Now it begins…

Operating System Concepts Overview

Operating System Objectives

- Convenience
  - Makes the computer more convenient to use.
- Efficiency
  - Allows computer system resources to be used in an efficient manner.
- Ability to evolve
  - Permit effective development, testing, and introduction of new system functions without interfering with service.

Operating System Services

- Program Development
  - Editors, debuggers, frameworks
- Program Execution
  - Initialization, scheduling
- Access to I/O Devices
  - Uniform interface, hides details
- Controlled Access to Files
  - Authorization, sharing, caching

Operating System Services

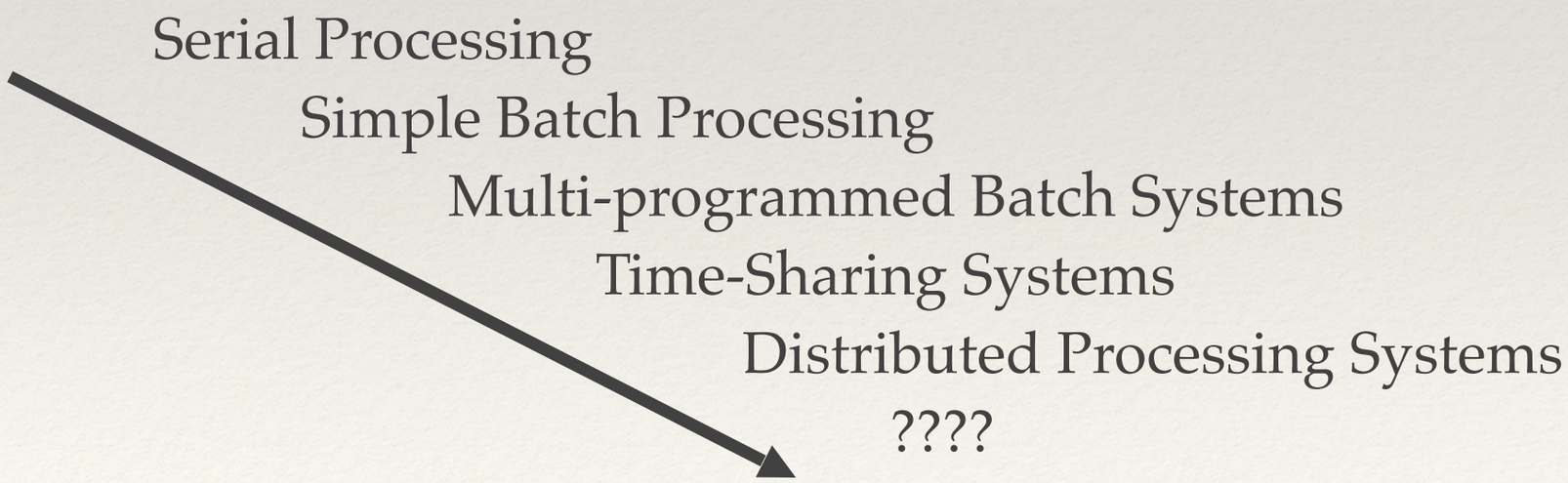
- System Access
  - Protection, authorization, resolve conflicts
- Error Detection and Response
  - Hardware errors: memory error or device failure
  - Software errors: arithmetic errors, access forbidden memory locations, allocation errors
- Accounting
  - Collect statistics (billing)
  - Monitor performance
  - Used to anticipate future enhancements

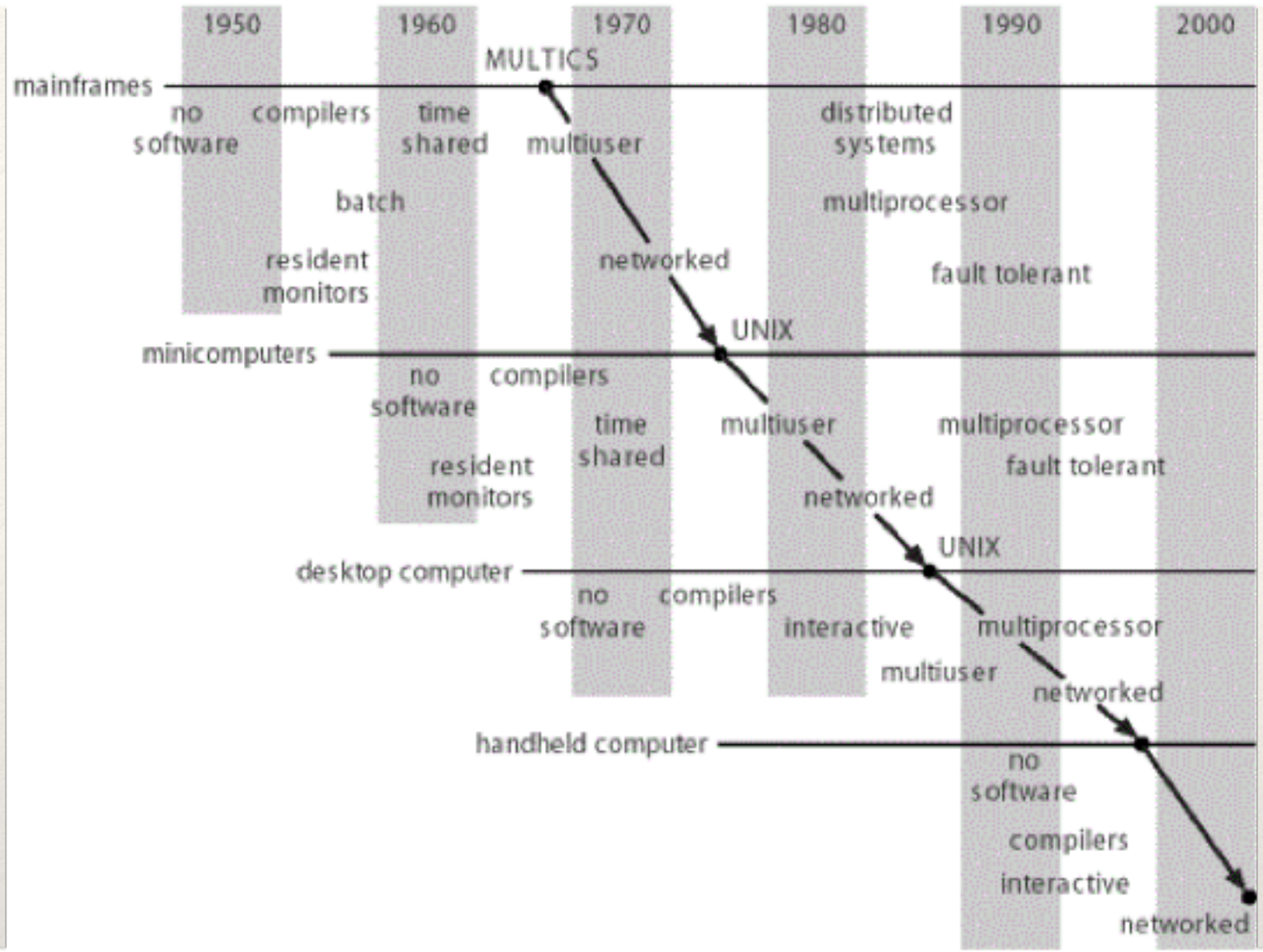
Operating System as a Resource Manager

- OS executes same way as ordinary computer software - it is a set of computer programs.
- The key difference is
  - to manage resources
  - to Schedule processor to execute programs
- Kernel
  - Portion of operating system that is in main memory
  - Contains most frequently used functions

Evolution of Operating Systems

- Operating systems have evolved because
  - New types of hardware and hardware upgrades
  - Development of new services and needs
  - Fixes to OS faults





Monkey see…monkey do…

The Evolution of Operating Systems

Migration of Operating System Concepts and Features

http://www.myassignmenthelp.net/migration-of-operating-system-concepts-and-features.php

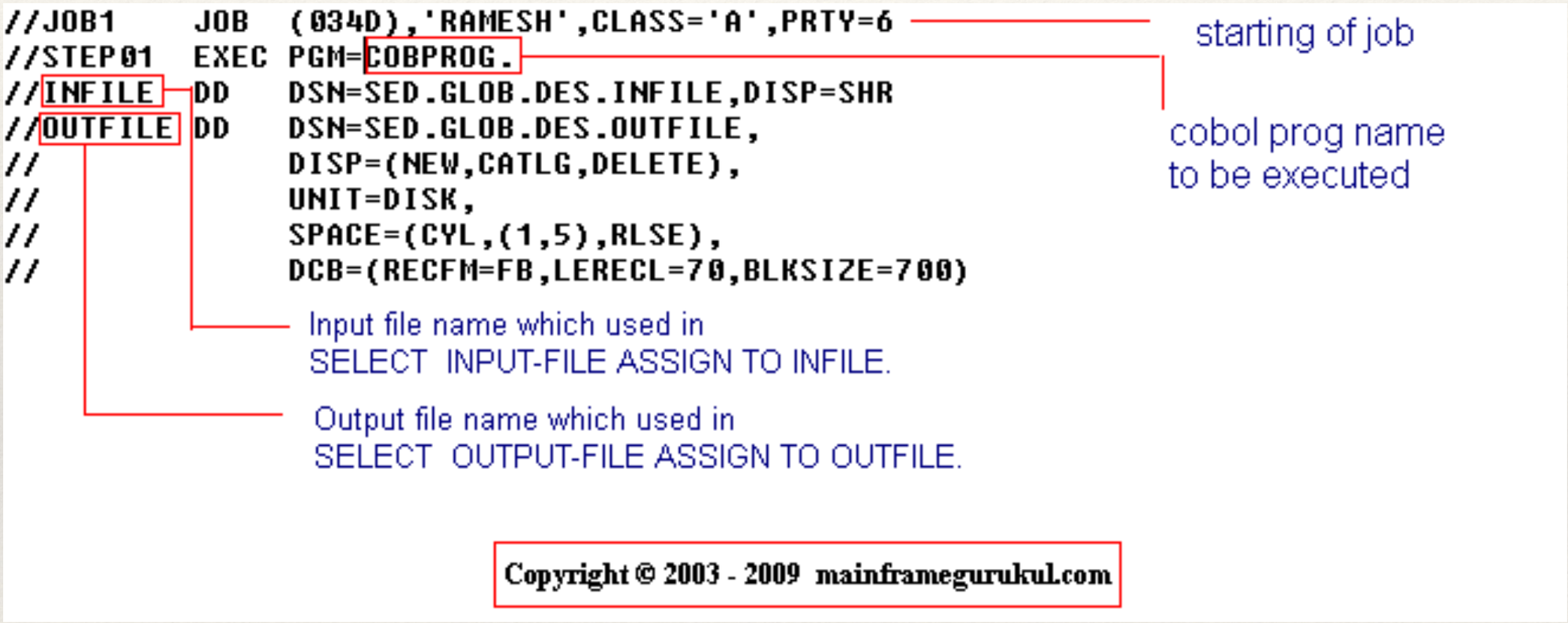
Serial Processing

- No operating system.
- Machines run from a console with display lights and switches.
- Users had to schedule time on the machine.
- Setup included loading the compiler, source program, saving compiled program, and loading and linking.

Simple Batch Systems

- **Monitors**
  - Software that controls the running programs and "***batches***" jobs together.
  - Program branches back to monitor when finished.
  - Resident monitor is in main memory and available for execution.
- **Job Control Language (JCL)**
  - Special type of programming language.
  - Provides instructions to the monitor (what compiler/data to use).
- **Hardware Features**
  - *Memory protection* - does not allow the memory area containing the monitor to be altered.
  - *Timer* - prevents a job from monopolizing the system.

JCL Example

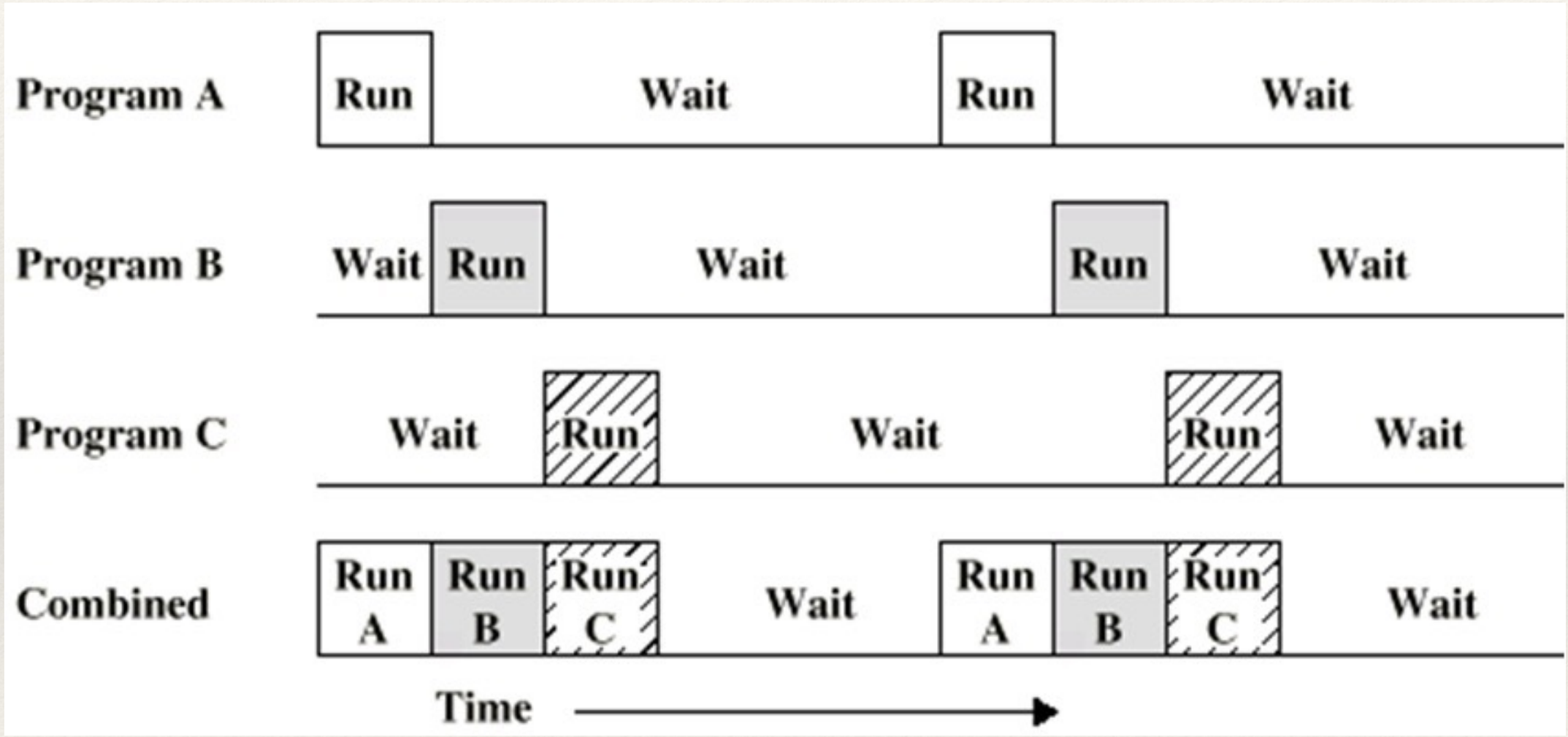


http://www.mainframegurukul.com/srcsinc/drona/programming/languages/jcl/jcl.chapter1.html

Multiprogramming

- Executing a number of programs simultaneously on one processor.
- Number of processes reside in main memory at the same time.
- OS selects one process and runs it until a wait condition (e.g. I/O) occurs and then it switches to another process.
- CPU is not idle if there are processes to be run.

Multiprogramming



https://thesampahman.wordpress.com/page/4/

Time Sharing Systems

- Allow several users to interact at the same time.
- Emphasizes response time over processor use.
- Multics (1964)
  - First operating system to provide a hierarchical file system.
  - Symbolic links between directories were supported.
  - First OS to have a command processor implemented as ordinary user code - inspired the Unix shell.
  - One of the first written in a high level language (PL/1).

Distributed Processing Systems

- **Symmetric Multiprocessing**
  - There are multiple processors.
  - The processors share same main memory and I/O facilities.
  - All processors can perform the same functions.
- **Distributed Operating Systems**
  - Provides the illusion of a single main memory and single secondary memory space.
  - Used for distributed computing.

**Multics was one of the earliest multiprocessor systems*

And the winner is…

Major Achievements

- Processes
- Memory Management
- Information protection and security
- Scheduling and resource management
- System structure

Processes

- **Processes** are the fundamental structure of operating systems.
  - A unit of activity characterized by a sequential thread of execution, current state, and an associated set of system resources
- Processes solved the problems introduced by
  - Multiprogramming batch operations
  - Time sharing
  - Real-time transaction systems
- Principle tool available to system programmers in developing multi-tasking systems was the **interrupt**.

Processes

- Processes consist of three components:
  - An executable program
  - Associated data (variables, workspace, buffers, stacks, *etc*.)
  - The execution context of the program
- Problems in the coordination of processes:
  - Improper synchronization
  - Failed mutual exclusion
  - Non-determinate program operation
  - Deadlocks

Processes in Posix

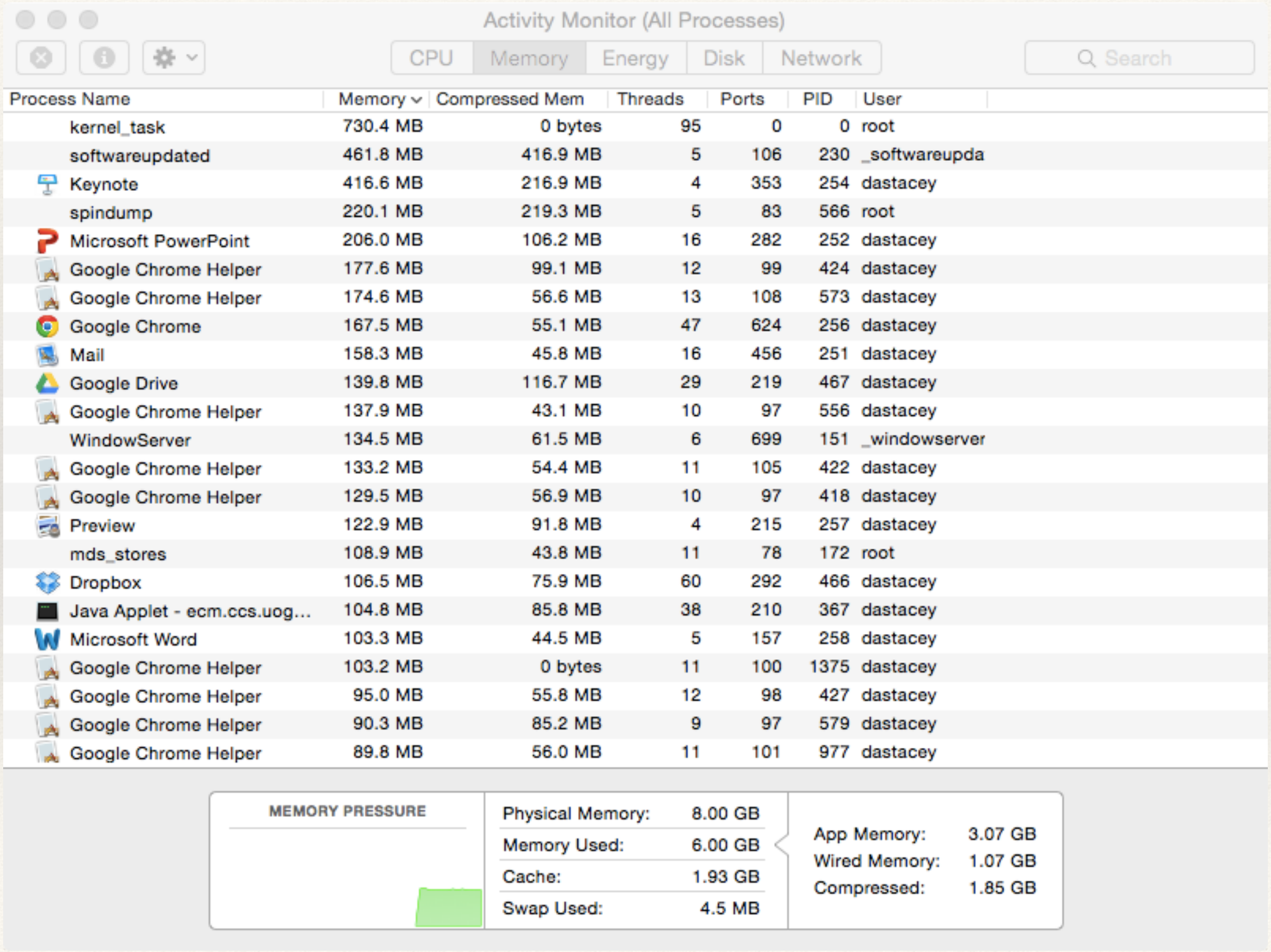
dastacey — bash — 127×38

```
Deb-MacBook-Air:~ dastacey$ ps -ax
  PID TTY           TIME CMD
    1 ??         0:28.23 /sbin/launchd
   40 ??         0:05.67 /usr/sbin/syslogd
   41 ??         0:05.55 /usr/libexec/UserEventAgent (System)
   43 ??         0:01.21 /usr/libexec/kextd
   44 ??         0:11.00 /System/Library/Frameworks/CoreServices.framework/Versions/A/Frameworks/FSEvents.framework/Versions/A
   46 ??         1:10.32 /opt/cisco/anyconnect/bin/vpnagentd -execv_instance
   47 ??         0:00.12 /usr/libexec/thermald
   48 ??         0:47.24 /System/Library/CoreServices/backupd.bundle/Contents/Resources/mtmd
   50 ??         0:00.78 /System/Library/CoreServices/appleeventsd --server
   51 ??         0:06.00 /usr/libexec/configd
   52 ??         0:01.59 /System/Library/CoreServices/powerd.bundle/powerd
   55 ??         0:14.46 /usr/libexec/airportd
   57 ??         0:00.32 /usr/libexec/warmd
   58 ??         1:47.59 /System/Library/Frameworks/CoreServices.framework/Frameworks/Metadata.framework/Support/mds
   62 ??         0:00.02 /Library/Application Support/Seagate/TBLoopDriveParams
   63 ??         0:00.05 /System/Library/CoreServices/iconservicesd
   64 ??         0:00.03 /System/Library/CoreServices/iconservicesagent
   65 ??         0:00.50 /usr/libexec/diskarbitrationd
   68 ??         0:16.19 /usr/libexec/coreduetd
   69 ??         0:00.10 /System/Library/CoreServices/backupd.bundle/Contents/Resources/backupd-helper -launchd
   70 ??         0:00.03 /usr/libexec/wdhelper
   72 ??         0:04.68 /System/Library/CoreServices/backupd.bundle/Contents/Resources/mtmfs --tcp --resvport --listen localh
   73 ??         0:00.07 /System/Library/PrivateFrameworks/WirelessDiagnostics.framework/Support/awdd
   74 ??         0:11.79 /usr/libexec/opendirectoryd
   75 ??         0:01.10 /usr/sbin/wirelessproxd
   77 ??         0:03.06 /System/Library/PrivateFrameworks/ApplePushService.framework/apsd
   78 ??         0:13.30 /System/Library/CoreServices/launchservicesd
   79 ??         0:00.11 /System/Library/PrivateFrameworks/MobileDevice.framework/Versions/A/Resources/usbmuxd -launchd
   80 ??         0:06.56 /usr/sbin/securityd -i
   82 ??         0:02.45 /usr/libexec/locationd
   85 ??         0:06.07 /usr/sbin/blued
   86 ??         0:00.03 autofsd
   88 ??         0:00.17 /System/Library/PrivateFrameworks/Heimdal.framework/Helpers/kdc
   90 ??         0:03.08 /usr/sbin/mDNSResponder
   92 ??         0:00.02 /usr/libexec/stackshot -t -O
   93 ??         0:02.98 /System/Library/PrivateFrameworks/GenerationalStorage.framework/Versions/A/Support/revisiond
```

Memory Management

- **Responsibilities** of memory management:
  - Process isolation
  - Automatic allocation and management
  - Support of modular programming
  - Protection and access control
  - Long-term storage
- These requirements typically met by:
  - Virtual memory
  - File system facilities

Memory Management



Information Protection and Security

- Time-sharing and computer networks require:
  - Availability
  - Confidentiality
  - Data integrity
  - Authenticity