Distributed Operating System

Processor Management

- The Processor Manager provides the policies and mechanisms to create, delete, abort, name, rename, find, schedule, block, run, and synchronise processes.
- If required, it provides real-time priority execution.
- The Processor Manager also manages the states of execution—READY, RUNNING, and WAIT—as per a standalone system.
- Each CPU in the network has its own run-time kernel that manages the hardware locally.
    - local statuses are aggregated into global tables
- The kernel controls and operates the CPU and manages the queues used for states of execution.
- Upper-level system policies direct how process control blocks (PCBs) are stored in queues and how they're selected to be run.
- Local kernel states are dependent on the global system's process scheduler and dispatcher.
    - global scheduler organises the queues within local CPUs, and chooses the running policy that's used to execute the processes on those queues.
- The system's scheduling function has three parts:
    1. **Decision mode**: determines which policies are used when scheduling a resource
        - options include preemptive, nonpreemptive, Round Robin, etc.
    2. **Priority function**: gives the scheduling algorithm the policy that's used to assign an order to processes in the execution cycle.
        - priority is determined using a calculation based on system characteristics
            - occurrence of events, task recurrence, system loading levels, or program run time characteristics
            - run-time characteristics include most time remaining, least time remaining, etc.
    3. **Arbitration rule**: a policy that's used to resolve conflicts between jobs of equal priority.
        - dictates the order in which jobs of the same priority are to be executed
            - rules include last-in first-out (LIFO) and first-in first-out (FIFO)
- Job scheduling relies on one of three theories:
    1. queuing theory
    2. statistical decision theory
    3. estimation theory
        - scheduler based on process priorities and durations
        - maximises system throughput by using durations to compute and schedule the optimal way to interleave process chunks
- To create a process, the process manager creates a Process Control Block.
    - PCB contains information identifying the process's location in the network
- To locate a process, the process manager uses a system directory that searches all kernel queue spaces.
    - requires interprocess communication support
- To synchronise processes, the process manager uses message passing or remote procedure calls.
- To delete a process, the process manager finds the correct PCB and deletes it.

Memory Management

- The global Memory Manager uses local kernels with a paging algorithm to track the amount of memory that's available.
    - algorithm is based on local system goals, but policies and mechanisms used locally are driven by global system requirements
- Memory allocation and deallocation depend on scheduling and resource-sharing schemes that optimise global resources.

- The global Memory Manager works the same way as it does for a standalone OS, but is extended to accept requests for memory from both local and global sources.
    - locally, the Memory Manager allocates pages based on the local policy
    - globally, it receives requests from the Process Manager to provide memory to new or expanding client or server processes
- The Memory Manager also uses local resources to perform
    - garbage collection in memory
    - compaction
    - decide which are the most and least active processes
    - determine which processes to preempt to provide space for others
- The Memory Manager handles requests from the Process Manager to allocate and deallocate space based on the network's usage patterns.
- Combined memory for the entire network is made up of several sub-pools (one for each processor).
    - the Memory Manager has a subcomponent that exists on each processor
- When a page fault occurs, the Memory Manager automatically brings that page into memory.
- When a page is modified while in memory, the Memory Manager writes the changed page back to disk when swapping the page out of memory.
- Before allocating space, the Memory Manager examines the total free memory table:
    - if the request can be filled, the memory is allocated and the table is modified to show the location of the allocated space
- Typical protection checks are performed on pages as they're loaded into memory (e.g., read, write, execute)

Network Operating System

- The Network Operating System is simply a standalone system with a network stack and networking capabilities.
- Process and memory management are handled as per a standalone system by the local process and memory managers.
- Networked nodes must connect and authenticate with other sites to submit requests to the remote host to schedule processes.
- Unlike distributed operating systems, the external systems, resources, and mechanisms involved in data transfer and remote resource management are exposed to the user.