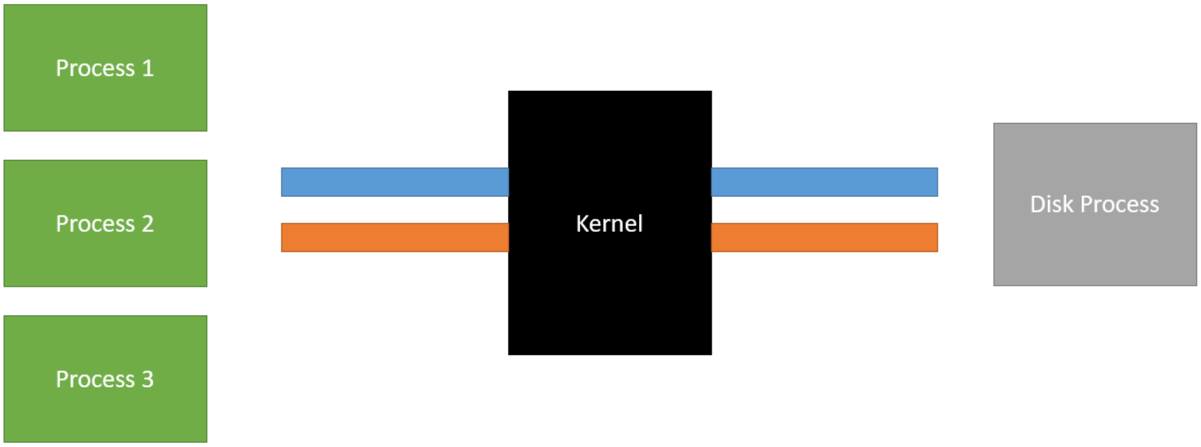
Mini Project 2

*Teams of 4

*all Code MUST be written in C/C++

The following figure describe the system overview



1-Disk Process:

- Has 10 slots.
- Every slot is about char array of size at max 64 char.
- Disk can communicate with the kernel process via 2 streams (Down, and Up)
- Every stream is about Message Queue
- Data from Kernel process to disk process transferred via Down stream.
- Data from Disk to kernel process transferred via the Up stream.

- Disk process when receive SIGUSR2 it increments its CLK variable by 1.
- The data transferred is about char array of size at max 64 char.
- Kernel process can request the disk free slots count. This is done by the following steps:
- A) Kernel Sends SIGUSR1 to disk process.
- B) disk sends a message on the Up stream with a mtype that indicate that is a disk status message.
- C) message contains the number of free slots.
- Disk latency is 3 seconds in ADD.
- Disk latency is 1 second in DELETE.
- When receive the add command. The disk search for a free slots from the 10 slots and write the data in it.
- When receive the delete command. The command contains the id(0 ~ 9) of the slot to be delete.

2-Process:

-every process can read a single file.

-the file format:

Time	**Operation**	**Data**

Time : specify the number of clk cycle that the process will request.

Operation: specify the request.

Data: carries the request data

Example

10	**ADD**	**“Hello”**
13	**ADD**	**“This is me”**
20	**DEL**	**<slot number 0 ~ 9>**

-the process when receive SIGUSR2 it increments the clk cycle variable.

-the delete message format

D	**<slot number>**

One char that contains ‘D’ for delete.

The second char contains the slot number to be deleted

-Add msg format

A	**The msg itself i.e. “Cairo University”**

-the process sends theses msgs to kernel via Up stream.

-kernel responds to these requests via down queue

Kernel responses are:

0 → successful ADD

1 → successful DEL

2 → unable to ADD

3 → unable to DEL

-process when send a request. It MUST wait the response message from the kernel (BLOCKING)

3- Kernel Process

-MUST log every event:

-A) requests from processes.

-B) Itself responses to processes.

-C) disk responses.

-D) requests to disk.

-sends SIGUSR2 every second to all the processes.

-when receive message from process:

-A) identify the type of message (ADD, or DELETE)

-B) check the disk status.

-C) if it is valid to do the operation, then send the appropriate message to the disk process.

-D) if not valid to do the operation, then send appropriate feedback message to the process.

Note:

The kernel is not the Linux Kernel. This project is a small simulation of a system. The kernel is just a process.