Chapter 3 : Process

1. The Process :

* Program in computer is the code of developer, it is the instruction of software.
* The process is a task in program. 1 program may do multiple tasks and each task is process. Process is the program in execution, working inside the CPU.
* The Memory layout of process includes : text section (the executable code, fixed size), data section (global variables, fixed size), heap section (dynamic allocating resources, not fixed size) and stack section(temporary variables such as function parameter, local variable, return address)
* Process State: New, ready, running, terminated, waiting,
  + New - > ready : admitted the new process into queue
  + Ready -> running: executed process
  + Running -> terminated : end of process
  + Running -> waiting : wait for I/O or event to be done
  + Waiting -> Ready: I/O or event completion
  + Running -> Ready : interrupt the process, it depend on scheduler algorithm (like : roundrobin,…)
* PCB : Process represent in Operating System by PCB, PCB includes :
  + Process state
  + Program Counter: indicates the address of next process will be executed after this process
  + CPU register:
  + CPU scheduling information : includes process priority
  + Memory management information : amount of CPU the process can use
  + I/O status information : the list of I/O devices is used in process
* Thread : One processor may have multiple thread, One single thread is control one flow of instruction. To simplify that, the thread is the small process. The different between process and thread is thread using shared memory in process.

1. Process Scheduling :

* Scheduling Queues :
  + Ready queue : whenever the process enter the system, it is placed in ready queue
  + Waiting queue: when the process is executed but terminate, interrupt, waits in middle will be placed in waiting queue.
* CPU Scheduler : The role of the CPU scheduler is to select from among the processes that are in the ready queue and allocate a CPU core to one of them.
* Context Switching : When the process is interrupted, the system save the current context so that it can resume in later.

1. Operation of process :

* Process Creation :
  + A process may create several new processes. The creating process is called a parent process, the new ones are called the children of that process.
  + A child process may be able to take its resources directly from operating system, or it may be constrained to a subset of the parent resource
* Process Termination :
  + A process is terminated when it finish execution and ask the operating system delete. All the resource using for the terminated process will be give back to OS.
  + Some OS don’t allow a child exists if their parent are terminated (cascading termination)
  + A process has terminated, but the parent has not called wait(), is known zombie process.
  + The child is orphan if their parent don’t call wait() and terminated.
* Android Process Hierachy:
  + Foreground Process : visible on screen and user can interact with it
  + Visible Process : performing activity that foreground referring to
  + Service Process : background process but performing the activity and apparent to the user
  + Background Process : performing the activity but not apparent
  + Empty Process : hold no active component.

1. Interprocess Communication :

* Cooperating process : Information sharing, computation speedup and modularity.
* We need to synchronization.
* There are two models of interprocess communication :
  + Shared memory : region of share memory is established, process can exchange information by reading and writing data in the region.
  + Message- passing : send messages exchanged between the cooperating process

1. IPC in shared memory :

* The process share memory to read and write data, but they are not writing in the same location at same time.
* Shared memory can solve the producer-consumer problem : whenever the process consume, it will be sure that producer have produced in this region of memory.
* Type of Buffer:
  + Unbounded Buffer : Infinitive size , producer don’t need to wait just put to buffer, consumer have to wait when buffer is empty.
  + Bounded Buffer: Fixed size, producer need to wait if the buffer is full, and consumer wait when buffer is empty

1. IPC in Message-passing :

* Two process can communicate through passing message.
* Direct communication has the disadvantage is limited modularity with indirect communication message is sent to mailbox. Mailbox may be owned by process or operating system.
* Synchronization:
  + Producer (Blocking send) : Do nothing until send the message to mailbox
  + Consumer(Blocking receive) : Do nothing until the message available in mailbox.

1. IPC in example :

* POSIX Shared memory : using memory – mapped file