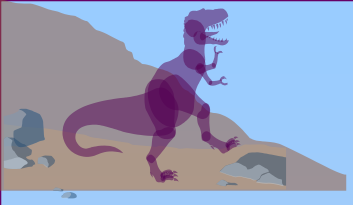
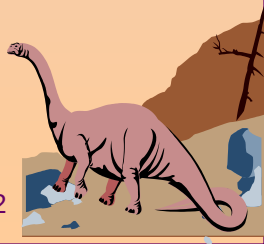
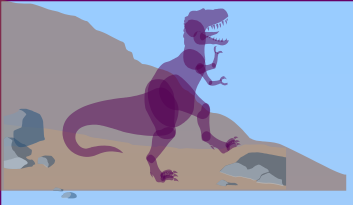


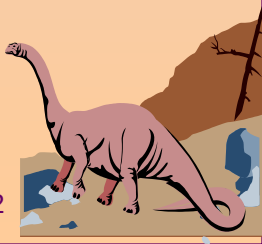
Evolution/Development of Operating Systems

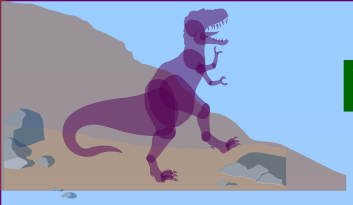




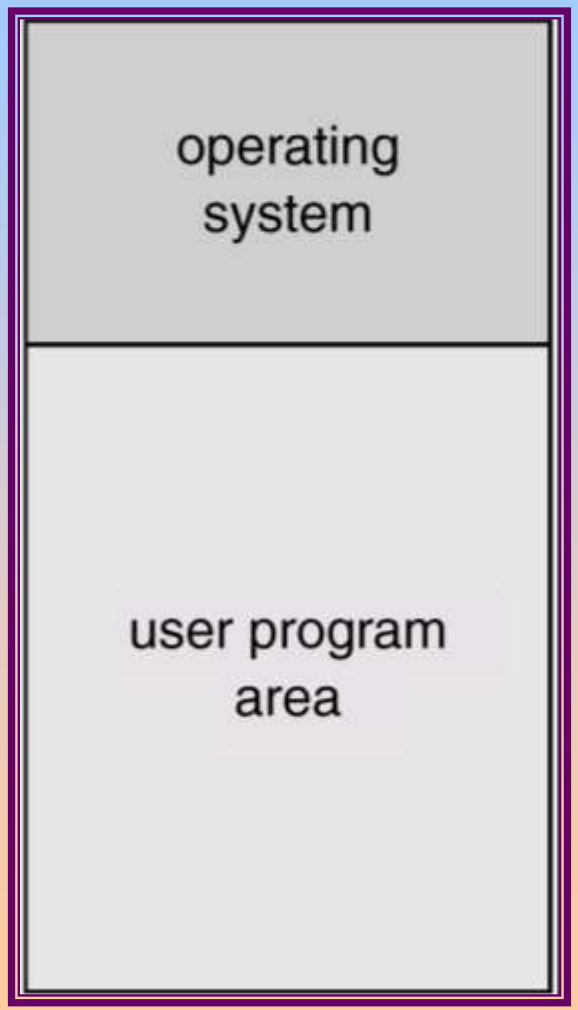
Simple Batch Systems

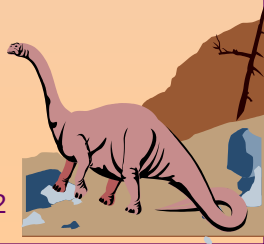
- Hire an operator
- User ≠ operator
- Add a card reader
- Reduce setup time by batching similar jobs
- Automatic job sequencing – automatically transfers control from one job to another. First rudimentary operating system.
- Resident monitor
  - initial control in monitor
  - control transfers to job
  - when job completes control transfers back to monitor

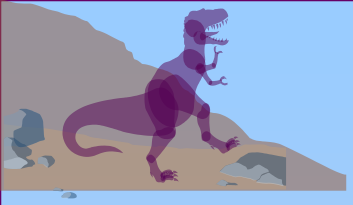




Memory Layout for a Simple Batch System

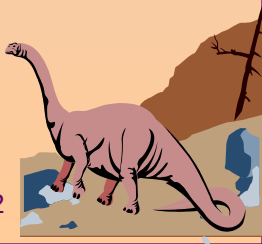


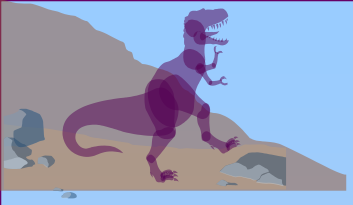




Control Cards

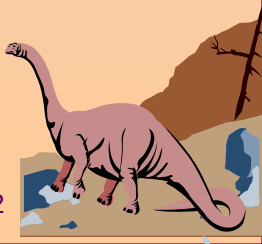
- Problems
  1. How does the monitor know about the nature of the job (e.g., Fortran versus Assembly) or which program to execute?
  2. How does the monitor distinguish
     (a) job from job?
     (b) data from program?
- Solution
  - ☞ Introduce control cards

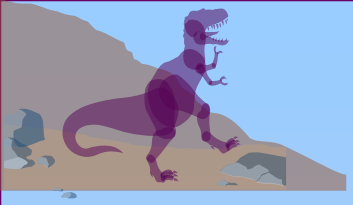




Control Cards (Cont.)

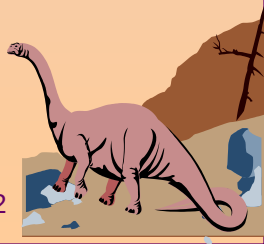
- Parts of resident monitor
  - Control card interpreter – responsible for reading and carrying out instructions on the cards.
  - Loader – loads systems programs and applications programs into memory.
  - Device drivers – know special characteristics and properties for each of the system's I/O devices.
- Problem: Slow Performance – I/O and CPU could not overlap; card reader very slow.
- Solution: Off-line operation – speed up computation by loading jobs into memory from tapes and card reading and line printing done off-line.

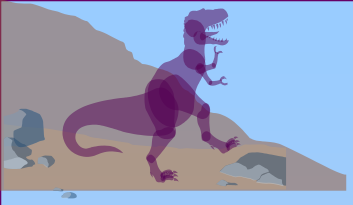




Spooling

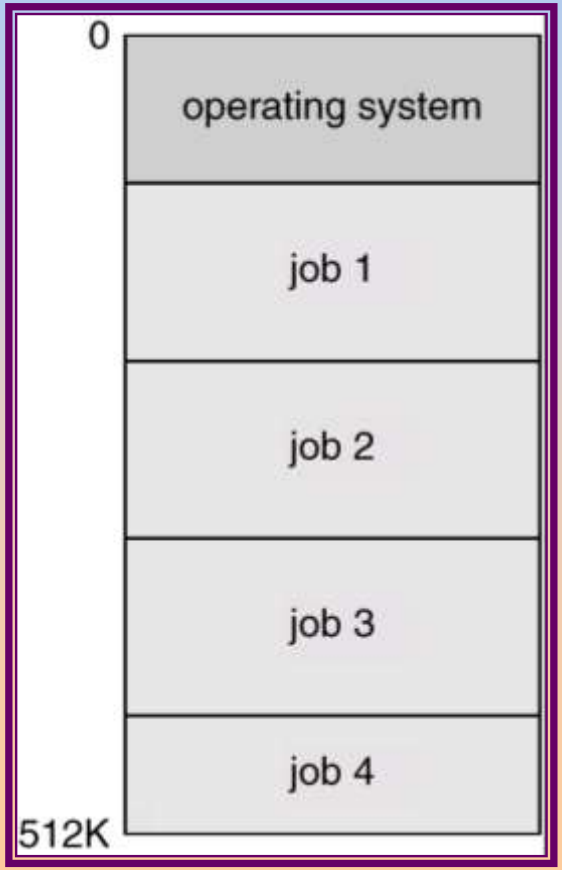
- Overlap I/O of one job with computation of another job. While executing one job, the OS:
  - Reads next job from card reader into a storage area on the disk (job queue).
  - Outputs printout of previous job from disk to printer.
- *Job pool* – data structure that allows the OS to select which job to run next in order to increase CPU utilization.
- Acronym for Simultaneous Peripheral Operation On-line.

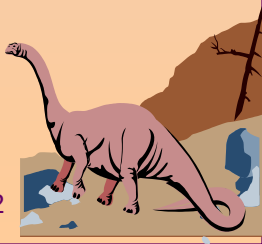


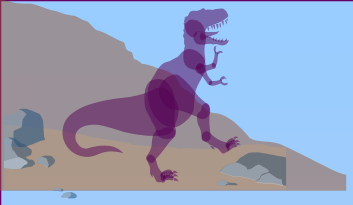


Multiprogrammed Batch Systems

- Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.
- *Job pool* – data structure that allows the OS to select which job to run next in order to increase CPU utilization.

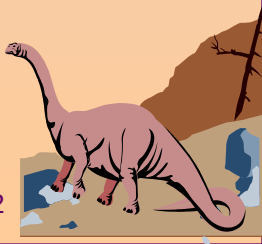


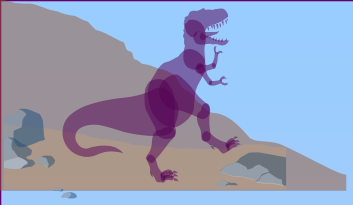




Multiprogramming

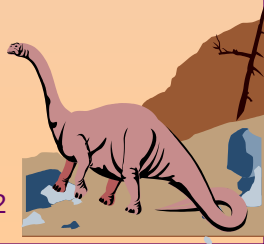
- Multiprogramming: In-order to increase CPU utilization by organizing jobs such that CPU has always one job to execute.
- CPU burst time & I/O burst time
- CPU does not sit idle, it is executing one or the other job at any given time.
- Job pool to RAM (Job Scheduling)





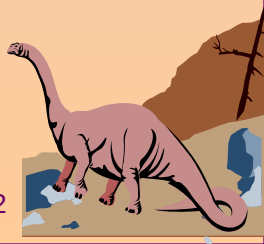
OS Features Needed for Multiprogramming

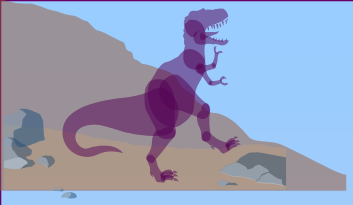
- Memory management – the system must allocate the memory to several jobs.
- CPU scheduling – the system must choose among several jobs ready to run.
- Allocation of devices.



Time-Sharing Systems–Interactive Computing

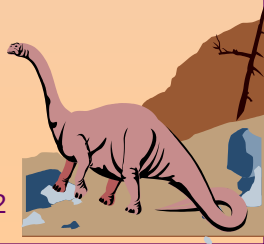
- Multiple jobs are executed by CPU switching b/w them.
- Little CPU time is given to each user.
- Switches occur so frequently that each user is given the impression that he/she owns the system.
- But actually, 1 computer is being shared among multiple users.

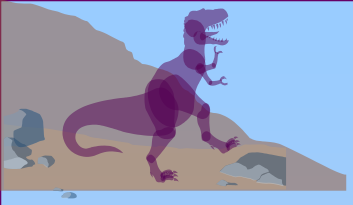




Desktop Systems

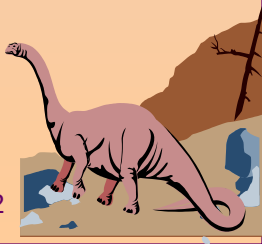
- *Personal computers* – computer system dedicated to a single user.
- I/O devices – keyboards, mouse, display screens, printers.
- User convenience and responsiveness.
- Often individuals have sole use of computer and do not need advanced CPU utilization or protection features.
- May run several different types of operating systems (Windows, MacOS, UNIX, Linux)

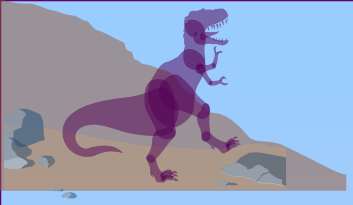




Parallel Systems

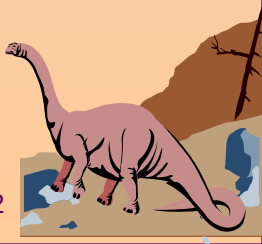
- Single processor systems
- Multiprocessor systems with more than one CPU in close communication.
- *Tightly coupled system* – processors share memory and a clock; communication usually takes place through the shared memory.
- Advantages of parallel systems:
  - Increased *throughput*
  - Economical
  - Increased reliability
    - graceful degradation
    - fault-tolerant systems

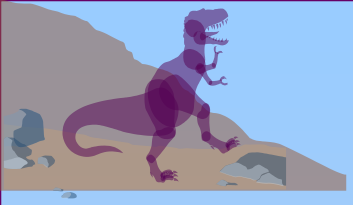




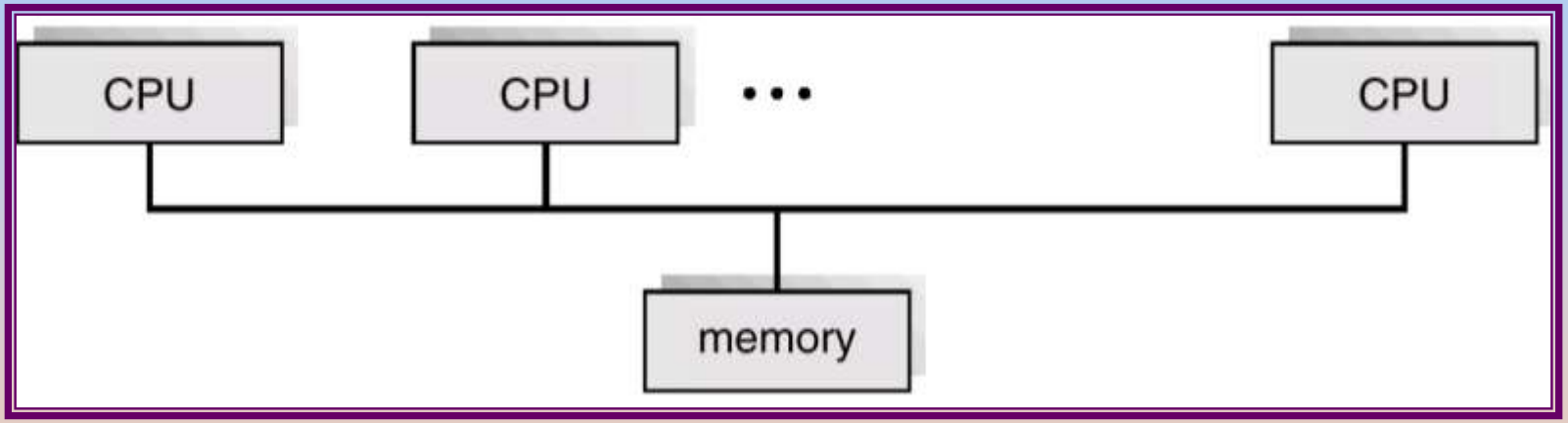
Parallel Systems (Cont.)

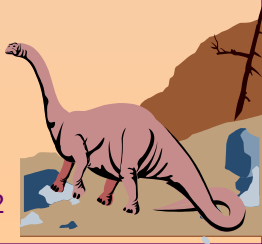
- *Symmetric multiprocessing (SMP)*
  - Each processor runs an identical copy of the operating system.
  - Many processes can run at once without performance deterioration.
  - Most modern operating systems support SMP
  - Jobs & resources are shared dynamically among various processes.
  - Peer to peer computing
- *Asymmetric multiprocessing*
  - Each processor is assigned a specific task; master processor schedules and allocates work to slave processors.
  - Master-slave or client-server architecture
  - More common in extremely large systems

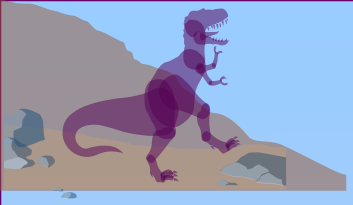




Symmetric Multiprocessing Architecture



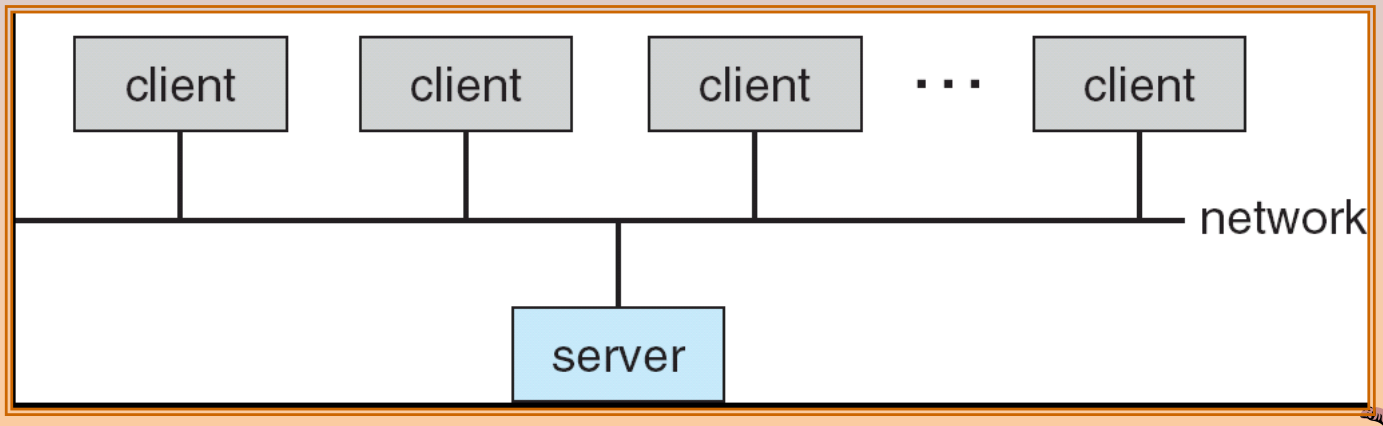


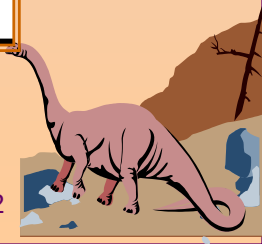


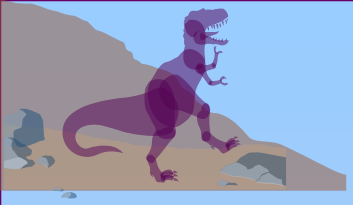
Computing Environments

Client-Server Computing

- Many systems now **servers**, responding to requests generated by **clients**
  - **Compute-server** provides an interface to client to request services (i.e. database)
  - **File-server** provides interface for clients to store and retrieve files

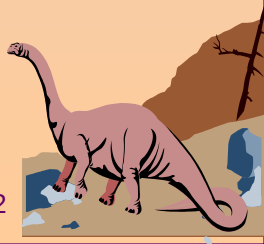


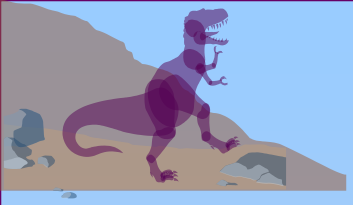




Peer-to-Peer Computing

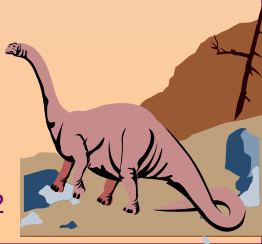
- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - May each act as client, server or both

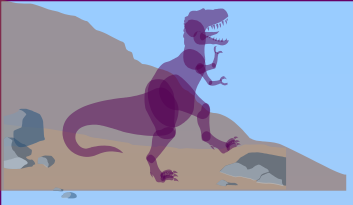




Distributed Systems

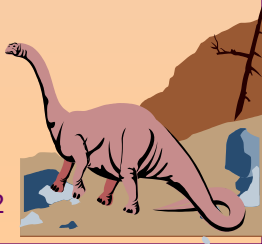
- Collection of physically separate, heterogeneous computer systems that are networked to provide the users with access to various resources.
- Distribute the computation among several physical processors.
- *Loosely coupled system* – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or telephone lines.
- Wired Network: Cu wires, fibre strands
- Wireless Network: Satellites, Radio frequency, Bluetooth and Wi-Fi.

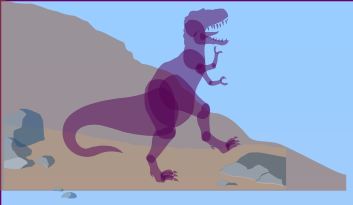




Distributed Systems (cont)

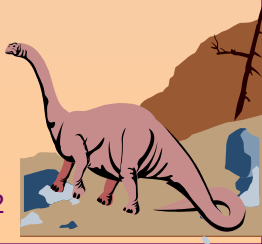
- Advantages of distributed systems.
  - Resource Sharing
  - Computation speed up – load sharing
  - Reliability
  - Communication
- Requires networking infrastructure
- Local area networks (LAN) or Wide area networks (WAN) or Metropolitan Area Network (MAN)
- May be either client-server or peer-to-peer systems.

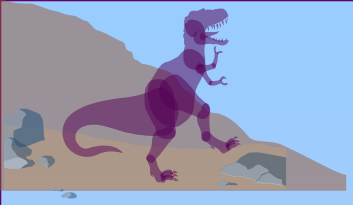




Real-Time Systems

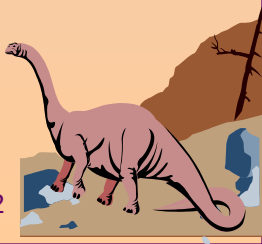
- Used when there are rigid time requirements on the operation of a processor or flow of data.
- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, weapon systems, home appliance controllers and some display systems.
- Well-defined fixed-time constraints.
- Processing must be done within the defined constraints or the system will fail.
- Real-Time systems may be either *hard* or *soft* real-time.

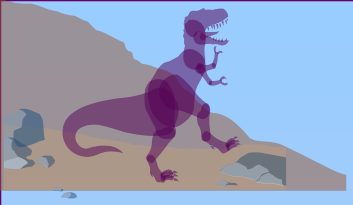




Real-Time Systems (Cont.)

- Hard real-time:
  - Guarantees that critical tasks should complete on time.
  - All delays in the system be bounded
  - Strict deadlines
  - Conflicts with time-sharing systems, not supported by general-purpose operating systems.
- Soft real-time
  - Less restrictive than hard RT systems
  - Critical RT tasks get priority over other tasks & retain that priority until that task is complete.
  - Bcoz of their lack of deadline support, they have limited utility in industrial control & robotics
  - Useful in applications like multimedia, virtual reality, scientific projects like undersea exploration & planetary rovers.





Handheld Systems

- Personal Digital Assistants (PDAs)
- Cellular telephones
- Issues:
  - Limited memory
  - Slow processors
  - Small display screens.