Thread vs process

Both processes and threads are independent sequences of execution.The typical difference is that threads (of the same process) run in a shared memory space, while processes run in separate memory spaces.

Process:

- *An executing instance of a program is called a process.*
- *Some operating systems use the term 'task' to refer to a program that is being executed.*
- *A process is always stored in the main memory also termed as the primary memory or random access memory.*
- *Therefore, a process is termed as an active entity. It disappears if the machine is rebooted.*
- *Several process may be associated with a same program.*
- *On a multiprocessor system, multiple processes can be executed in parallel.*
- *On a uni-processor system, though true parallelism is not achieved, a process scheduling algorithm is applied and the processor is scheduled to execute each process one at a time yielding an illusion of concurrency.*

Thread:

➢ *A thread is a subset of the process.*

➢ *It is termed as a 'lightweight process', since it is similar to a real process but executes within the context of a process and shares the same resources allotted to the process by the kernel.*

➢ *Usually, a process has only one thread of control – one set of machine instructions executing at a time.*

➢ *A process may also be made up of multiple threads of execution that execute instructions concurrently.*

➢ *Multiple threads of control can exploit the true parallelism possible on multiprocessor systems.*

➢ *On a uni-processor system, a thread scheduling algorithm is applied and the processor is scheduled to run each thread one at a time.*

➢ *All the threads running within a process share the same address space, file descriptors, stack and other process related attributes.*

➢ *Since the threads of a process share the same memory, synchronizing the access to the shared data withing the process gains unprecedented importance.*