31268 Web Systems

Faculty of Information Technology

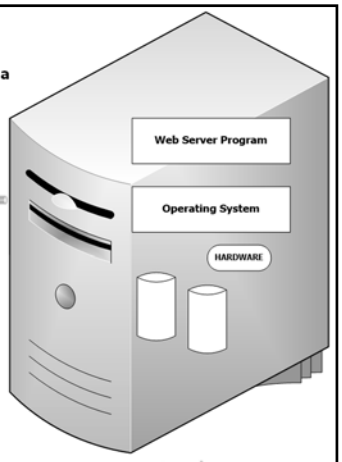
Week 6: Operating Systems 3
Part 1 - Process Management

1

Recap: The web...

A bunch of computers and a network of networks...

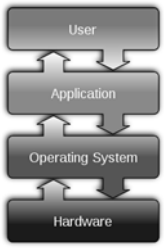
... and a whacking **big** computer running the web site program on an operating system running on HARDWARE



www.uts.edu.au 2

Recap: Operating System

- Manages your computer
- Runs programs
- Interface between user and hardware
- Provides services to programs & users
- Protects users and programs from each other….

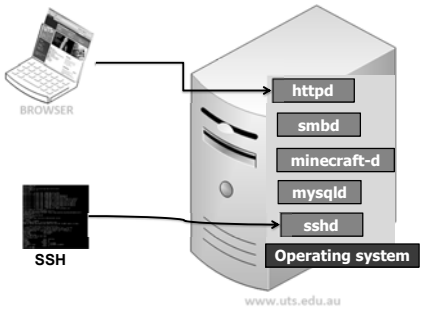


wikipedia

3

Web Example

Our Web Servers run various programs to provide services …..

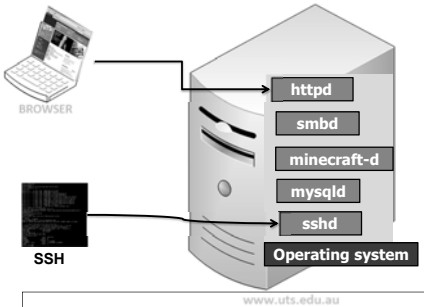


www.uts.edu.au

4

Web Example

Our Web Servers run various programs to provide services …..



www.uts.edu.au

- **Today we will look at how O/S runs programs**

5

Operating Systems - Outline

- *Operating Systems*
- *file systems and file manipulation*
  - *Disk physical structure*
  - *Disk logical structures*
  - *File allocation methods*
  - *Unix: files, directories and redirection*
- *Scripting and regular expressions*
  - *Unix and the Command-Line interface*
  - *Variables and bash scripts*
- **processes, threads, piping and redirection**
  - Programs vs processes, IPC, deadlocks
- memory and process management
  - Physical, Logical & Virtual memory

6

Programs vs Processes

Faculty of Information Technology

- Processes are programs in execution.
  (also called "tasks")
- Every time a program is **run**, a new process is created.
- In addition there are "system processes", "services" or "background tasks"
  - Eg mouse, antivirus, …

7

Process Management

- The role of the computer is to run our programs
- Process management is how the OS handles this step
  - starting processes
  - managing running processes
  - performing interprocess communication
  - terminating processes

8

Managing processes

Faculty of Information Technology

- Modern operating systems can **appear** to run multiple programs concurrently
- Run one program at a time
  - single tasking – *e.g. traditional embedded systems*
  - batch processing – *e.g. ancient mainframe*

9

Managing processes

Faculty of Information Technology

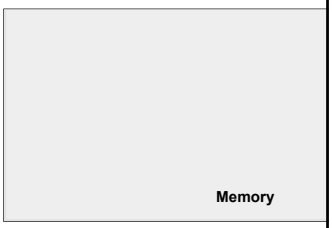
- Modern operating systems can **appear** to run multiple programs concurrently
- Run one program at a time
  - single tasking – *e.g. traditional embedded systems*
  - batch processing – *e.g. ancient mainframe*
- Concurrent processing
  - Multiprogramming – *e.g. windows 3*
  - Multitasking – *all modern operating systems*
  - Multithreading – *most modern operating systems pentium 4+*
  - multiprocessing – *multicore CPUs, dual/quad CPU*

10

Process States

Faculty of Information Technology

Processes can be in various "states"



wikipedia

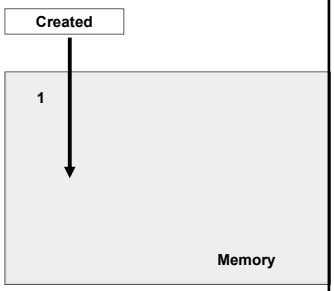
11

Process States

Faculty of Information Technology

Processes can be in various "states"

1. **Created** (ie: loaded into memory)



wikipedia

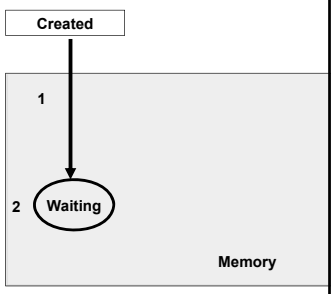
12

Process States

Faculty of Information Technology

Processes can be in various "states"

1. **Created** (ie: loaded into memory)
2. **Waiting** to be run by CPU



wikipedia

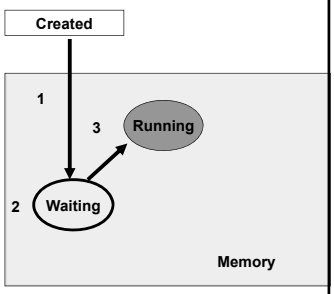
13

Process States

Faculty of Information Technology

Processes can be in various "states"

1. **Created** (ie: loaded into memory)
2. **Waiting** to be run by CPU
3. O/S **runs** the process



wikipedia

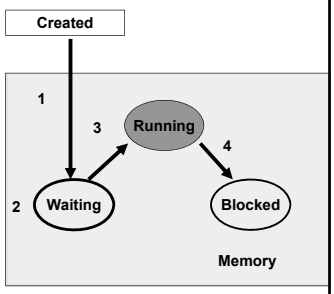
14

Process States

Faculty of Information Technology

Processes can be in various "states"

1. **Created** (ie: loaded into memory)
2. **Waiting** to be run by CPU
3. O/S **runs** the process
4. If process needs a resource, "**blocked**" (ie: waiting) until it gets the resource.



wikipedia

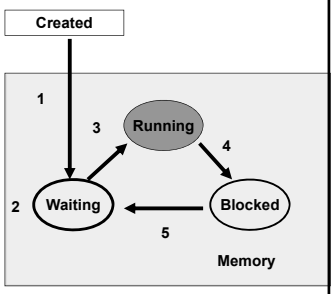
15

Process States

Faculty of Information Technology

Processes can be in various "states"

1. **Created** (ie: loaded into memory)
2. **Waiting** to be run by CPU
3. O/S **runs** the process
4. If process needs a resource, "**blocked**" (ie: waiting) until it gets the resource.
5. It then waits again until the OS re-starts the process.



wikipedia

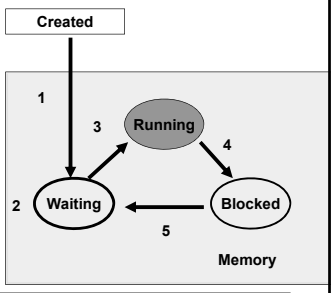
16

Process States

Faculty of Information Technology

Processes can be in various "states"

1. **Created** (ie: loaded into memory)
2. **Waiting** to be run by CPU
3. O/S **runs** the process
4. If process needs a resource, "**blocked**" (ie: waiting) until it gets the resource.
5. It then waits again until the OS re-starts the process.



Keep looping until finished

wikipedia

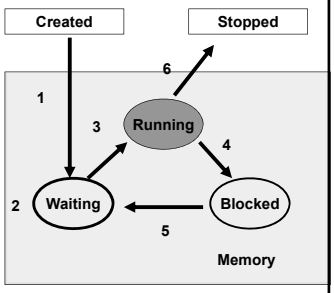
17

Process States

Faculty of Information Technology

Processes can be in various "states"

1. **Created** (ie: loaded into memory)
2. **Waiting** to be run by CPU
3. O/S **runs** the process
4. If process needs a resource, "**blocked**" (ie: waiting) until it gets the resource.
5. It then waits again until the OS re-starts the process.
6. When finished, O/S **stops** the process



wikipedia

18

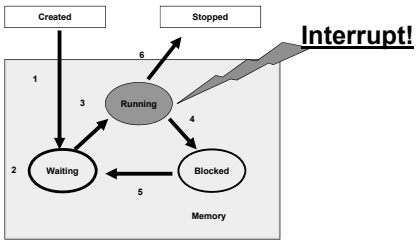
Interrupts

- The O/S runs processes continuously until interrupted
- Interruptions come from
  - Hardware
    - E.g. Ctrl-Alt-Delete keystroke, move mouse, network
  - Programs
    - E.g. runtime error, pause for input, wait for block of data to load from disk

19

Interrupts

- O/S will force current running process into "blocked" state & runs a special "interrupt handler"
  - Often it will pass control to a different process while waiting .



20

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

21

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

Process 1

22

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

Process 1 | Process 2

23

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

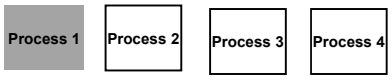
Process 1 | Process 2 | Process 3

24

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

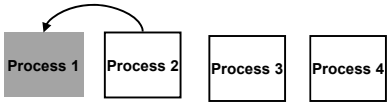


25

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

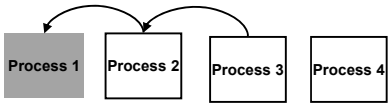


26

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

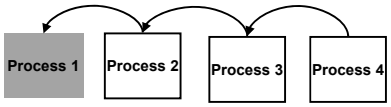


27

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

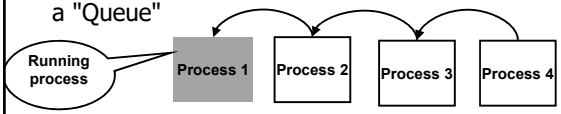


28

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

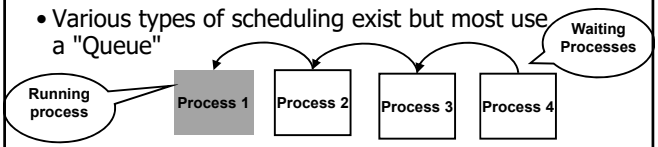


29

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"

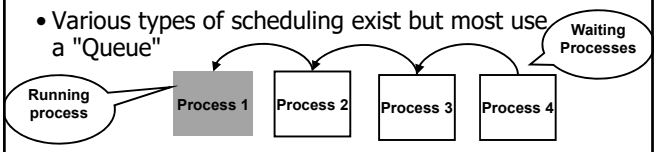


30

Process Scheduling

Faculty of Information Technology

- How does O/S decide when to run each process on which CPU?
  → This is called "**Scheduling**".
- Various types of scheduling exist but most use a "Queue"



- Different types of algorithms to decide which process runs next eg:
  - First-in first-out
  - Pre-emptive
  - Round-robin
  - … + heaps more!

31

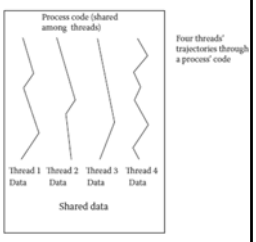
Concurrent programming

- Various O/S uses different techniques to run processes:
  - **Multiprogramming** – "cooperative"
    - when process waits for I/O, run next process
    - Can get "hangs" if a process does no I/O or doesn't relinquish the CPU
  - **Multitasking** - "pre-emptive"
    - Set a timer on all processes – "fair share" CPU time. When timer stops, switch to next process.
    - Can have priorities eg – low, normal, high, real-time
    - Choice of Queuing algorithm decides which process runs next

32

Concurrent programming -2

- Various O/S uses different techniques to run processes:
  - **Multithreading**
    - Multiple instances of the same code eg same process in memory, but 2 or more execute paths
      - Eg powerpoint – might have 1 thread per presentation
      - Allows O/S to run a process on more than 1 CPU (e.g. 1 per concurrent thread)
      - Very efficient use of resources eg memory
  - **Multiprocessing**
    - Multitasking spread amongst 2 or more CPU's



33

Inter Process Communication

Faculty of Information Technology

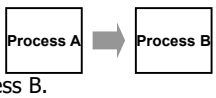
- Processes often need to communicate with each other. This is called **IPC**.
- Operating Systems allow communication between processes via…
  - File sharing
  - Shared memory
  - Signals
  - Sockets
  - Messages
  - Pipes
  - Semaphores
  - Locks

34

IPC one-way example: Pipes

Faculty of Information Technology

- One way communication.



- Process **A** sends data to process B. Process B accepts the data.
- EXAMPLE: Bash "pipe" operator
  → output of one command (process) becomes the input of another command (process).

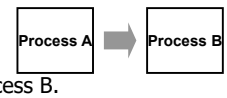
ls | more

35

IPC one-way example: Pipes

Faculty of Information Technology

- One way communication.



- Process **A** sends data to process B. Process B accepts the data.
- EXAMPLE: Bash "pipe" operator
  → output of one command (process) becomes the input of another command (process).

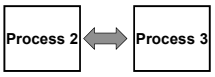
ls more

36

Two-way IPC

UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Information Technology

- Processes can communicate in both directions
  - shared memory
    (bit of memory which acts as like a file)
  - Named pipe
    (special 2 way pipe, acts like a file)
  - Socket
    (either a network or internal interface)
  - Message Queue/Message passing
    (special programming interface – passes data like internal messages/SMS)
  - Semaphore
    (special flag/file which controls access to resources)



37

Summary

- We learnt about the difference between Programs and processes, how they are scheduled and about Inter Process Communications (IPC)
- In the linuxgym chapter 5 lab we will make use of the the IPC process called PIPES

38