# 94.301 Operating Systems and Data Bases

- **Instructor: Thomas Kunz**
- **tkunz@sce.carleton.ca**
- **Location: 5050 MC**
- **Classes: Mon 11:30 1 hour, Tue 13:30 1 hour, Thu 12:30 1 hour**



# 94.301 Operating Systems and Data Bases

- **From calendar:**
  - Management of CPU, processes,memory, files, and data.
  - Implications of concurrency.
  - Concurrent programming, including interprocess communication in distributed systems.
  - Data models and query languages.

## 94.301 Operating Systems and Data Bases

- Textbook: William Stallings, Operating Systems: Internals and Design Principles, 4th edition, Prentice Hall 2001, ISBN 0-13-031999-6.
- Prerequisites: Engineering 94.202 or 94.210, and 94.203
- Four programming assignments, worth 5% - 10%
- Midterm in class, worth 20%
- Final exam in December exam period, worth 45%
- Three (?) TAs, office hours TBD
- Course webpage: http://kunz-pc.sce.carleton.ca/sce301/



## Computer Systems Overview

- Basically Chapter 1 in textbook
- OS mediates between programs and hardware
- Need some basic understanding of computer organization and hardware:
  - processor
  - memory
  - I/O
- “Left as an exercise to the reader”

# Operating System

- **Is a program that controls the execution of application programs**
  - OS must relinquish control to user programs and regain it safely and efficiently
  - Tells the CPU when to execute other pgms
- **Is an interface between the user and hardware**
- **Masks the details of the hardware to application programs**
  - Hence OS must deal with hardware details



# Services Provided by the OS

- **Facilities for Program creation**
  - editors, compilers, linkers, and debuggers
- **Program execution**
  - loading in memory, I/O and file initialization
- **Access to I/O and files**
  - deals with the specifics of I/O and file formats
- **System access**
  - Protection in access to resources and data
  - Resolves conflicts for resource contention

## Services Provided by the OS

- **Error Detection**
  - internal and external hardware errors
    - memory error
    - device failure
  - software errors
    - arithmetic overflow
    - access forbidden memory locations
  - Inability of OS to grant request of application
- **Error Response**
  - simply report error to the application
  - Retry the operation
  - Abort the application



## Services Provided by the OS

- **Accounting**
  - collect statistics on resource usage
  - monitor performance (eg: response time)
  - used for system parameter tuning to improve performance
  - useful for anticipating future enhancements
  - used for billing users (on multiuser systems)

## Evolution of an Operating System

- **Must adapt to hardware upgrades and new types of hardware. Examples:**
  - Character vs graphic terminals
  - Introduction of paging hardware
- **Must offer new services, eg: internet support**
- **The need to change the OS on regular basis place requirements on it's design**
  - modular construction with clean interfaces
  - object oriented methodology

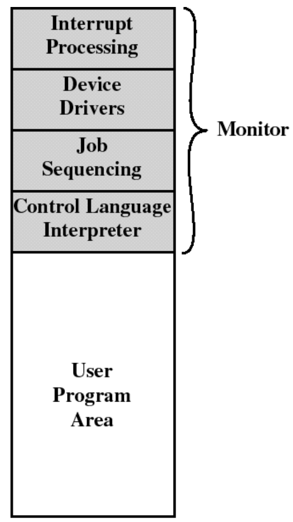


## Simple Batch Systems

- **Are the first operating systems (mid-50s)**
- **The user submit a job (written on card or tape) to a computer operator**
- **The computer operator place a batch of several jobs on a input device**
- **A special program, the monitor, manages the execution of each program in the batch**
- **Resident monitor is in main memory and available for execution**
- **Monitor utilities are loaded when needed**

# The Monitor

- **Monitor reads jobs one at a time from the input device**
- **Monitor places a job in the user program area**
- **A monitor instruction branches to the start of the user program**
- **Execution of user pgm continues until:**
  - end-of-pgm occurs
  - error occurs
- **Causes the CPU to fetch its next instruction from Monitor**




Memory Layout of Resident Monitor



# Job Control Language (JCL)

- **Is the language to provide instructions to the monitor**
  - what compiler to use
  - what data to use
- **Example of job format: ------->>**
- **`$FTN` loads the compiler and transfers control to it**
- **`$LOAD` loads the object code (in place of compiler)**
- **`$RUN` transfers control to user program**

```
$JOB
$FTN
...
FORTRAN
program
...
$LOAD
$RUN
...
Data
...
$END
```

## Job Control Language (JCL)

- **Each read instruction (in user pgm) causes one line of input to be read**

- **Causes (OS) input routine to be invoke**
  - checks for not reading a JCL line
  - skip to the next JCL line at completion of user program



## Batch OS

- **Alternates execution between user program and the monitor program**

- **Relies on available hardware to effectively alternate execution from various parts of memory**

# Desirable Hardware Features

- **Memory protection**
  - do not allow the memory area containing the monitor to be altered by user programs
- **Timer**
  - prevents a job from monopolizing the system
  - an interrupt occurs when time expires

15

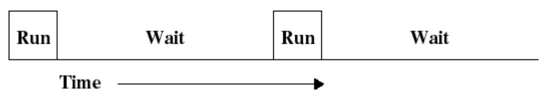
# Desirable Hardware Features

- **Privileged instructions**
  - can be executed only by the monitor
  - an interrupt occurs if a program tries these instructions
- **Interrupts**
  - provides flexibility for relinquishing control to and regaining control from user programs

16

## Multiprogrammed Batch Systems

- I/O operations are exceedingly slow (compared to instruction execution)
- A program containing even a very small number of I/O ops, will spend most of its time waiting for them
- Hence: poor CPU usage when only one program is present in memory

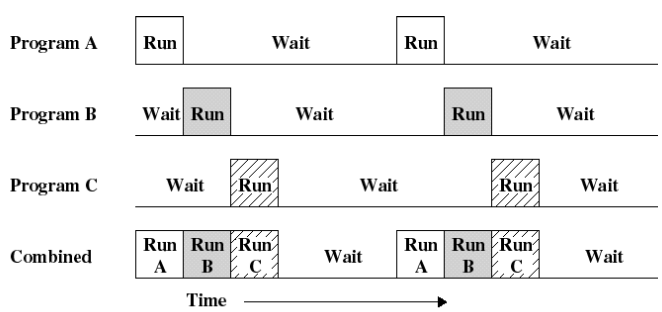



Uniprogramming



## Multiprogrammed Batch Systems

- If memory can hold several programs, then CPU can switch to another one whenever a program is awaiting for an I/O to complete
- This is multitasking (multiprogramming)




Multiiprogramming with three programs

## Requirements for Multiprogramming

- **Hardware support:**
  - I/O interrupts and (possibly) DMA
    - in order to execute instructions while I/O device is busy
  - Memory management
    - several ready-to-run jobs must be kept in memory
  - Memory protection (data and programs)
- **Software support from the OS:**
  - Scheduling (which program is to be run next)
  - To manage resource contention



## Example: three jobs are submitted

| | JOB1 | JOB2 | JOB3 |
|---|---|---|---|
| Type of job | Heavy compute | Heavy I/O | Heavy I/O |
| Duration | 5 min. | 15 min. | 10 min. |
| Memory req. | 50K | 100 K | 80 K |
| Need disk? | No | No | Yes |
| Need terminal | No | Yes | No |
| Need printer? | No | No | Yes |

- **Almost no contention for resources**
- **All 3 can run in minimum time in a multitasking environment (assuming JOB2/3 have enough CPU time to keep their I/O operations active)**

## Advantages of Multiprogramming

| | Uniprogramming | Multiprogramming |
|---|---|---|
| Processor use | 17% | 33% |
| Memory use | 30% | 67% |
| Disk use | 33% | 67% |
| Printer use | 33% | 67% |
| Elapsed time | 30 min. | 15 min. |
| Throughput rate | 6 jobs/hr | 12 jobs/hr |
| Mean response time | 18 min. | 10 min. |



## Time Sharing Systems (TSS)

- **Batch multiprogramming does not support interaction with users**
- **TSS extends multiprogramming to handle multiple interactive jobs**
- **Processor's time is shared among multiple users**
- **Multiple users simultaneously access the system through terminals**

## Time Sharing Systems (TSS)

- **Because of slow human reaction time, a typical user needs 2 sec of processing time per minute**
- **Then (about) 30 users should be able to share the same system without noticeable delay in the computer reaction time**
- **The file system must be protected (multiple users…)**



## Difficulties with OS Design

- **Improper synchronization**
  - ensure a program waiting for an I/O device receives the signal
- **Failed mutual exclusion**
  - must permit only one program at a time to perform a transaction on a portion of data
- **Deadlock**
  - It might happen that 2 or more pgms wait endlessly after each other to perform an operation.

## An example of deadlock

- Program A wants to copy from disk1 to disk2 and takes control of disk1
- Program B wants to copy from disk2 to disk1 and takes control of disk2

- Program A must wait that program B releases disk2 and program B must wait that program A releases disk1
- Program A and B will wait forever



## Major Achievements of OS

- To meet the difficult requirements of multiprogramming and time sharing, there have been 5 major achievements by OS:
  - Processes
  - Memory management
  - Information protection and security
  - Scheduling and resource management
  - System structure

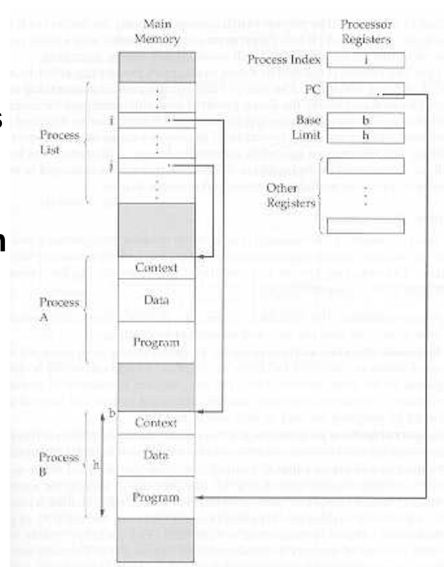
# Process

- **Introduced to obtain a systematic way of monitoring and controlling pgm execution**
- **A process is an executable program with:**
  - associated data (variables, buffers…)
  - execution context: ie. all the information that
    - the CPU needs to execute the process
      - content of  the processor registers
    - the OS needs to manage the process:
      - priority of the process
      - the event (if any) after which the process is waiting
      - other data (that we will introduce later)



# A simple implementation of processes

- **The process index register contains the index into the process list of the currently executing process (B)**
- **A process switch from B to A consist of storing (in memory) B's context and loading (in CPU registers) A's context**
- **A data structure that provides flexibility (to add new features)**



# Memory Management

- **The key contribution is virtual memory**
- **It allows programs to address memory from a logical point of view without regard to the amount that is physically available**
- **While a program is running only a portion of the program and data is kept in (real) memory**
- **Other portions are kept in blocks on disk**
    - the user has access to a memory space that is larger than real memory



# Virtual Memory

- **All memory references made by a program are to virtual memory which can be either**
    - a linear address space
    - a collection of segments (variable-length blocks)
- **The hardware (mapper) must map virtual memory address to real memory address**
- **If a reference is made to a virtual address not in memory, then**
    - (1) a portion of the content of real memory is swapped out to disk
    - (2) the desired block of data is swapped in

## File System

- **Implements long-term store (often on disk)**
- **Information stored in named objects called files**
  - a convenient unit of access and protection for OS
- **Files (and portions) may be copied into virtual memory for manipulation by programs**



## Security and Protection

- **Access control to resources**
  - forbid intruders (unauthorized users) to enter the system
  - forbid user processes to access resources which they are not authorized to

## Scheduling and Resource Management

- **Differential responsiveness**
  - discriminate between different classes of jobs
- **Fairness**
  - give equal and fair access to all processes of the same class
- **Efficiency**
  - maximize throughput, minimize response time, and accommodate as many users as possible



## Key Elements for Scheduling

- **OS maintains queues of processes waiting for some resource**
  - Short term queue of processes in memory ready to execute
    - The dispatcher (short term scheduler) decides who goes next
  - Long term queue of new jobs waiting to use the system
    - OS must not admit too many processes
  - A queue for each I/O device consisting of processes that want to use that I/O device

## System Structure

- **Because of it’s enormous complexity, we view the OS system as a series of levels**
- **Each level performs a related subset of functions**
- **Each level relies on the next lower level to perform more primitive functions**
- **Well defined interfaces: one level can be modified without affecting other levels**
- **This decomposes a problem into a number of more manageable sub problems**



## Characteristics of Modern Operating Systems

- **New design elements were introduced recently**
- **In response to new hardware development**
  - multiprocessor machines
  - high-speed networks
  - faster processors and larger memory
- **In response to new software needs**
  - multimedia applications
  - Internet and Web access
  - Client/Server applications

## Microkernel architecture

- **Only a few essential functions in the kernel**
  - primitive memory management (address space)
  - Interprocess communication (IPC)
  - basic scheduling
- **Other OS services are provided by processes running in user mode (servers)**
  - device drivers, file system, virtual memory…
- **More flexibility, extensibility, portability…**
- **A performance penalty by replacing service calls with message exchanges between process...**



## Multithreading

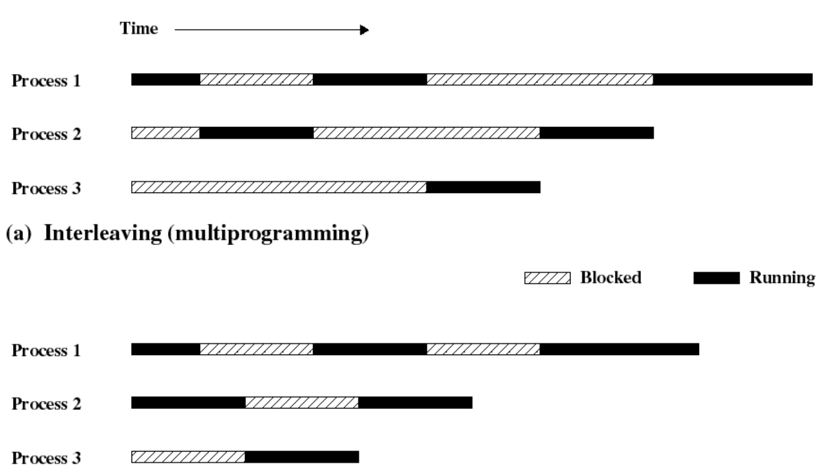
- **A process is a collection of one or more threads that can run simultaneously**
- **Useful when the application consists of several tasks that do not need to be serialized**
- **Gives the programmer a greater control over the timing of application-related events**
- **All threads within the same process share the same data and resources and a part of the process's execution context**
- **It is easier to create or destroy a thread or switch among threads (of the same process) than to do these with processes**

# Symmetric Multiprocessing (SMP)

- A computer with multiple processors
- Each processor can perform the same functions and share same main memory and I/O facilities (symmetric)
- The OS schedule processes/threads across all the processors (real parallelisme)
- Existence of multiple processors is transparent to the user.
- Incremental growth: just add another CPU!
- Robustness: a single CPU failure does not halt the system, only the performance is reduced.



# Example of parallel execution on SMP




(b) Interleaving and overlapping (multiprocessing)