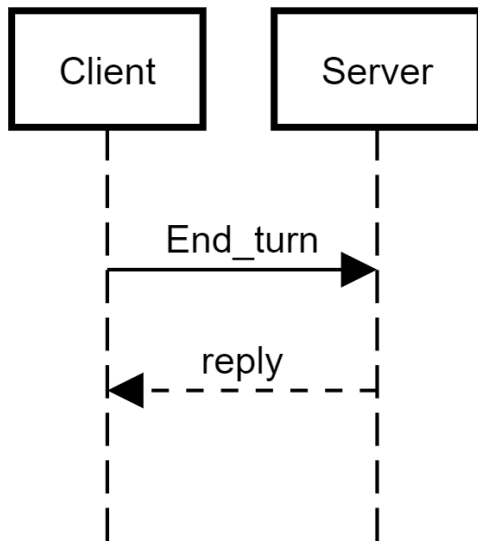
reply (server):
response\n
response: Error/"ok"

Error1: DepositSlotMaxDimExceeded
Error2: DifferentResourceType
Error3: NotEnoughResources
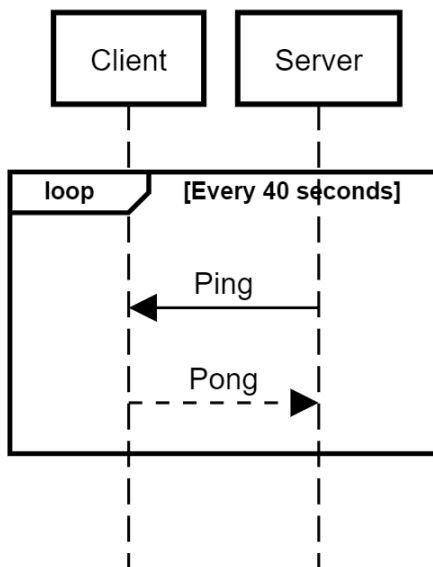Error4: ResourceTypeAlreadyStored
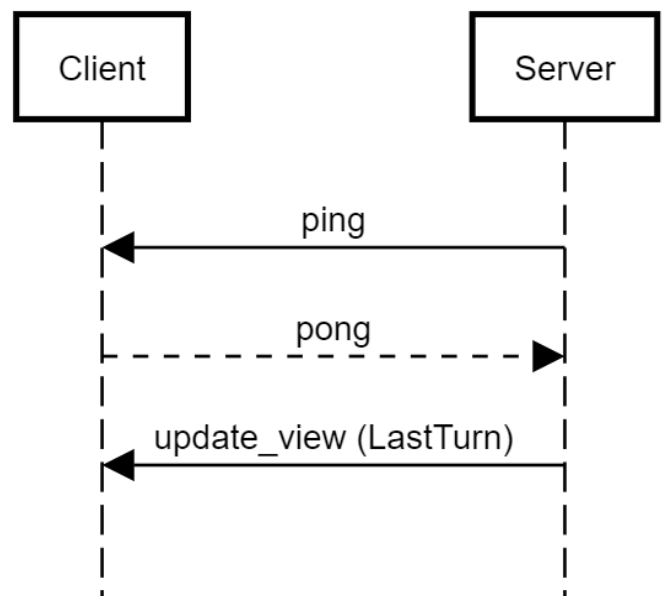
End Turn

(6) Action

End_turn (client):
endturn\n

reply (server):
response\n
response: Error/"ok"
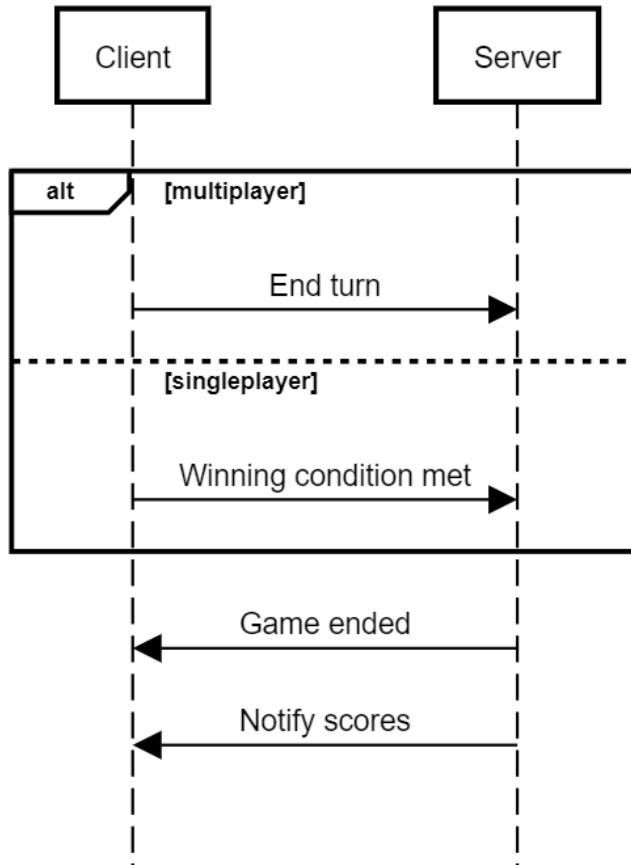
Error: You can't pass your turn

isClientConnected

Last Turn Notify

End Game

Game ended (server)
Command.END_GAME\n

Notify scores (server)
Command.NOTIFY_SCORES\n
JSON: {
    List<String> winners,
    List<String> players,
    List<Integer> points
}
Command.REPLY\n
info: "You are now in the lobby"

update_view
the scoreboard and the winners are
printed, changes the screen to the
lobby