CS420: Operating Systems

Operating System Structure

James Moscola
Department of Physical Sciences
York College of Pennsylvania

Operating System Design and Implementation

- **Design and Implementation of OS not "solvable"**
  - There is no single OS that works best for every use case
- **How does one design an operating system??**
  - Start by defining goals and specifications
  - Affected by choice of hardware, type of system
  - User goals and System goals
    - User goals – operating system should be convenient to use, easy to learn, reliable, safe, and fast
    - System goals – operating system should be easy to design, implement, and maintain, as well as flexible, reliable, error-free, and efficient

An OS With Simple Structure

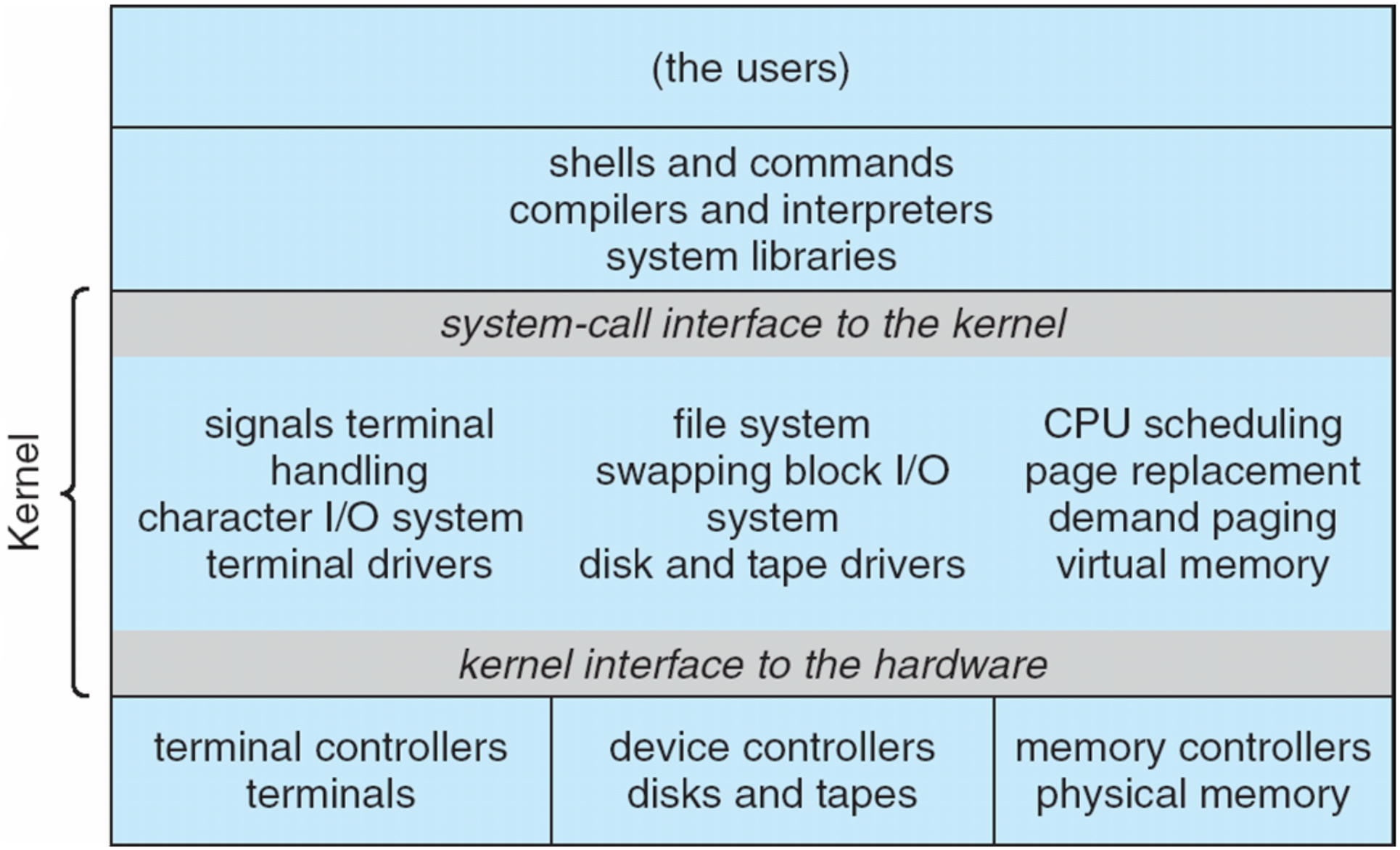
- **MS-DOS – written to provide the most functionality in the least space**
  - Not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated
  - Not well protected

Layered Approach

- **The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.**

- **With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers**
  - An OS can be designed from the lower layers up; ensure that lower layers work before moving to higher layers

Traditional UNIX System Structure



Microkernel System Structure

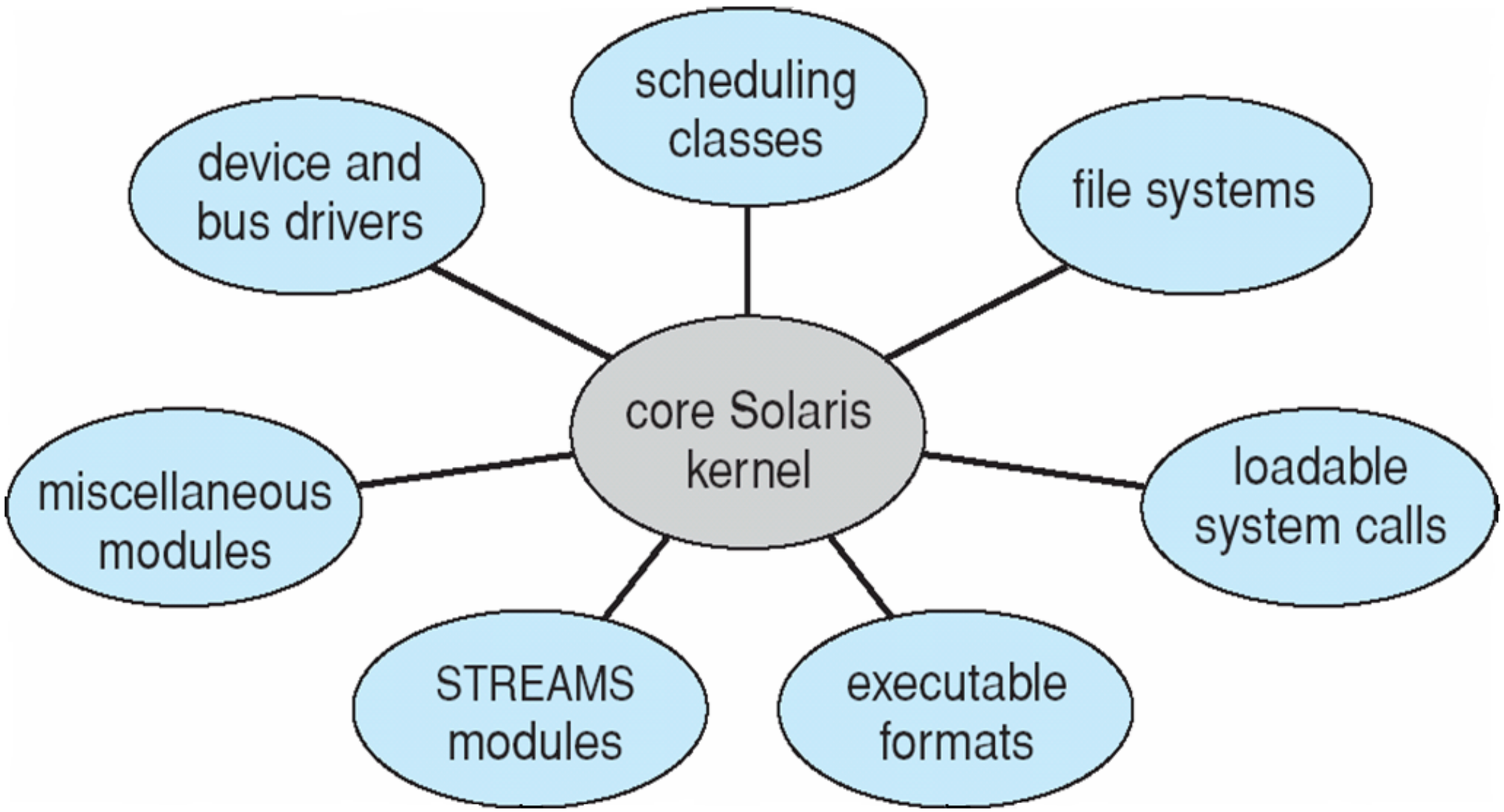
- **Moves as much from the kernel into “user” space**
- **Communication takes place between user modules using message passing**
- **Benefits:**
  - Easier to extend a microkernel
  - Easier to port the operating system to new architectures
  - More reliable (less code is running in kernel mode)
  - More secure
- **Detriments:**
  - Performance overhead of user space to kernel space communication

Kernel Modules

- **Most modern operating systems implement kernel modules**
  - Uses object-oriented approach
  - Each core component is separate
  - Each talks to the others over known interfaces
  - Each is loadable as needed within the kernel

- **Overall, similar to layers but more flexible**

Solaris Modular Approach



Virtual Machines

- **A virtual machine takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware.**

- **A virtual machine provides an interface identical to the underlying bare hardware.**

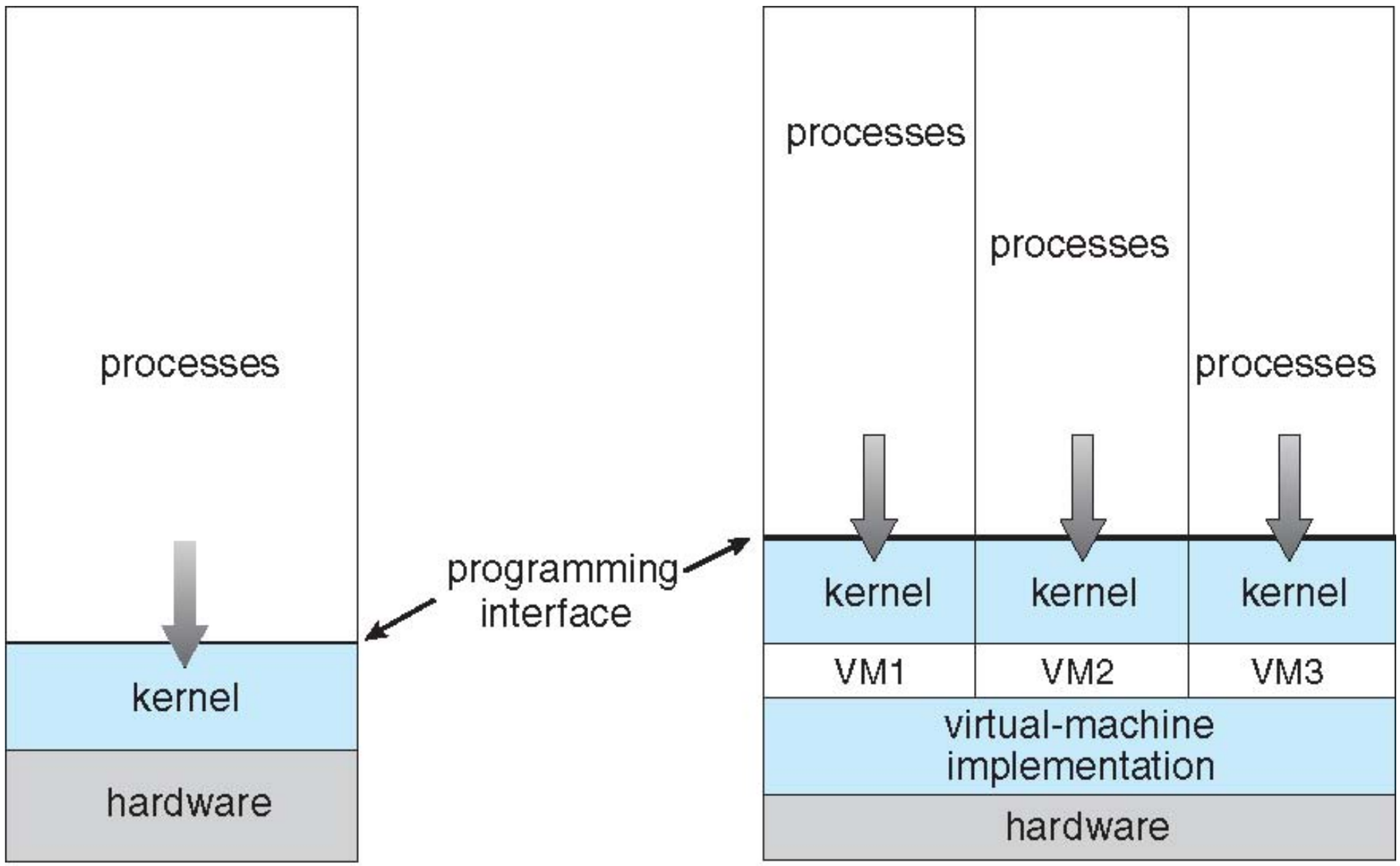
- **The operating system host creates the illusion that a process has its own processor and (virtual memory).**

- **Each guest provided with a (virtual) copy of underlying computer.**

Virtual Machines History and Benefits

- **First appeared commercially in IBM mainframes in 1972**
- **Fundamentally, multiple execution environments (different operating systems) can share the same hardware**
- **Protected from each other**
- **Some sharing of file can be permitted, but controlled**
- **Communicate with each other, other physical systems via networking**
- **Useful for development, testing**
- **Consolidation of many low-resource use systems onto fewer busier systems**
- **“Open Virtual Machine Format”, standard format of virtual machines, allows a VM to run within many different virtual machine (host) platforms**

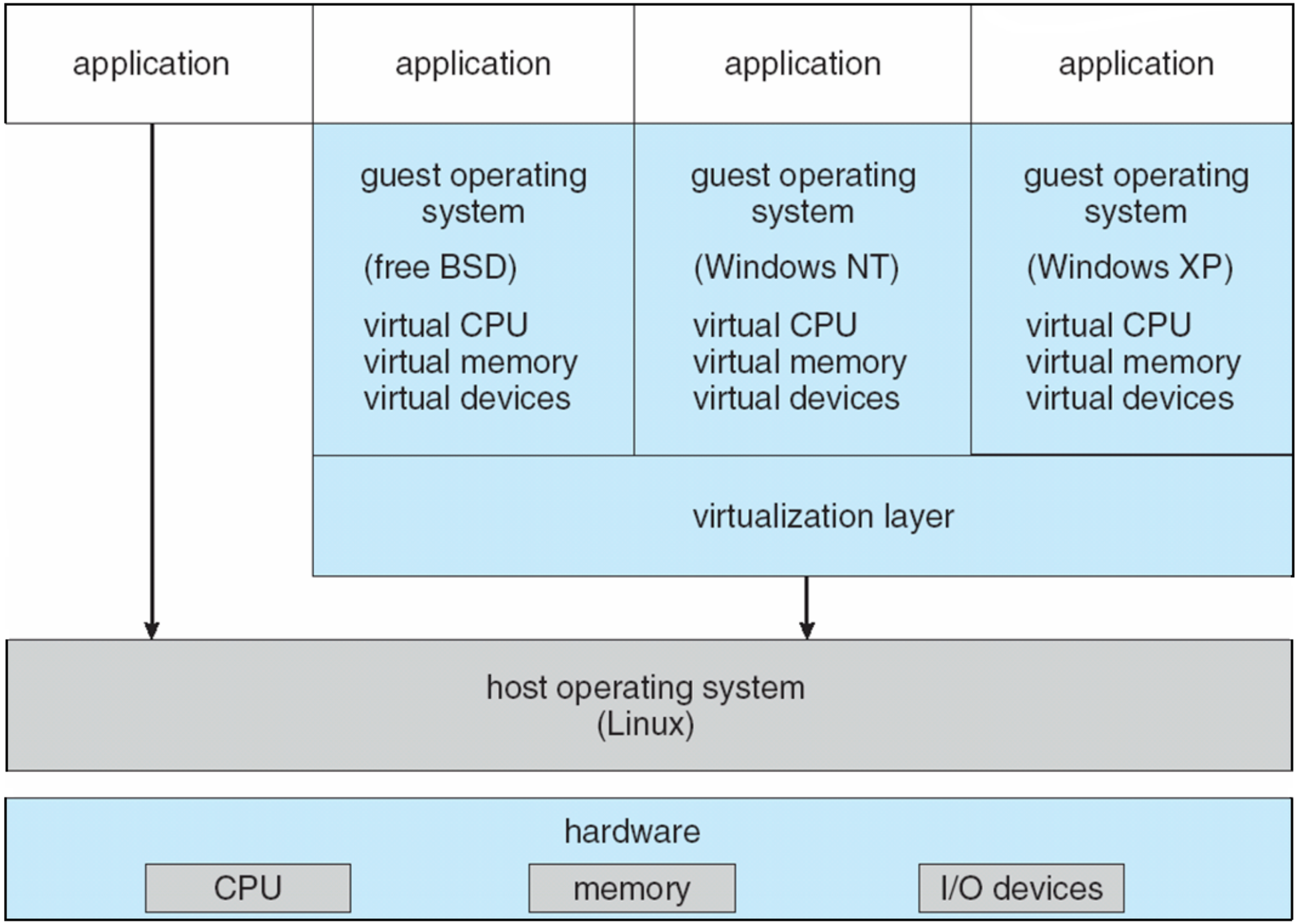
Virtual Machines (Cont.)



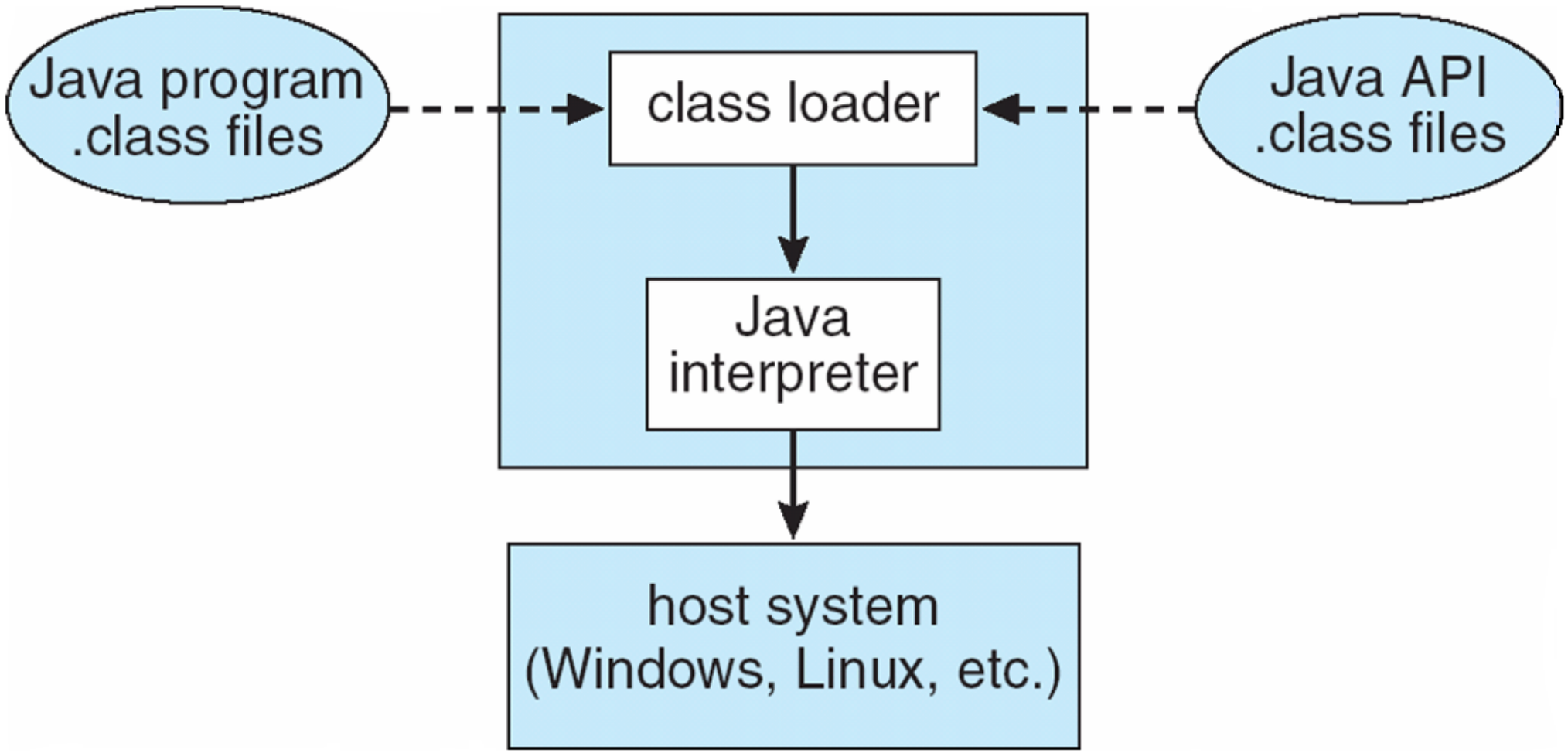
(a) (b)

Nonvirtual machine **virtual machine**

VMware Architecture



The Java Virtual Machine



System Boot

- **Operating system must be made available to hardware so hardware can start it**
  - Small piece of code – bootstrap loader, locates the kernel, loads it into memory, and starts it
  - Sometimes two-step process where boot block at fixed location loads bootstrap loader
  - When power initialized on system, execution starts at a fixed memory location
    - Firmware used to hold initial boot code