Processes and Process Control

1. Processes and Process Control
2. Definitions of a Process
3. Systems state vs. Process State
4. A 2 State Process Model
5. Process Creation, what does it mean?
6. Process Termination's meaning
7. A Five State Model of Process Control
8. Queuing Models of Systems having I/O
9. Swapping (Suspending) Processes
10. CPU with I/O and Suspended State

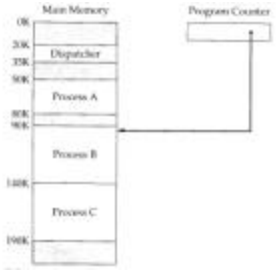
11. Operations on Processes
12. O/S Support for Processes
13. O/S Global Process Structures
14. Memory Tables
15. I/O Tables
16. File Tables
17. Process Tables
18. Process Image
19. Process Control Blocks
20. Process Memory Management
21. O/S Use of PCBs
22. O/S Kernel Services

23. Process Switching
24. Running the O/S
25. Processing Threads
26. Thread Management in Windows NT
27. Examples - MVS
28. MVS List Structures
29. Examples - Unix
30. Examples - Windows NT

Definitions of A Process

1. A program in execution
2. An asynchronous activity
3. the ``animated spirit'' of a procedure
4. the ``locus of control'' of a program in execution
5. that which is manifest by the existence of a process control block in the O/S
6. that entity which is assigned to processors
7. the ``dispatchable'' unit

Systems state vs. Process State



1. Some processes are resident in memory.

2. The kernel is always resident.

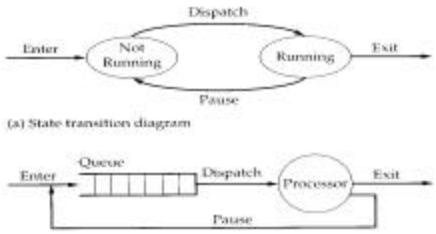
3. One process runs at a time.

A 2 State Process Model

CPU is the only resource for many jobs.

1. Enter --- Process Creation
2. Dispatch --- Scheduling/Queuing Discipline
3. Pause --- Give other jobs a chance
4. Exit --- Process Termination

Figure 2. State Diagram of CPU only System



Process Creation, what does it mean?

Some reasons for process creation include:

1. A new batch job
2. Interactive O/S login
3. O/S created
4. Spawned by an existing process

The frequency of process creation reflects the expense of creation.

Process Termination's meaning

Some reasons for process termination include:

1. Normal Completion

2. Excessive Resource Use
   a) CPU Time Out
   b) Insufficient Memory
   c) File system full error

Process Termination's meaning
(continued)

3. Security Violations/Programmer Errors
    a) Illegal Address
    b) Illegal Instruction
    c) Privileged Instruction
    d) Data Misuse (type error/ initialization error)
4. Systems Control
    a) Parent Job Terminated
    b) Terminated by Parent
    c) Operator or O/S intervention

Process Termination's meaning
(continued)

The frequency of process creation reflects the expense of creation.

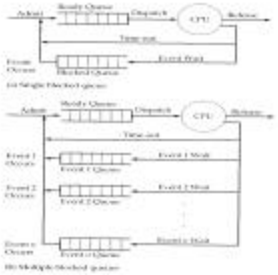
A Five State Model of Process Control

Blocking - When a process waits on a non-CPU service (typically I/O).

Real systems have I/O, so a more realistic model is:

Figure 3

Queuing Models of Systems having I/O



An architecture with I/O and CPU is:

Figure 4

Swapping (Suspending) Processes

Suspending a Process - When the O/S saves the state of a non-running program from main memory to auxiliary memory.

Activation - When the O/S reloads a suspended process into main memory from auxiliary memory.

Swapping (Suspending) Processes **(continued)**

Some reasons for swapping out processes from memory

1. System malfunction - Save state and resume after fix
2. User suspicious about partial results- Debugging/ check-pointing
3. Correct short term load (or memory requirement) fluctuations
4. Fairness (one big process prevents others from running)

CPU with I/O and Suspended State

We can either treat a suspension as independent from blocking, or as mutually exclusive.

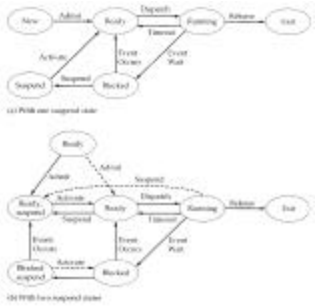


Figure 5: O/S supporting Blocking and Suspension

Operations on Processes

Some operations on processes:

1. Create a process
2. Destroy (terminate) a process
3. Block a process
4. Suspend a process
5. Resume (activate) a process
6. Change a process's priority
7. Wake up a process
8. Wake up a process (put it into the ready state)
9. Enable a process to communicate with another process (interprocess communication)

O/S support for Processes

The O/S in its role as resource manager and as run time interface controls resource access by mapping processes to resources.

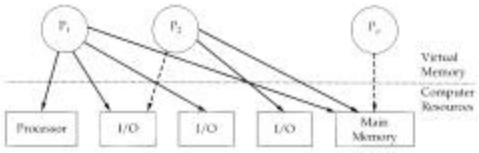


Figure 6: Process and Resources

O/S Global Process Structures

The big picture for process management looks like:

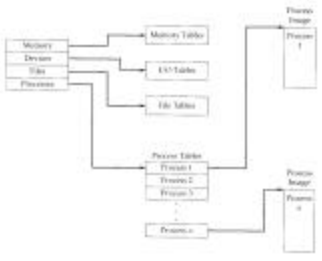


Figure 7: Processes and Resources

Memory Tables

Memory tables record the following information:

1. The allocation of main and auxiliary memory to processes
2. Memory protection (O/S vs. users, users from each other, read only vs. write instructions vs. data)
3. Control information for the virtual memory manager

I/O Tables:

Manage hardware control (and perhaps higher level control) of channels and peripherals in the system.

File Tables:

Provide security, access control, and naming support for persistent objects.

Process Tables:

Manages each individual process's data structures, stores security permissions, and process state info.

Process Image

Process Image --- The state information (attributes) of the process (data/stack/instructions, I/O state). Process images typically contain:

1. User Data
2. User Program (Instructions)
3. System Stack
4. Process Control Block

Process Control Blocks

Process Control Blocks (PCBs)--- The data structures the O/S allocates for managing each process, containing:

1. Process identifiers --- process id, parent id, user id.
2. Process State Info --- User Visible Registers, Control Registers (and PC), and stack pointer.

Process Control Blocks (continued)

3. Process Control Info --- Scheduling and State info, System Data Structures, Inter-process Communications, Process privileges, Memory Management, Resource Ownership and Utilization.

Process Memory Management

Each process image is allocated its own virtual memory.

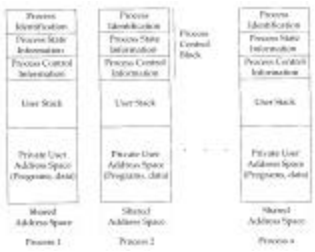
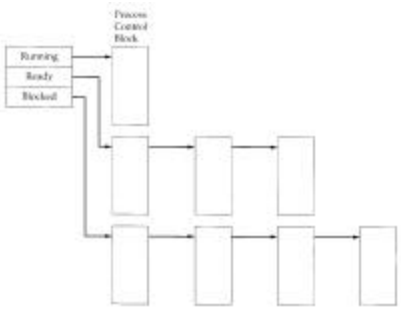


Figure 8: Process Memory Management

O/S Use of PCBs

The system accesses processes via their PCBs for state transitions and scheduling as per state diagram and queuing models.

Process Lists --- Correspond to queues and service structures in queuing model.



O/S Kernel Services

An O/S kernel provides privileged access to system resources, running in systems mode, control mode, or kernel mode.

Typical Kernel services include:

1. Process Management --- Process Creation, Process Termination, Process Switching Process Synchronization, Inter-process Communication.

O/S Kernel Services (continued)

2. Memory Management --- Allocation of address space to processes, Swapping, virtual memory management.
3. I/O Management --- Buffer management, device and channel allocation to processes
4. Systems Support --- Interrupt handling, Accounting, Monitoring.

Process Switching

Process switching is the O/S transferring control from one process to another. Issues include:

1. When to switch? Preemption vs. Non Preemption

Process Switching (continued)

2. Context Switching --- done as follows:
   a) Preserve the running process's state in the PCB, if swapping save image.
   b) Load process image's PCB into system registers and memory restore the program counter.

Running the O/S

The following are typical of O/S run time support structures:

1. Non-process O/S --- A more primitive structure (MS-DOS, CP/M)
2. Single separate Kernel Process --- A more monolithic approach, has the efficiency advantage of fewer context switches, but less flexible (macrokernel?).
   (Unix/Linux, Mac OS, VM, MVS).

Running the O/S **(continued)**

3. System services via Kernel and User processes --- A more flexible approach (microkernel?). (Windows NT, OS/2, MACH, GNU HURD, Amiga DOS).

Processes and Threads

Some people consider threads as a special form of process.

1. Processes control a unit of resource ownership.
2. A process is typically the unit of dispatching.
3. Threads share process context,
4. Threads are asynchronous,
5. Threads have less context than processes.

Processes and Threads

6. Threads can be created/terminated at a lower cost.
7. Threads cooperate to do a process in parallel with (relatively) fine granularity of parallelism

Threads are suited to shared memory SMP machines.

Thread Management in Windows NT

Threads typically cooperate to do the same work as a traditional process. Often the system services are done in user space in a micro-kernel system (to make them run time configurable/flexible with an efficiency penalty).

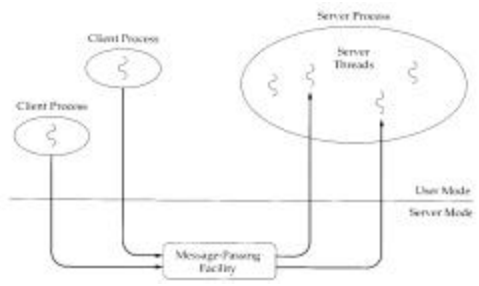


Figure 10: Cooperation in a Typical Thread System-NT

Examples --- MVS

MVS has 3 task (process) states: Ready, Active, Waiting. Entire task may be swapped to auxiliary storage.

Consider a task composed of:

1. a main program,
2. a customer inquiry module,
3. an order entry module and
4. a production tracking module

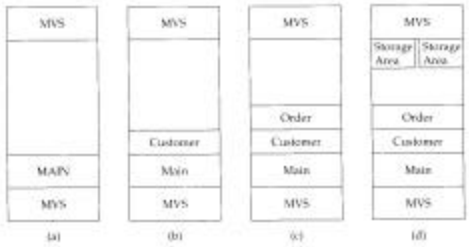


Figure 11: MVS Address Space Example

MVS List Structures

MVS tracks system resources used and tasks using list structures.

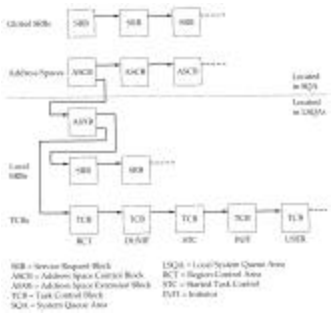


Figure 12: MVS List Structures Example

Examples --- Unix

Unix has a macro-kernel, and processes (some versions now have thread support).

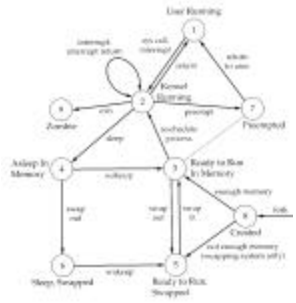


Figure 13: Unix Process States

Examples - Windows NT

Windows NT is multithreaded, and allocates handles for managing processes and resources.

The access token identifies the user, and their security permissions.

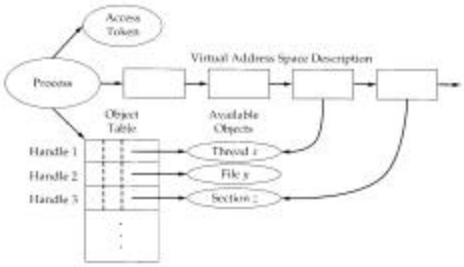


Figure 14: An NT Process and Its Resources