Operating Systems

Dr. Vivek Nallur (vivek.nallur@ucd.ie)

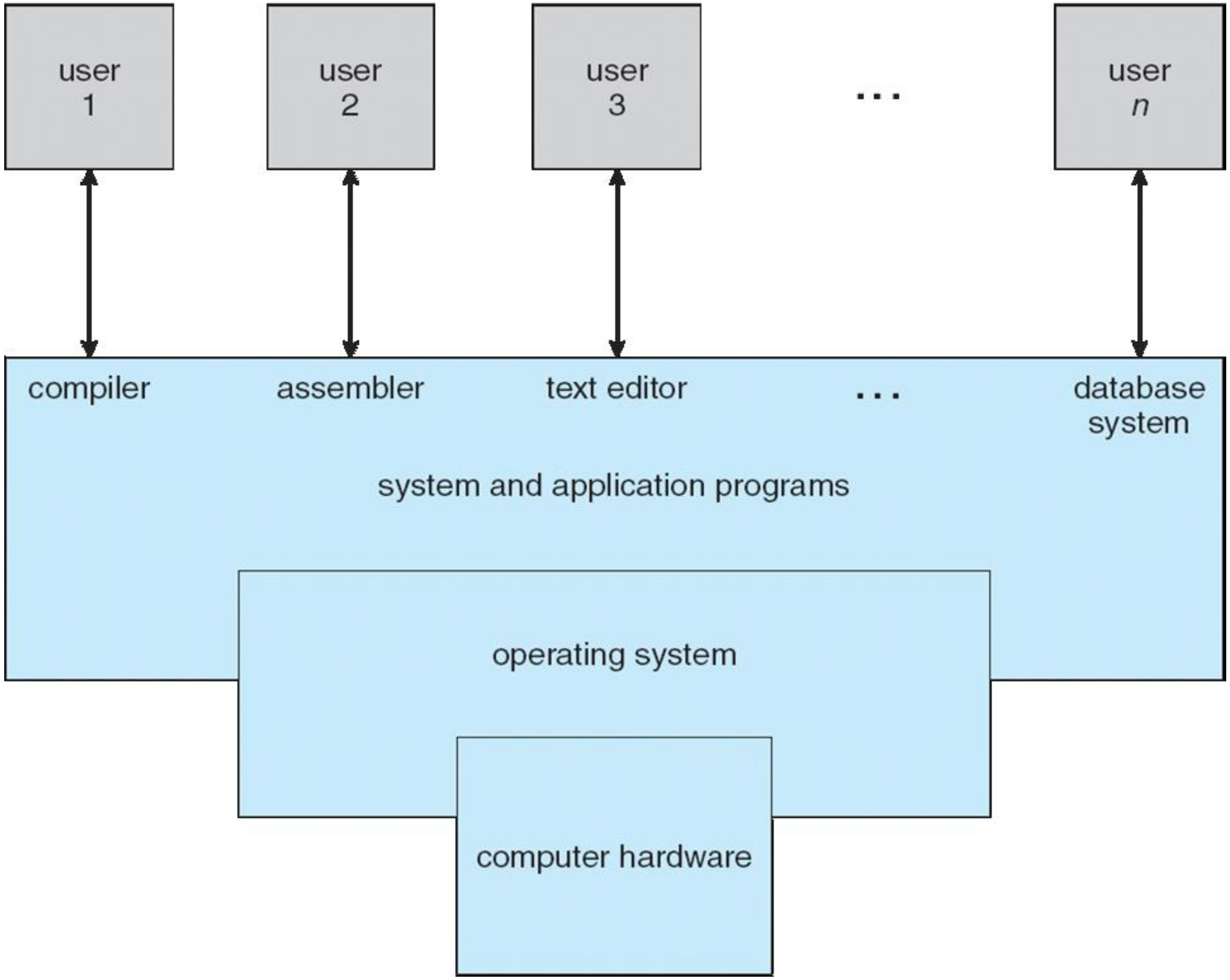
Introduction to Operating Systems

Computer System Components

Four main components in a computer system:

- Hardware
  - basic computing resources (CPU, memory)
- Operating Systems
  - Controls and coordinates the use of the hardware among the various application programs for the various users
- Application programs
  - Use system resources to solve problems or complete tasks
- Users
  - People, machines or other computers

Four Components of a Computer System



What Operating Systems Do?

- Interface between user and hardware
- Control interactions between users and programs
- Provides a controlled and efficient environment for the execution of programs
- Provides mechanisms (functionality) and policies (rules) to manage the whole resources of the system

Operating System Definition

- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is a good approximation
  - But varies wildly
- “The one program running at all times on the computer” is the **kernel**.
- Everything else is either
  - a system program (ships with the operating system) , or
  - an application program.

Why Study Operating Systems?

- Arguments against
  - "very few designers/implementers needed"
  - "all I need to know is in the manual pages"
  - "I'll stick to my favorite OS anyway"

Why Study Operating Systems?

- Arguments for
  - crucial for understanding application-hardware interaction
  - growing need for OSs: mini-OSs in many environments, embedded systems. . .
  - the study of general OSs includes important design and optimisation problems in computer science
    - achievement of flexibility, robustness, security, and performance, whilst keeping things as simple as possible
    - trade-offs are inevitable

What this course is about?

- This course is about understanding the general principles of OS design
  - focus on general-purpose, multi-user systems
  - emphasis on widely applicable concepts
  - stress on problems, solutions, and design choices
- This course is not about specific features of particular OSs: "how do I do X in operating system Y?"

What this course is about?

- To be able to identify core issues, elements, and techniques in any operating system

- To be able to design efficient solutions when faced with problems similar to ones already seen

Major Issues in Operating Systems

- Structure
  - How is the operating system organised
- Sharing
  - How are resources shared between different users and programs
- Accounting
  - How to control resource usage

Major Issues in Operating Systems

- Naming
  - How are resources name (by users or programs)
- Persistence
  - How to make data last when programs have finished
- Protection
  - How to protect users and programs from each other

Major Issues in Operating Systems

- Security
  - How to restrict information flow
- Performance
  - How to handle the system in an efficient way
- Reliability & fault tolerance
  - What to do when something goes wrong

Major Issues in Operating Systems

- Extensibility
  - How to easily add new features
- Scalability
  - What will happen when demands and/or resources increase
- Portability
  - Can the OS work in different hardware configurations?

Major Issues in Operating Systems

- Concurrency
  - How are simultaneous activities created & controlled
- Communication
  - How and with whom can a component/user exchange information

Operating Systes View

- An OS performs two basically unrelated functions:
  - extending the hardware functionality (virtual machine)
  - managing the hardware resources (resource manager)

Virtual or Extended Machine View

- Often called top-down view or user's view
- Present a nice and simple view of the computer use:
  - hide low-level detail of programming the hardware (awkward to handle hardware directly)

Virtual or Extended Machine View

- Build layers of software that provide more and more functionality

- Make the user believe that there are more hardware resources than in reality

Resource Manager View

- Colled bottom-up view or system's view

- A computer can be seen as a set of hardware resources for processing, storing and moving data
  - CPU
  - Memory
  - Input and Output (I/O) etc.

Resource Manager View

- under this view, the OS is there to provide an orderly and controlled allocation of those resources

- Performs functions like time & space multiplexing of resources allowing them to be share by multiple users

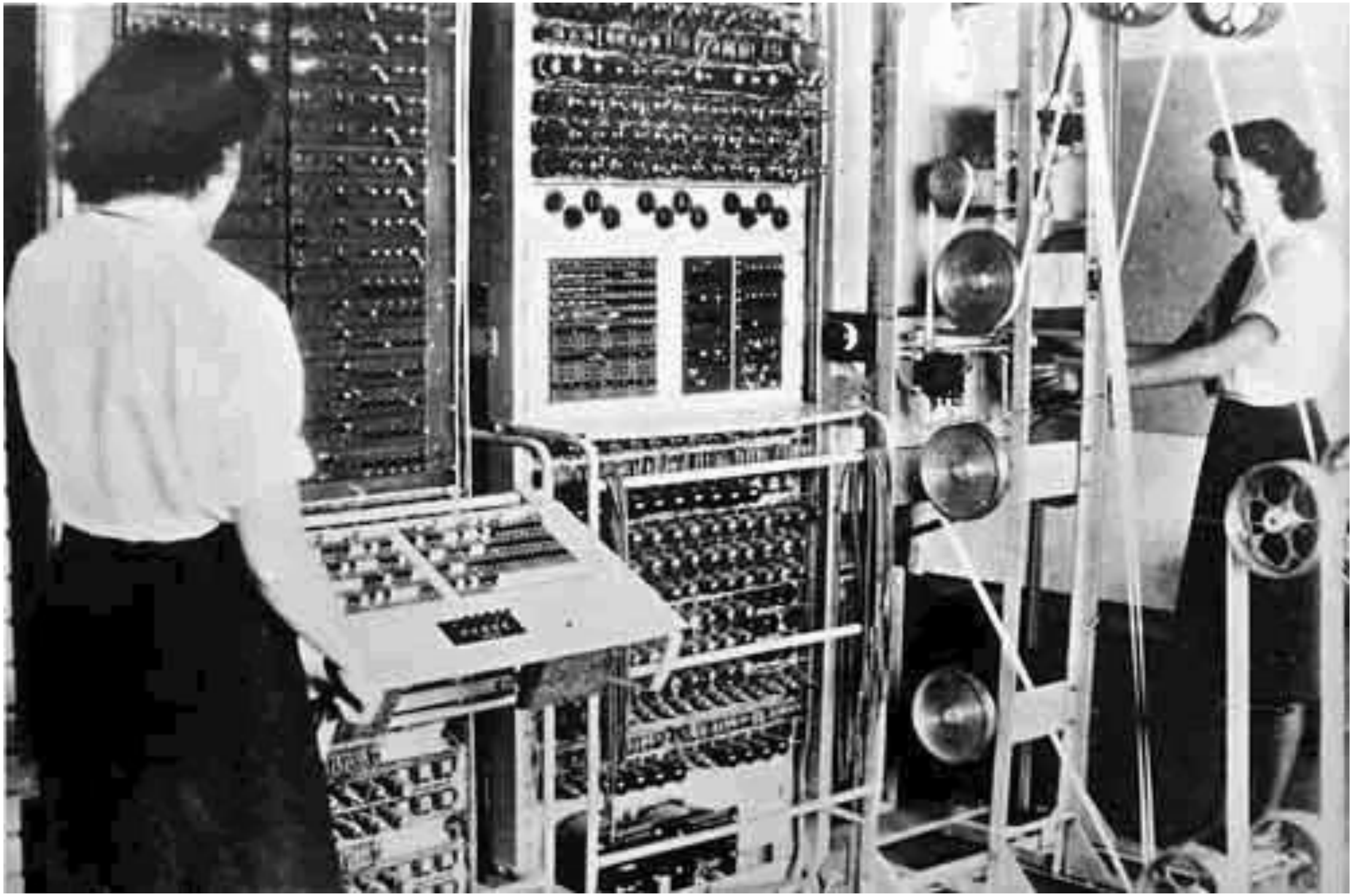
History of Operating Systems

The First Computers

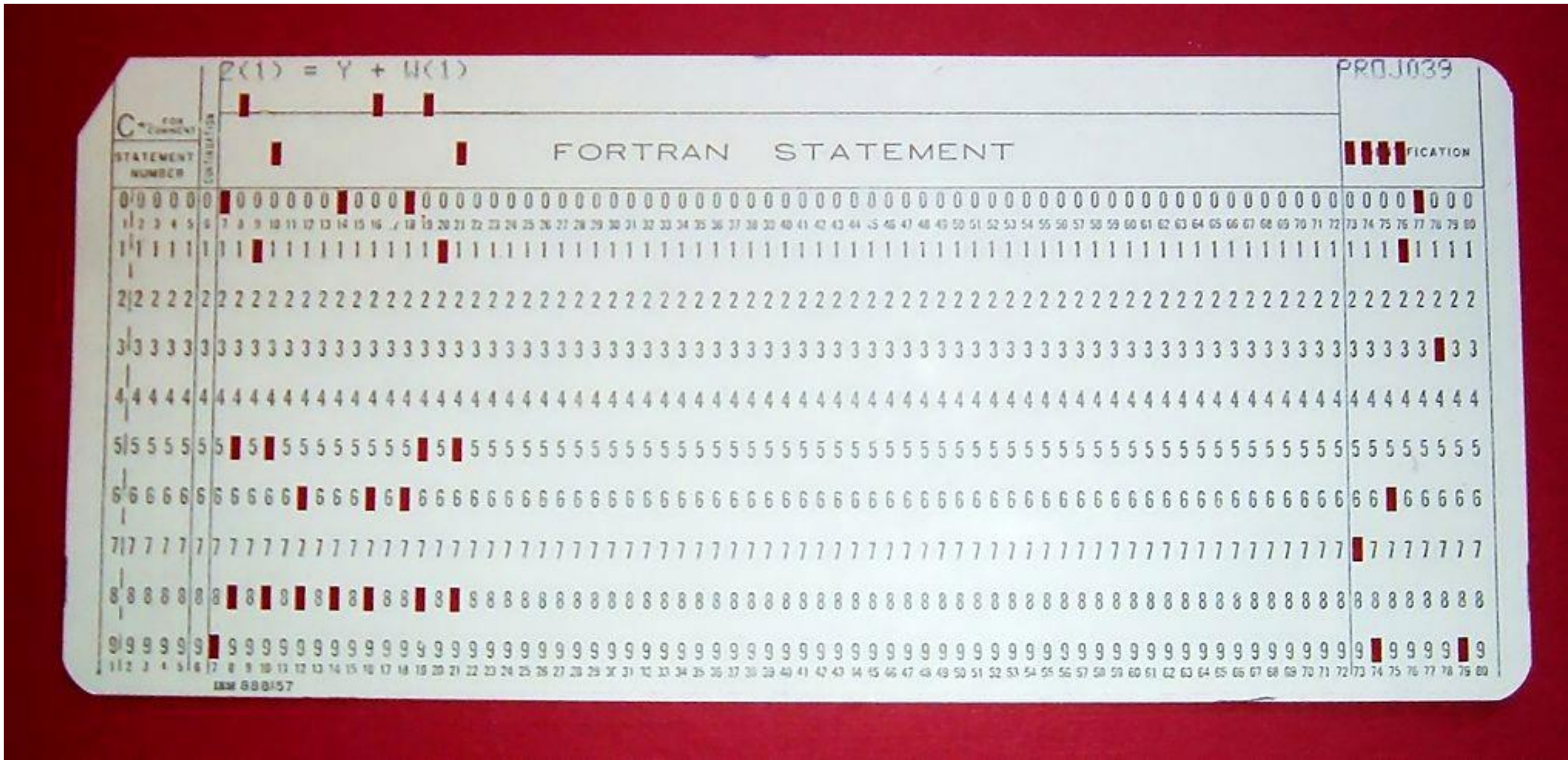
- Early machines (1940s to mid-1950s) had no Operating System

- The user interacted directly with the hardware
  - initial interfaces: console of switches (input) & lights (output)
  - later interfaces: punched cards, printers, etc.

Colussus



Punch Card


Z(1) = Y + W(1)
PROJ039
FORTRAN STATEMENT
STATEMENT NUMBER
FICATION

Issues With First Computers

- Long setup time for a program to run
- Users accessed the system one at a time
- scheduling (what program is run next?) made by hand
- no sharing of libraries, drivers, . . .

Mainframes: Batch Systems

- The earliest Operating Systems were used in mainframes (1950s)
- These OSs were batch systems, which attempted to
  - eliminate the manual set-up of programs to be run
  - provide reusable code to access hardware (i.e. drivers)

Mainframes: Batch Systems

- Operating System was stored in main memory (was called a monitor)
- One job (program) loaded at a time from a punched card/tape reader into remaining memory
- Job control instructions told the Operating System what to do

Mainframes: Batch Systems

- These simple OSs were code to which one linked one's program (loaded as a whole into main memory) to be run

- basically, the OS was just a run-time library

Issues with Mainframes

- Input/output (I/O) operations were very slow
- No computations were done while performing I/O
- This decreased CPU usage

IBM 701

Mainframes: Multiprogramming

- Idea: expand memory to hold two or more programs and switch among all of them (multitasking or multiprogramming)
- Multiprogramming systems were rendered possible by the first integrated circuits (IC) in the early 1960s

Mainframes: Multiprogramming

- Multiple runnable jobs loaded in memory at the same time
- Overlap I/O operations of a job with the computations of another
- benefit from I/O devices that can operate asynchronously (interrupts and direct memory access —DMA)

Mainframes: Multiprogramming

- increase the processor utilisation and attempt to optimise throughput (i.e., jobs completed per unit time)
- degree of multiprogramming: number of jobs that can be managed at once by the OS
- Multiprogramming (aka multitasking) is the central theme of modern OSs

Mainframes: Multiprogramming

- Multiple runnable jobs loaded in memory at the same time
- Overlap I/O operations of a job with the computations of another
- benefit from I/O devices that can operate asynchronously (interrupts and direct memory access —DMA)

Mainframes: Timesharing

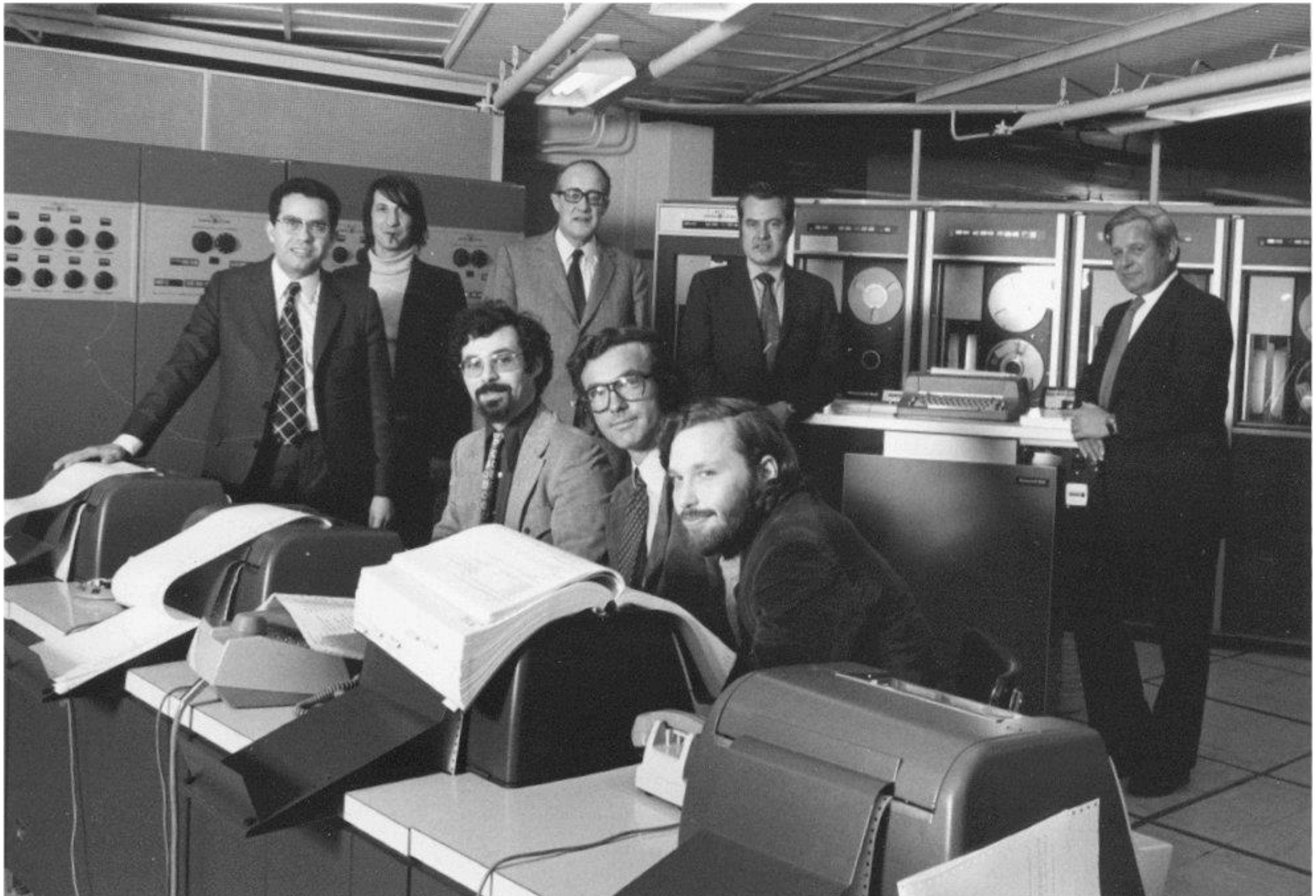
- Initially multiprogramming was still batch-based
  - turnaround time could be long for any particular job
  - no interactivity

- Idea: to have multiple users simultaneously using terminals, with the OS interleaving the execution of each user program in short quanta of computation

Timesharing systems

- based on time slicing (a.k.a. time multiplexing)
- each user feels like using the shared computer on his/her own
- challenge: to optimise response time
- it allows the users to view, edit, debug, and run their programs interactively

GE645 running Multics



Desktop Operating Systems (1980s)

- Very Large Scale of Integration (VLSI) circuits made it cheaper to manufacture complex hardware
- Hardware became cheaper
- Easier to have one computer per user than share mainframe

Desktop Operating Systems (1980s)

- usability facilitated by the introduction of graphical user interfaces (GUI)

- Idea: to maximize user convenience and responsiveness (apart from CPU & I/O use, such as in multiprogrammed & timesharing systems)

Parallel Operating System

- Idea: to run and manage parallel applications efficiently on tightly coupled parallel computers (multiprocessors)
- gives support for parallel applications composed of several time-consuming but separable subtasks

Parallel Operating System

- Provides primitives for assigning (scheduling) parallel subtasks to different processors
- Provides primitives for dividing a task into parallel subtasks, if possible
- Supports efficient communication between parallel activities
- Supports synchronisation of activities to coordinate data sharing

Distributed Operating System

- Idea: a common operating system shared by a network of loosely coupled independent computers
- Facilitates the sharing of resources located in different places (hardware and software)
- looks to its users like an ordinary centralised operating system

Distributed Operating System

- Supports communication between parts of a job, or between different jobs, across the network

- it allows for some parallelism, but speed is not the main goal

Real-Time Operating System

- Idea: to guarantee a response to physical events in a fixed interval of time
- used for specialised applications: subway systems, flight control, factories, power stations, etc.
- all activities scheduled in order to meet critical requirements

Real-Time Operating System

- performs operations within predetermined timeframes

- Soft real-time: implemented by all OSs in modern PCs to run multimedia applications

Next Week

- Topic:
  - Operating System Structure
- Study time:
  - Review Chapters 1 and 2 in book.