Section 5
Concurrent Computing

1. Concurrent systems
2. Processes
3. Threads

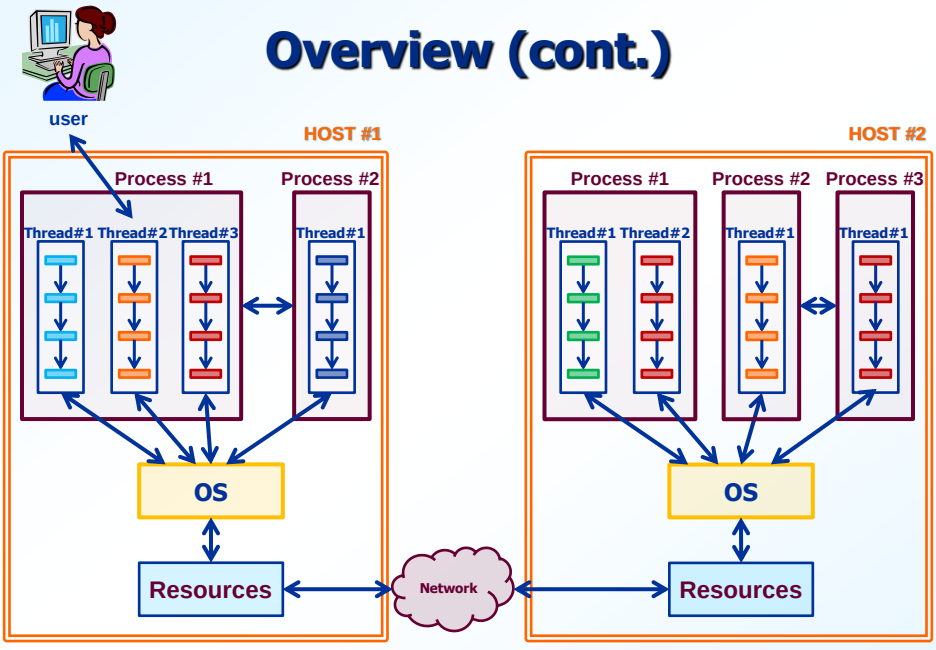
Section 5.1
Concurrent Systems

1. Overview
2. Types of concurrent systems
3. Issues in concurrency

5.1.1 Overview

- What is concurrency?
  - doing more than one thing at a time
- What is concurrent computing?
  - a program with more than one control flow
  - a large *system* (i.e. application) can be
    - multithreaded
    - multi-process
    - distributed

Overview (cont.)



5.1.2 Types of Concurrent Systems

- Distributed systems
- Multi-process systems
- Multithreaded systems

Distributed Systems

- What is a distributed system?
  - system (large program) that executes over multiple physical hosts
    - typically in different locations, cities or countries
  - each host has different resources
    - different filesystems
    - different processing capabilities
    - ... everything ...
  - hosts must be networked together in order to communicate
    - intranet: network internal to an organization
    - internet: network external to all organizations (public network)

Distributed Systems (cont.)

- Why a distributed system?
  - users are in different physical locations
  - server hosts are in different physical locations
  - single host has insufficient processing power
  - example:
    - server computers store data
    - client computers access data

Multi-Process Systems

- What is a multi-process system?
  - system made up of multiple processes (executables)
  - multiple processes can be:
    - different executables
    - multiple copies of the same executable
  - each with independent control flow(s) and virtual memory
  - processes typically need to communicate with each other
    - must use inter-process communication (IPC) techniques

Multi-Process Systems (cont.)

- Why a multi-process system?
  - system has different tasks to perform
  - tasks are very independent from each other
  - tasks use different resources from each other
  - example:
    - one client process to communicate with user
    - one server process to handle user requests
    - one process to regulate access to database

Multithreaded Systems

- What is a multithreaded system?
  - a process with multiple control flows
  - share the process
    - virtual memory
    - address space
    - resources
  - different threads may need to synchronize
  - possible issues with
    - race conditions
    - deadlocks

Multithreaded Systems (cont.)

- Why a multithreaded system?
  - process has different tasks to perform
  - tasks are somewhat dependent on each other
  - example:
    - one thread blocks and waits for user input
    - other threads deal with user requests

5.1.3 Issues in Concurrency

- Shared resources
  - multiple processes or threads may need same resource
  - operations that make changes to resources must be *atomic*
    - processes accessing same file
      - file can be locked
    - threads accessing shared variable
      - semaphores
      - mutexes

Issues in Concurrency (cont.)

- Deadlocks
  - when multiple threads are blocked, waiting for a condition that will never occur
    - due to improper handling of semaphores or mutexes
- Races

  - when correctness of program depends on one thread reaching a point in control flow before another thread
    - this order cannot be guaranteed