You have been tasked with creating a hardware abstraction layer, also referred to as a driver, to communicate with a power supply. The driver must read measurement values from the power supply and send commands to set the current output, set voltage output, and enable the power supply.

The driver must read from the power supply at various rates without blocking the main process. The driver must use Standard Commands for Programmable Instruments (SCPI) via USB-serial RS-232 adapter to read measurements and send commands. The driver should listen for connections from telemetry clients, create the connection and then broadcast updated measurement values to the clients at a specific interval.

The driver uses a TCP connection to send a JSON payload of values to any connected telemetry clients. The command client uses a TCP connection to send a JSON payload of command values to the driver which are then passed to the power supply as SCPI commands. Consider that these clients exist already and do not need to be designed.

Requirements

- Read from and write to a piece of hardware asynchronously using SCPI commands
    - Read and write functions must be non-blocking for telemetry broadcast
    - The read rate must be capable of being much higher or lower than the telemetry broadcast rate
- Broadcast telemetry to multiple telemetry client connections
    - TCP JSON payloads for the channel-value pairs
    - Telemetry should be broadcasted at 10 Hz
- Receive commands from remote command clients
    - TCP JSON payloads for the channel-value pairs
    - Sets channel values
- Channel names
    - voltage: measured voltage
    - current: measured current
    - enabled: commanded output enabled
    - set_current: commanded current set point
    - set_voltage: commanded voltage set point

The Task

1) Write a high-level abstraction document that could be used to develop the software by a peer. This document should utilize pseudocode, programming language snippets of your choice, a flow diagram, or other ways to express the architecture you have designed.
2) Through UML diagrams or other design tools indicate the program flow, data paths, cyclic events or any other notable design choices.
3) Describe how each requirement listed above is met.