Concurrent Systems

3D4 ⟷ CS2016

Operating Systems

Andrew Butterfield
ORI.G39, Andrew.Butterfield@scss.tcd.ie

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

with thanks to Mike Brady

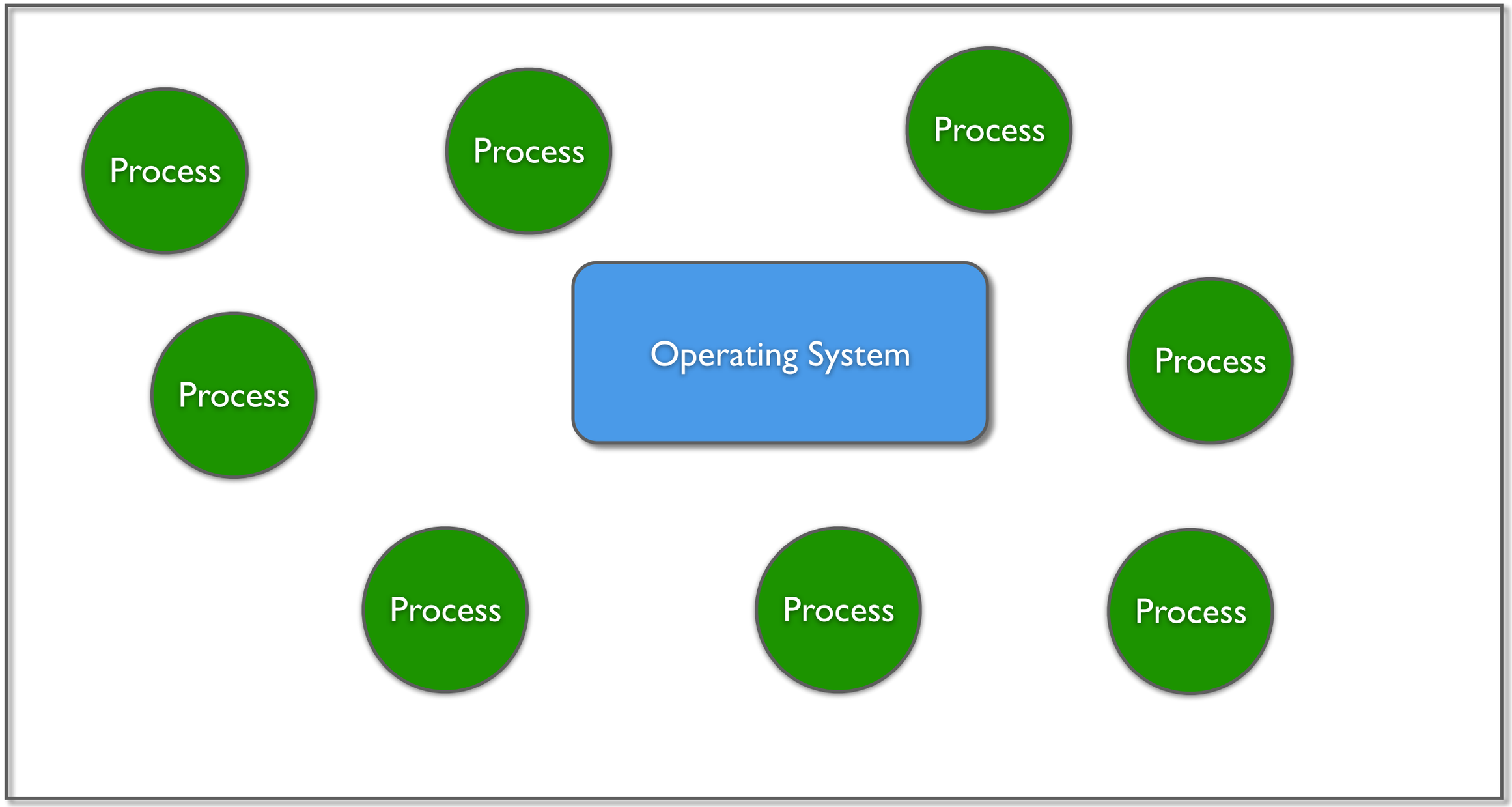
Two Views

- A Resource Manager
  - The OS is responsible for the fair and efficient management of the resources of the computer system
    - CPU (Scheduling, etc.)
    - Memory (Memory Management, Virtual Memory, Paging, etc.)
    - Devices (Disks, Special-purpose Devices)
    - Files & Directories
    - Network Connections (Ports)

Resource Manager view of the OS

- The OS manages:
  - Processors – scheduling of processes/tasks and threads
  - Memory – physical & virtual memory
  - Devices – device access and device drivers
  - Files and Directories
  - Communication with other processes – ports

A View of a Computer with an OS



Processes (aka Tasks)

- Processes are the 'entities' that run as self-contained programs in the computer, under the complete control of the OS
- Processes, also known as tasks, are given ownership of resources, such as memory, certain devices, etc. by the OS.

Multitasking or Time-Sharing

- Usually there are many processes but only one or a few processors. The OS gives processes the use of a processor from time to time – it shares some processor time with the processes.
- All eligible processes should get a 'slice' of time fairly frequently, say 10 ms every 200 ms, depending (a lot!) on what's going on.
- This is called multitasking or time-sharing.
- If the processors and the OS scheduler are fast enough, the illusion is that all processes are running simultaneously.

Eligible Processes?

- A process could be ready to execute if it had a processor.
- It could be blocked if it's waiting for some event to happen.
- Lots of contending processes could have different priorities, e.g.
  - A video-playing process needs processor time urgently from time to time -- high priority.
  - A very long scientific calculation might just need to 'soak up' any spare processor time that happened to be available -- low priority.
- So, there could be eligible/ineligible processes with various levels of priority.

Types of Multitasking

- The OS can grant a processor to a ready process easily, say using a kind of return-from-exception sequence.
- To get the processor back from the process
  - If the process must relinquish the processor, this is called co-operative multitasking.
  - If the processor is removed from the process by the OS (typically by a timer interrupt exception handler), then this is pre-emptive multitasking.

Co-operative Multitasking

- Advantages
  - Lower overhead: processes can be prepared for yielding the processor, since the yield calls are known.
  - No danger that the processor will be given up at a time-critical point.
- Disadvantages
  - Can be hard to program yield calls into a program, if done badly, a program can ‘hog’ a processor and never give it up
- Overall
  - Lower overhead – potentially useful where the various processes can be relied on to be well written.
  - Otherwise, inadvisable, e.g. in the older Mac OS, Windows up to Windows NT.

Preemptive Multitasking

- Advantages
  - More robust against badly-written or malicious processes.
  - Easier to write applications for – programmers don't have to do anything special, as preemption is handled transparently.
- Disadvantages
  - May be expensive, as full contexts must be stored and restored.
  - May cause problems if a process is preempted at an inopportune time.