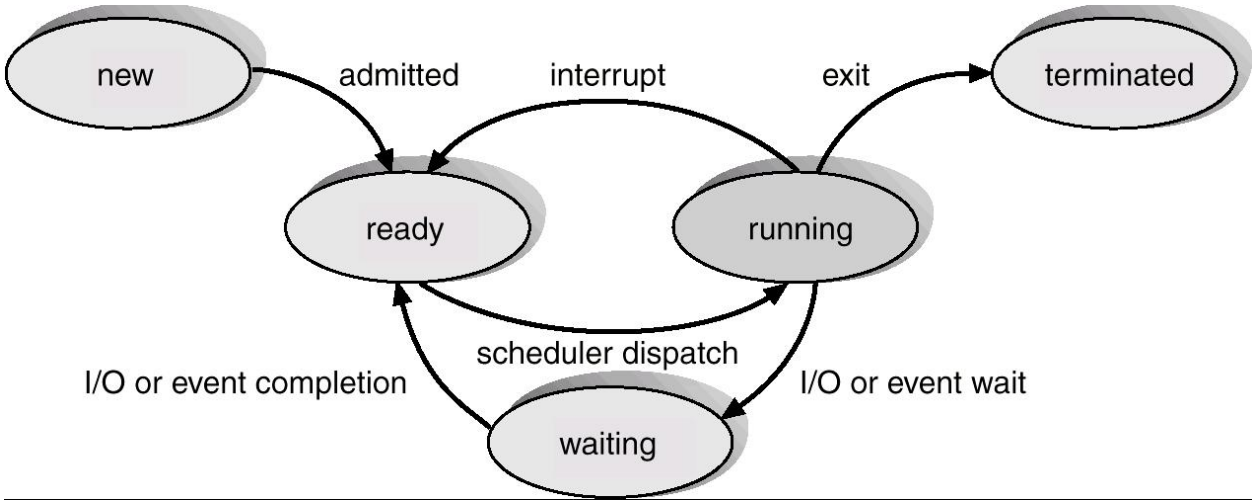
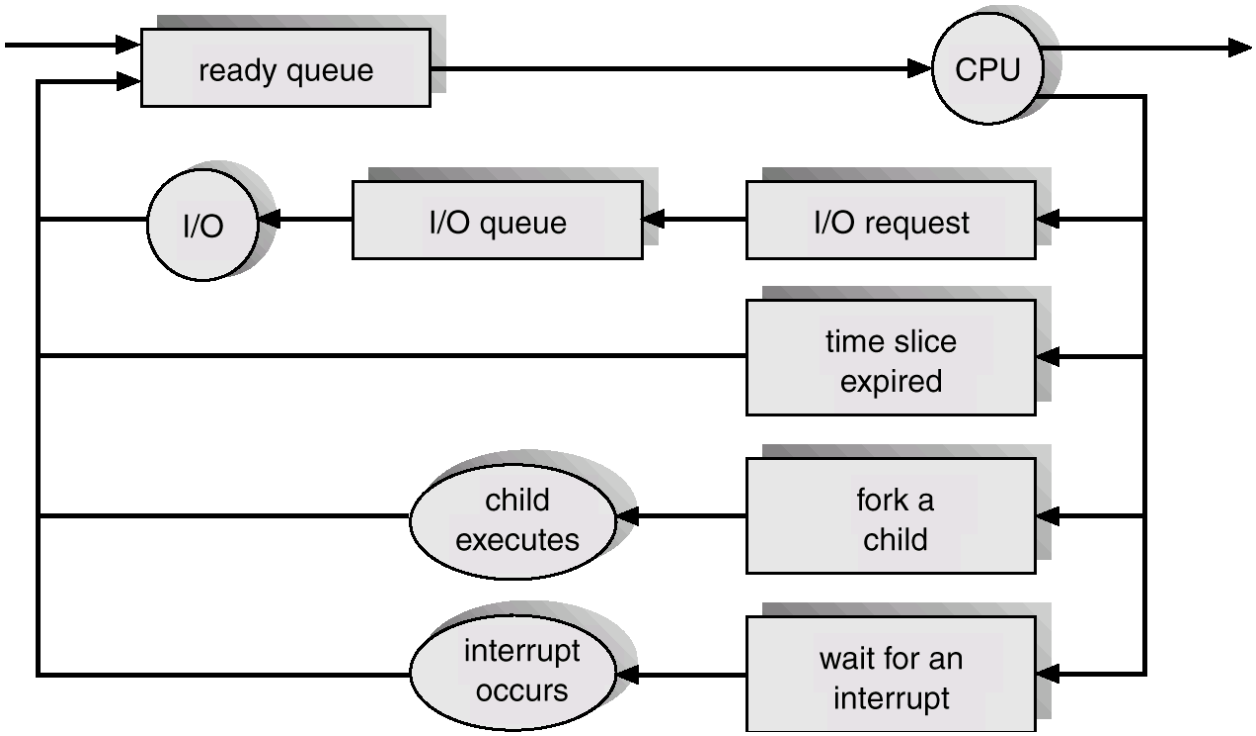
// PROCESS STATE DIAGRAM
==========



- IO or event wait expanded



- **Null ====> New :**
  - a new process is created to execute the program
- New ====> ready:

    - OS will move a process from prepared to ready state when it is prepared to take additional processes.
- Ready ====> Running:
    - when it is time to select a new process to run, the OS selects one of the processes in the ready state.
- Running ====> terminated:
    - The currently running process is terminated by the OS if the process indicates that it has completed, or if it aborts.
- Running ====> Ready:
    - The process has reached the maximum allowable time or interrupt.
- Running ====> Waiting:
    - A process is put in the waiting state, if it requests something for which it must wait.
- Waiting ====> Ready:
    - A process in the waiting state is moved to the ready state, when the event for which it has been waiting occurs.
- Ready ====> Terminated:
    - If a parent terminates, child process should be terminated
- Waiting ====> Terminated:
    - If a parent terminates, child process should be terminated

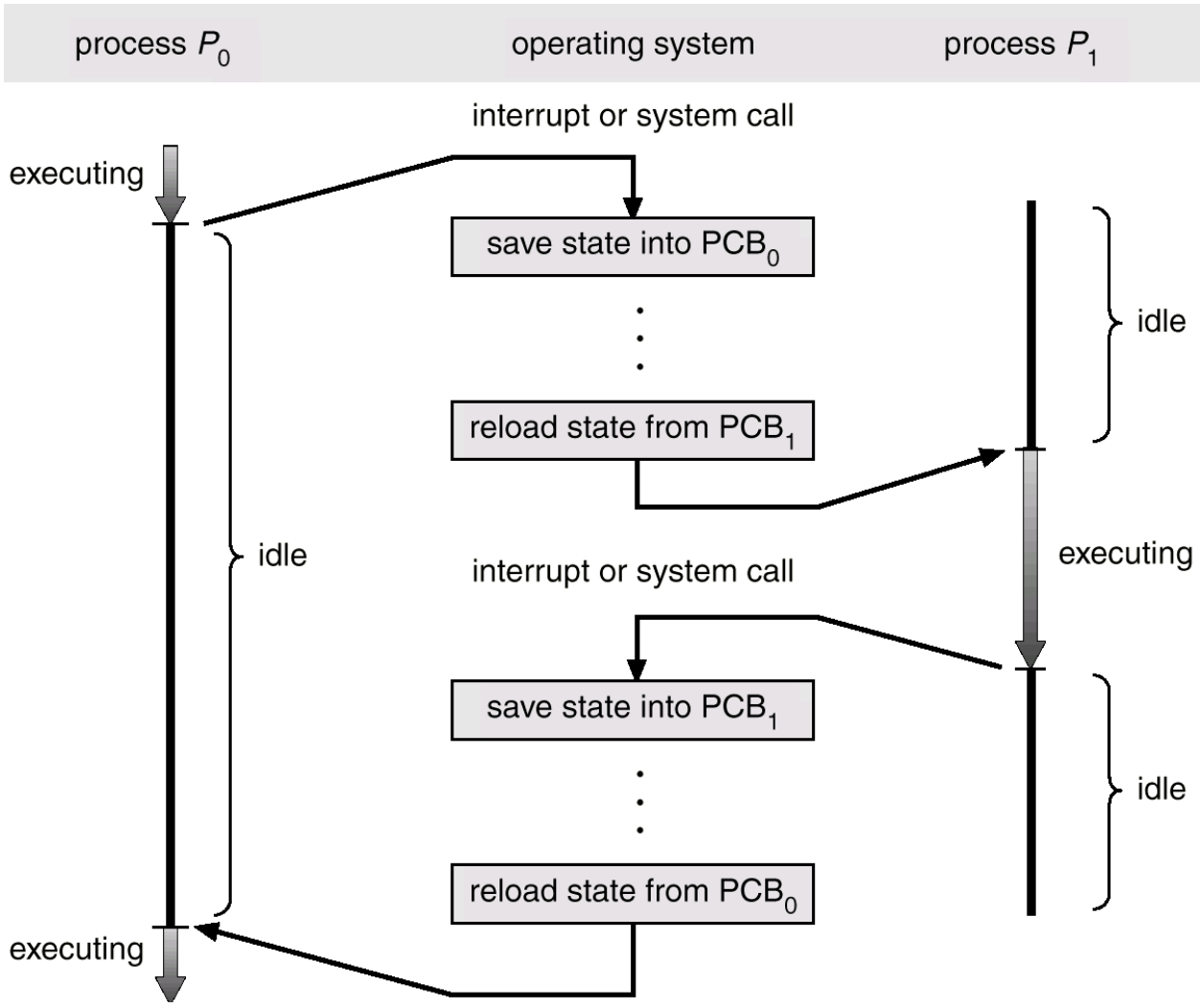
NOTES

==========

- Timesharing System: Switch the CPU frequently that users can interact can interact with the program while it is running.
- Long-term scheduler (or job scheduler) – selects which processes should be brought into the ready queue.
- Short-term scheduler (or CPU scheduler) – selects which process should be executed next and allocates CPU.
- Most processes are either I/O bound or CPU bound.
    - I/O bound process spends more time doing I/O than it spends doing computation.
    - CPU bound process spends most of the time doing computation.
- Some OSs introduced a medium-term scheduler using swapping.
    - Key idea: it can be advantageous, to remove the processes from the memory and reduce the multiprogramming.
    - Swapping: removal of process from main memory to disk to improve the performance. At some later time, the process can be reintroduced into main memory and its execution can be continued when it left off.

// CONTEXT SWITCH

==========



PROCESS CREATION

==========

- A system call is used to create process.
  - Assigns unique id and space
  - PCB is initialized.
  - The creating process is called parent process.

PROCESS TERMINATION
==========

- Process executes last statement and asks the operating system to decide it (exit).
- Output data from child to parent (via wait).
- Process' resources are deallocated by operating system.
- Parent may terminate the execution of children processes (abort).
    - Child has exceeded allocated resources.
    - Task assigned to child is no longer required.
    - Parent is exiting.
- Operating system does not allow child to continue if its parent terminates.
- Cascading termination may happen like: grandparent, parent, child.

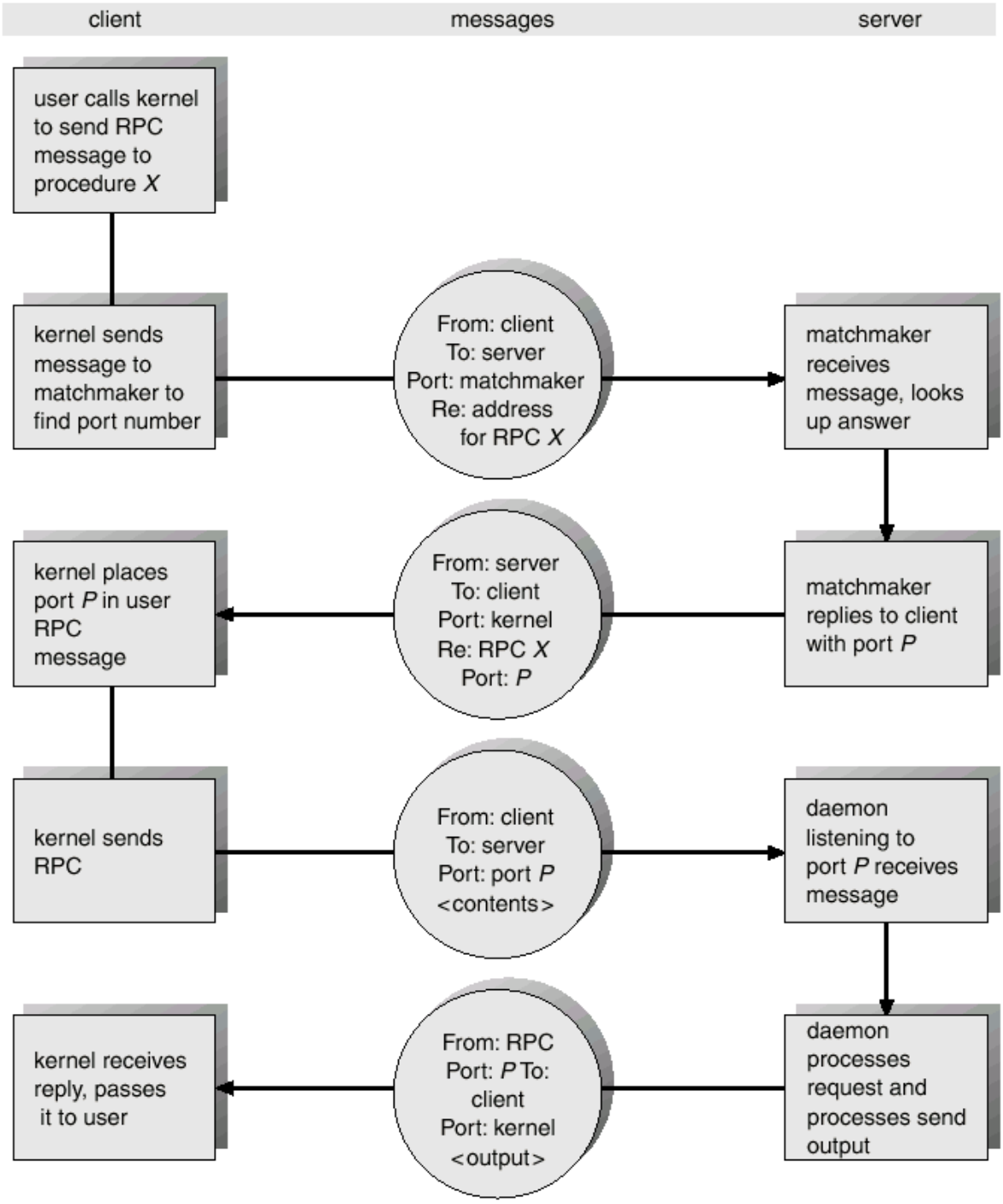
NOTE
==========

- Independent process cannot affect or be affected by the execution of another process.
- Cooperating process can affect or be affected by the execution of another process
- Interprocess communication
    - Shared memory
    - Message passing
        - Fixed sized message
        - Variable sized message

- Methods to implement a link for communication
    - Direct communication
        - Processes must name each other explicitly:
    - Indirect Communication
        - The messages are sent and received from mailboxes
    - Synchronous or asynchronous communication
        - Message passing may be either blocking or non-blocking.
        - Blocking is considered synchronous
            - Sender gets blocked(waits for response)
        - Non-blocking is considered asynchronous
            - Sender does not get blocked(does not wait for response)
    - Automatic or explicit buffering
        - Buffer in between
        - Receiver executes only after whole msg is recieved

Socket: Concatenation of IP address and port

RPC
==========

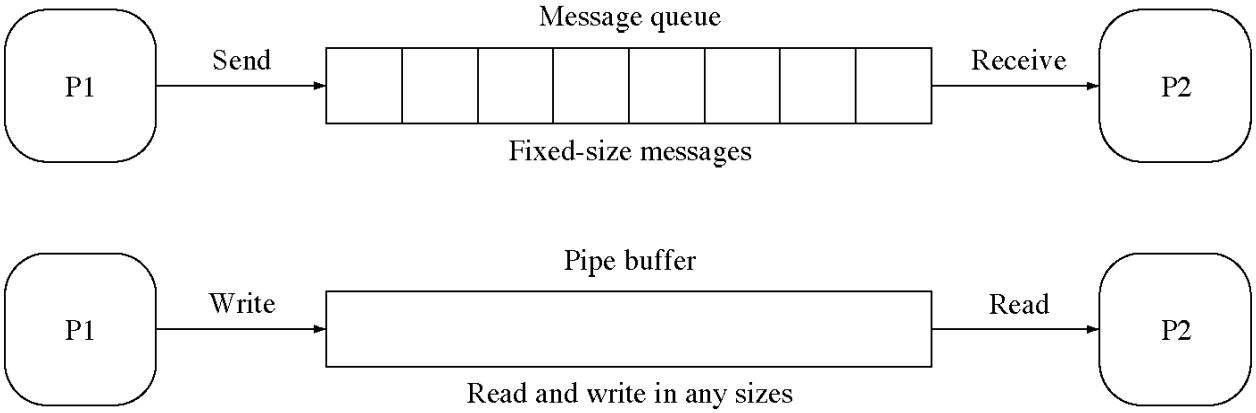


Message Deliver Semantics
==========

- Atmost Once delivery

    - ideal for applications that need high throughput and low latency due to the fire-and-forget nature
- Atleast oce Delivery
    - ideal for applications where receiving every message is more important than having high throughput and low latency
- Exactly Once Delivery
    - highest implementation overhead and highest cost as well as potentially the worst performance of all of the delivery semantics

PIPES
==========



- Connects an open file of one process to an open file of another process
- Often used to connect the standard output of one process to the standard input of another process