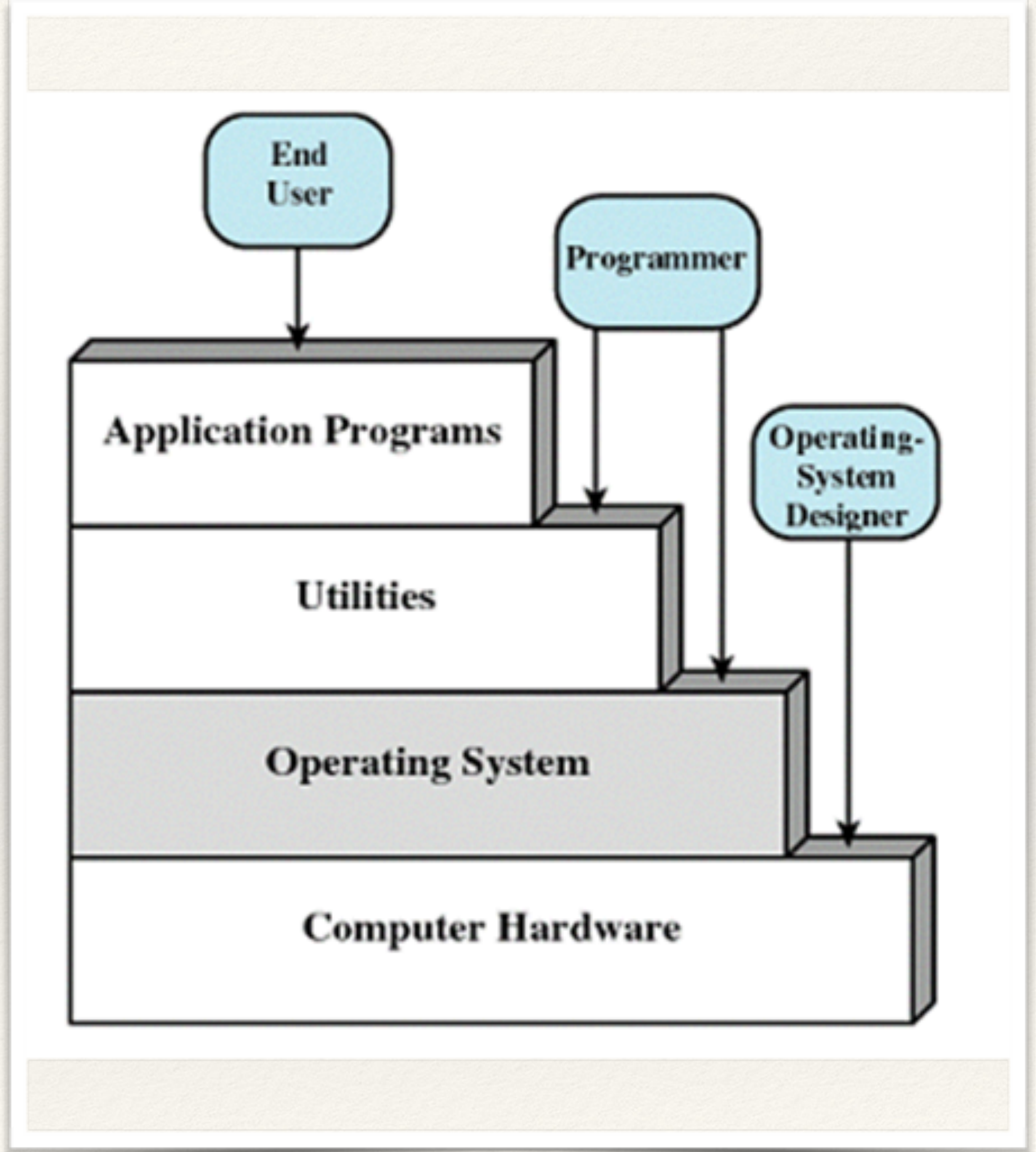
Circles within circles…

Operating System Structures

- Layered Structure
- Multi-kernels for Multi-core processors
- Microkernel Structure
- Virtual Machine Structure

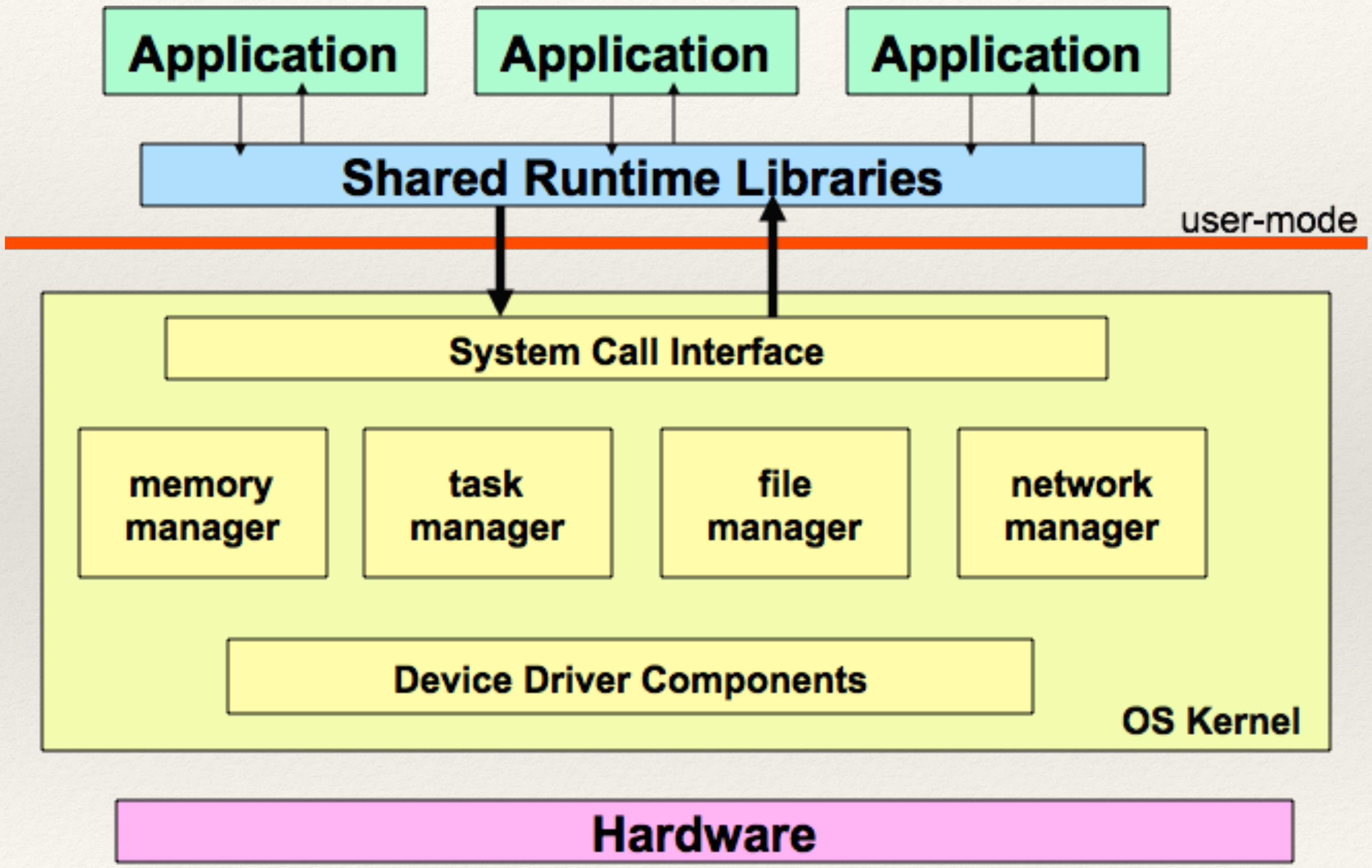
Layered System Structure

- System structured as a series of levels - each level performs a related subset of functions.
- Each level relies on the next lower level for more primitive functions.
- Layer Concepts
  - Kernel
  - Basic services and I/O drivers
  - Resource management



http://backup-ilmu.blogspot.ca/

A Modern OS Design



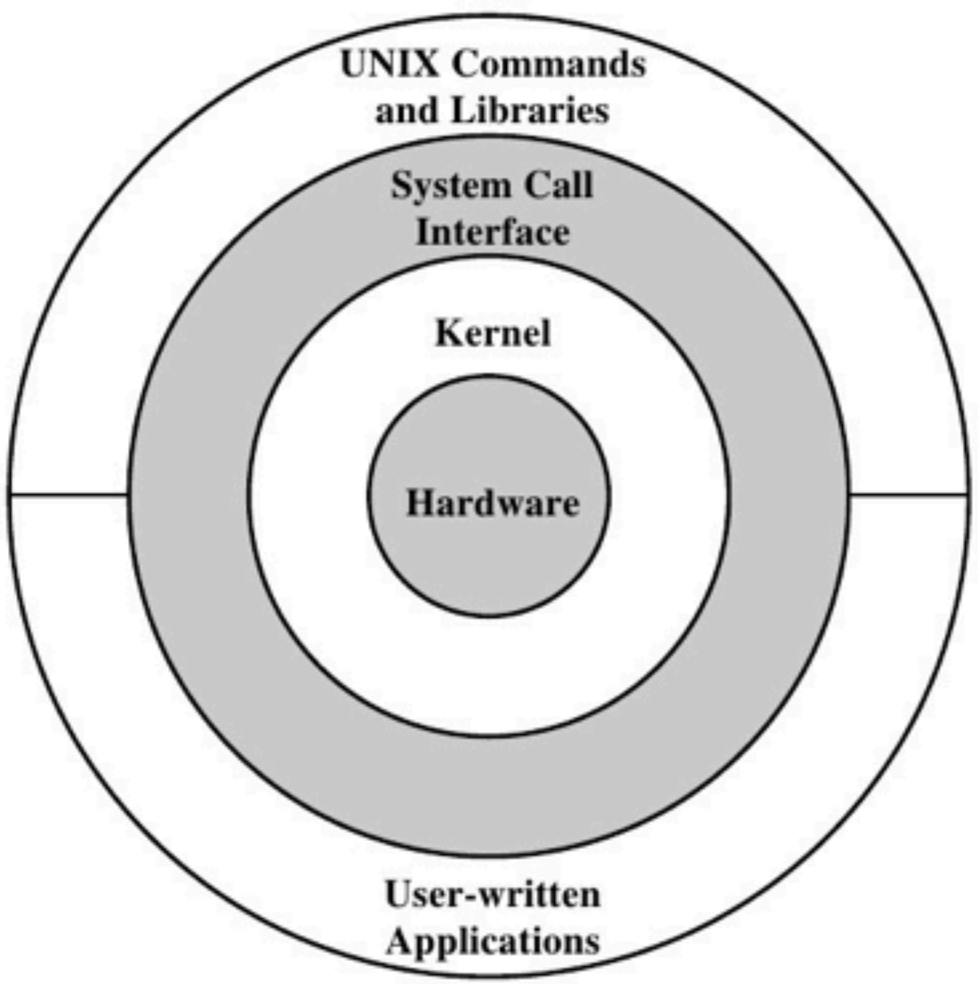
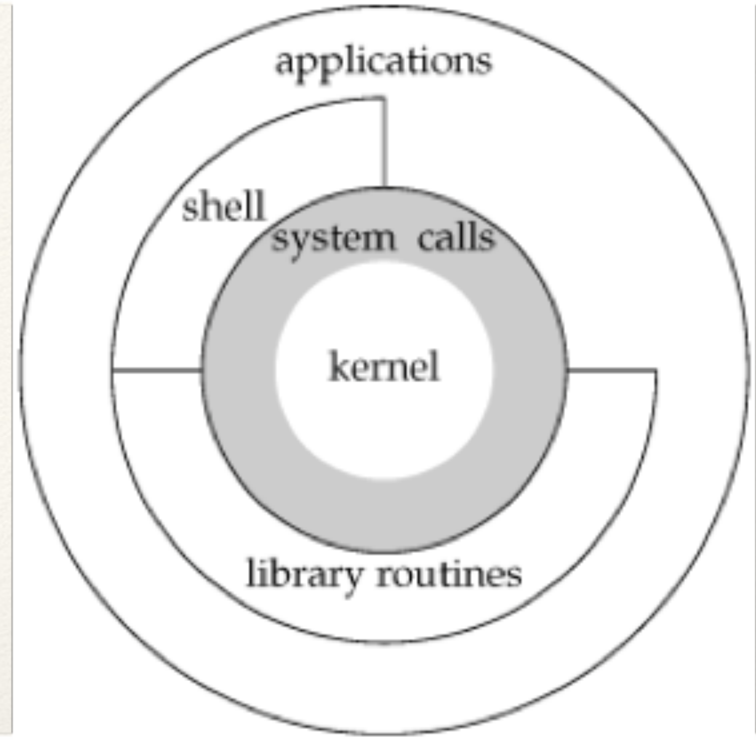


Figure 2.15 General UNIX Architecture

http://www.slideshare.net/thanhlinh9191/unix-william-stalling



http://poincare.matf.bg.ac.rs/~ivana/courses/tos/sistemi_knjige/pomocno/apue/APUE/0201433079/ch01lev1sec2.html

UNIX System Structure

(the users)		
shells and commands compilers and interpreters system libraries		
system-call interface to the kernel		
signals terminal handling character I/O system terminal drivers	file system swapping block I/O system disk and tape drivers	CPU scheduling page replacement demand paging virtual memory
kernel interface to the hardware		
terminal controllers terminals	device controllers disks and tapes	memory controllers physical memory

http://www.slideshare.net/smileybemust/operating-system-structure

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.

Multi-Processors

- Typically each processor does self-scheduling from the pool of available process or threads.
  - Timer interrupt
  - Ready queue
- OS Support
  - Any thread (including kernel threads) can run on any processor.
  - **Soft affinity** (close relation) – Try to reschedule a thread on the same processor.
  - **Hard affinity** – Restrict a thread to certain processors.

Multi-Core Processors

- In a multi-core environment, the control over which core to run a specific thread or application on is essential.
- Without this control, the threads/applications may get assigned to the wrong processors or cores and cause unnecessary performance degradation.
- Generally each processor has its own cache, shared cache, shared memory and I/O

Multi-Core OS Design Issues

- **Single kernel or Multi-kernel**
  - Kernel routines must be reentrant to allow multiple treads to execute them
- **Scheduling**
  - Must avoid conflicts
  - May be able to run processes or threads concurrently
- **Synchronization**
  - Mutual exclusion, event ordering
- **Memory Management**
  - Deal with multi-port memory
  - Have a unified paging scheme
- **Reliability and Fault Tolerance**

Microkernel Structure

- Popularized by use in Mach OS
- **Monitors** (1960s):
  - Built as a single large program, any routine can call any other routine
  - Used in most early systems
- **Layered OS** (1970s – now):
  - Based on modular programming
  - Major changes still had wide-spread effects on other layers
- **Microkernel** (1980s – now):
  - Only essential functions in the kernel
  - File System, device drivers, *etc.*, are now external subsystems/processes
  - Processes interact through (IPC) messages passed through the kernel

Microkernel

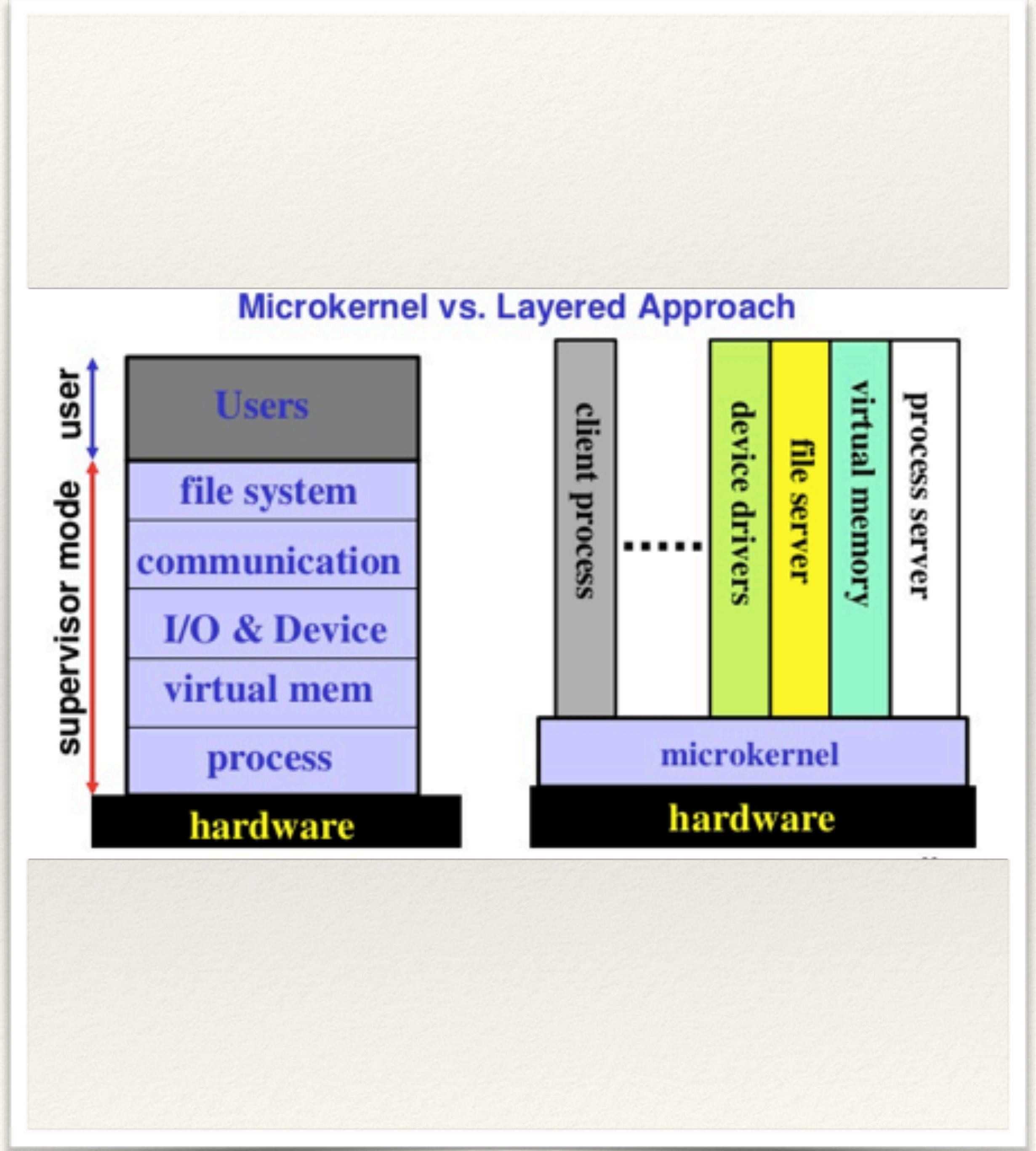
- Identify and isolate a small operating system core that contains only essential OS functions.
- Move many services included in the traditional kernel OS to external subsystems
  - device drivers
  - file systems
  - virtual memory manager
  - windowing system and security services

Microkernel Design

- **Primitive Memory Management**
  - Kernel handles virtual → physical mapping, rest is a user mode process
    - VM module can decide what pages to move to/from disk
    - Module can allocate memory
  - Three microkernel memory operations
    - *Grant* – Grant pages to someone
    - *Map* – Map pages in another space
    - *Flush* – Reclaim pages granted or mapped
- **Inter-process Communication (IPC)**
  - Based on messages
- **I/O and Interrupts**
  - Handle interrupts as messages

Microkernel Benefits

- **Uniform Interface**
  - Same message for user/system services
- **Extensibility**
  - Easy to add new services
  - Modifications need only change directly affected components
  - Could have multiple file servers
- **Flexibility**
  - Can customize system by omitting services
- **Portability**
  - Isolate nearly all processor-specific code in the kernel



http://www.slideshare.net/mohammedarif89/os-structure

Microkernel Benefits

- **Reliability**
  - Easy to test kernel
  - Fewer system calls to master
  - Less interaction with other components
- **Distributed System Support**
  - Just as easy to send a message to another machine as this machine
    - Need system-wide unique Ids
  - Processes do not have to know where a service resides
- **Object-Orientated OS**
  - Lends OO disciplines to the kernel

Kernel Performance

- Sending a message generally slower than simple kernel call (soft interruption)
- Depends on size of the microkernel
  - First generation systems slower
  - Then tried to include critical system items into kernel (Mach)
    - Fewer user/system mode switches
    - Lose some microkernel benefits
  - Trying approach of very small kernel
    - Mach L4 kernel (version 2) - 12K code, 7 system calls. Speed seems to match Unix
    - Mac OS X adopt this concept

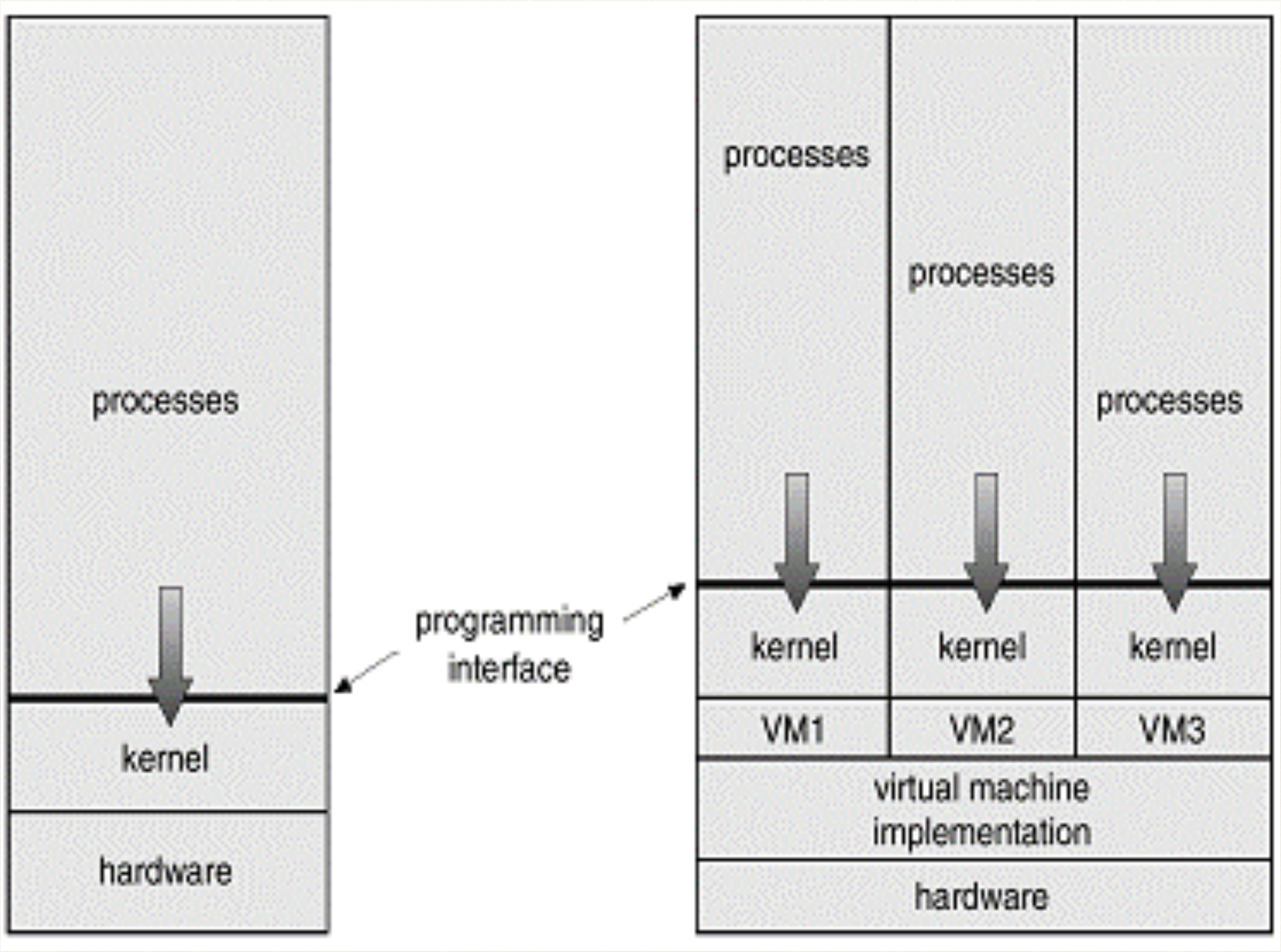
Virtual Machine Structure

- 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 creates the illusion of multiple processes, each executing on its own processor with its own (virtual) memory.

Virtual Machines

- The resources of the physical computer are shared to create the virtual machines.
  - CPU scheduling can create the appearance that users have their own processor.
  - Spooling and a file system can provide virtual hard drivers and virtual printers.
  - A normal user display device serves as the virtual machine operator's console.

System Models



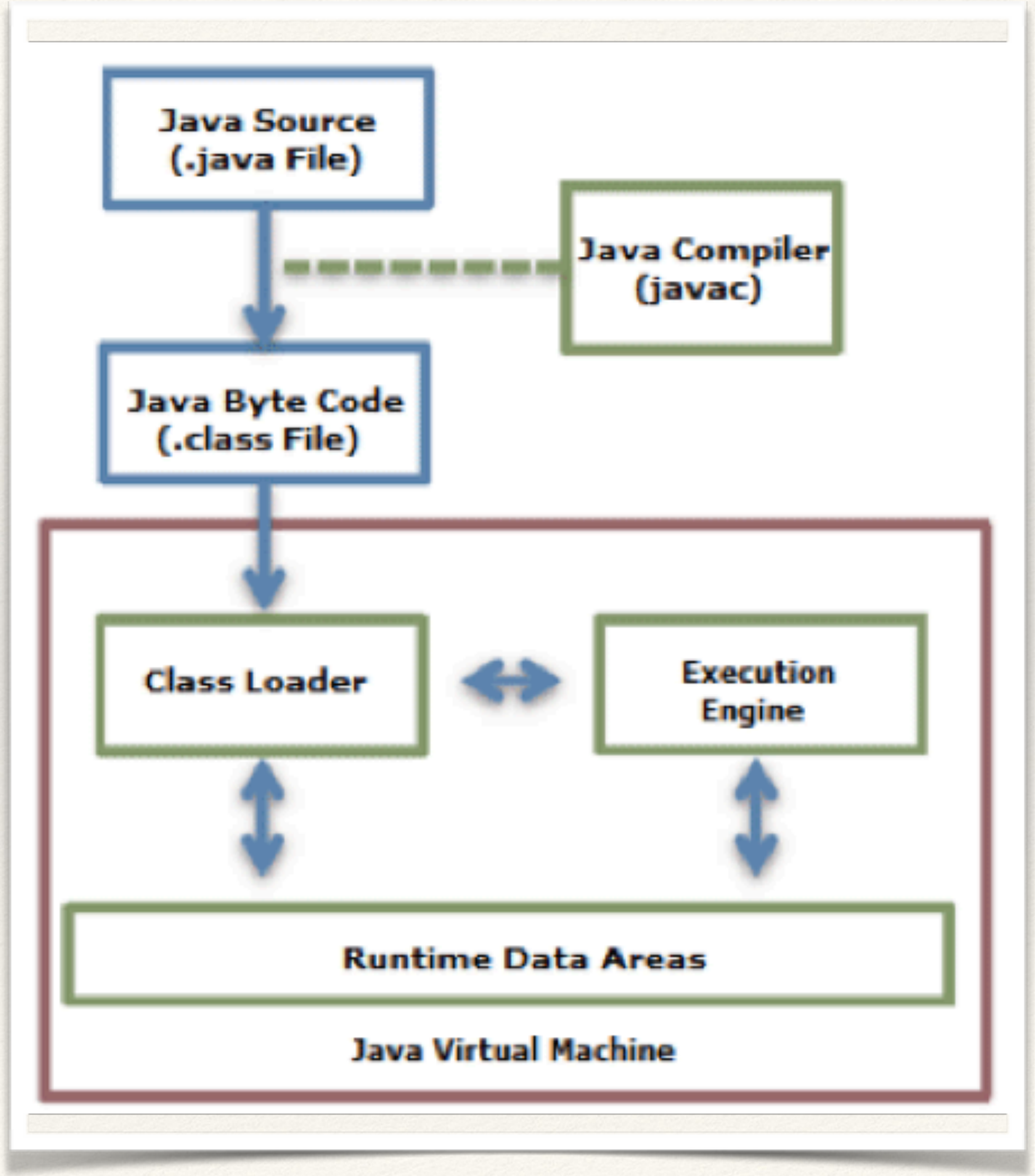
http://www.gitam.edu/eresource/comp/gvr(os)/3.4.htm

Advantages/Disadvantages of VM's

- The virtual-machine concept provides complete protection of system resources since each virtual machine is isolated from all other virtual machines. This isolation, however, permits no direct sharing of resources.
- A virtual-machine system is a perfect vehicle for operating systems research and development.
- System development is done on the virtual machine, instead of on a physical machine, and so does not disrupt normal system operation.
- The virtual machine concept is difficult to implement due to the effort required to provide an exact duplicate to the underlying machine.

Java Virtual Machine

- Compiled Java programs are platform-neutral bytecodes executed by a Java Virtual Machine (JVM).
- JVM consists of
  - class loader
  - class verifier
  - runtime interpreter
- Just-In-Time (JIT) compilers increase performance.



http://www.cubrid.org/blog/dev-platform/understanding-jvm-internals/

System Design 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.

Mechanisms and Policies

- **Mechanisms** determine how to do something, **policies** decide what will be done.
- The separation of policy from mechanism is a very important principle, it allows maximum *flexibility* if policy decisions are to be changed later.

System Implementation

- Traditionally written in assembly language, operating systems can now be written in higher-level languages.
- Code written in a high-level language:
  - can be written faster
  - is more compact
  - is easier to understand and debug
- An operating system is far easier to port (move to some other hardware) if it is written in a high-level language.