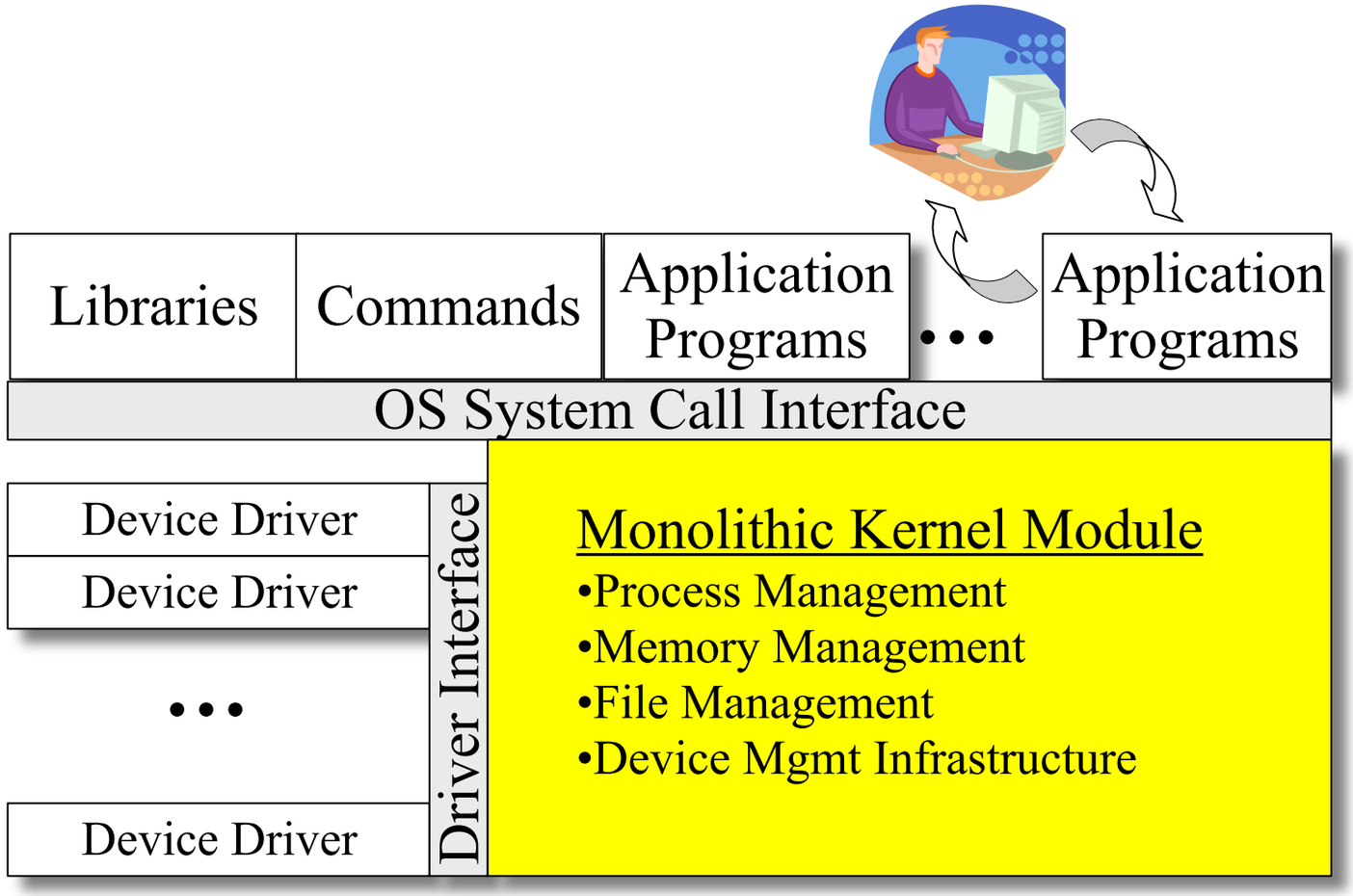
Microkernels

John Criswell
University of Rochester

Onwards to user-space!

Microkernels

Monolithic Kernel (aka Everything and the Kitchen Sink)



Monolithic Kernel Limitations

- Poor security
  - Buffer overflow gains access to ***<u>everything!</u>***
- Poor reliability
  - Bug in kernel can affect unrelated subsystems
  - Difficult to restart faulty subsystem

Processes Don't Have This Problem

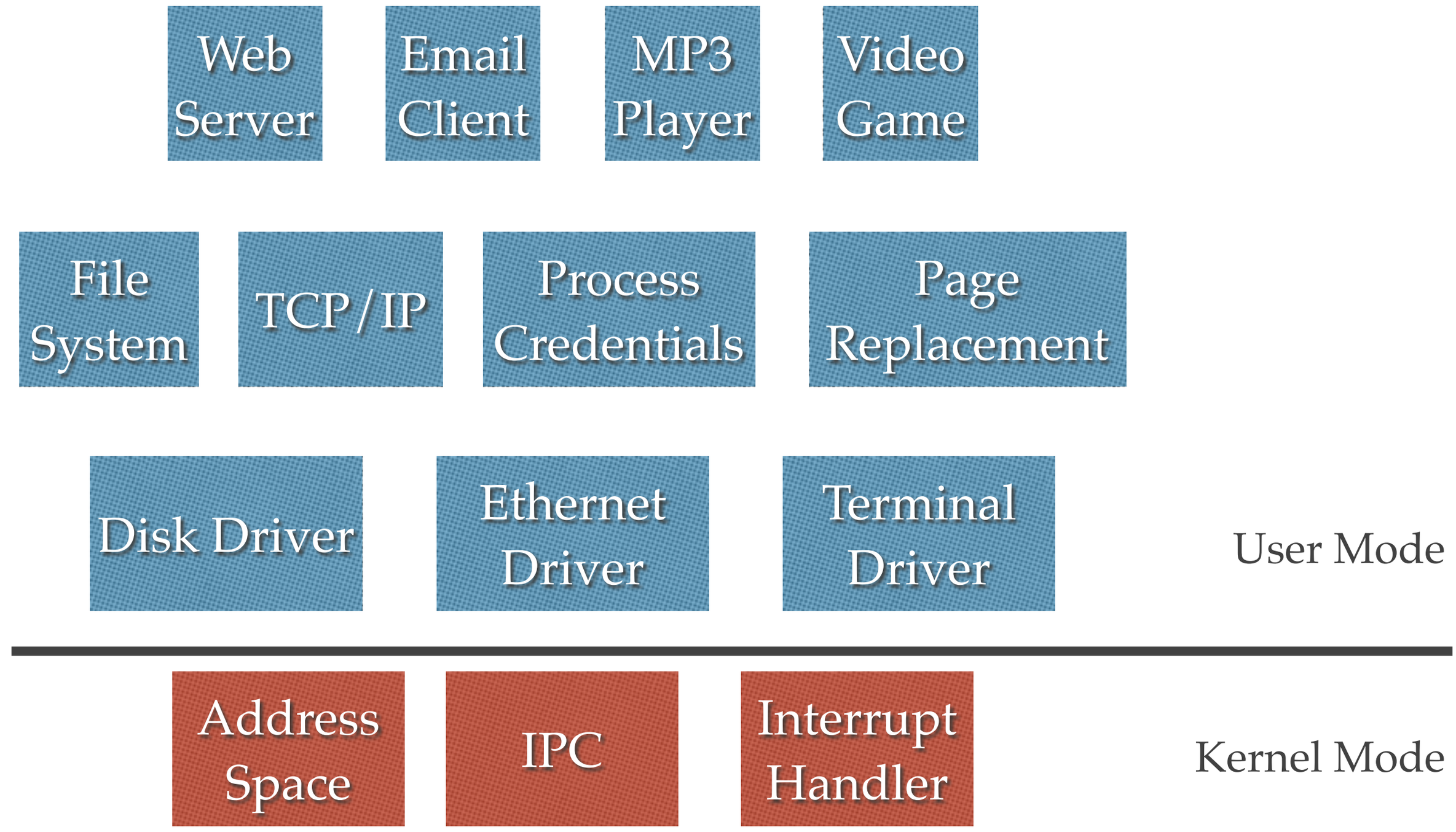
- Isolated memory
- Communication via
  - Pipes
  - Explicitly shared memory
- Self-contained programs
  - No access to irrelevant data structures

Could kernel components be processes?

Microkernel

- Move kernel functionality into user-space processes
  - File systems
  - Networking subsystem
    - Drivers
- Kernel provides
  - Protection
  - Communication mechanisms

Microkernel



Advantages of Microkernels

- Faults are localized
  - Bug in network code doesn't corrupt disk data
- Easier to improve reliability
  - Can monitor and restart processes (e.g., filesystem)
- Easier to apply security techniques
  - Randomization and re-randomization (Guiffruida)
  - Apply memory safety or type-safe language to critical processes

Disadvantages of Microkernels

- Communication overhead
  - Semantics of message passing affects performance
  - What is placed in user-space affects performance
- User/Kernel boundary crossing overhead
- Context switching overhead
  - Monolithic libraries are always available
  - User-space service may not have CPU when needed
  - TLB Flush when switching page table pages

Microkernel Advantages are Not Magic

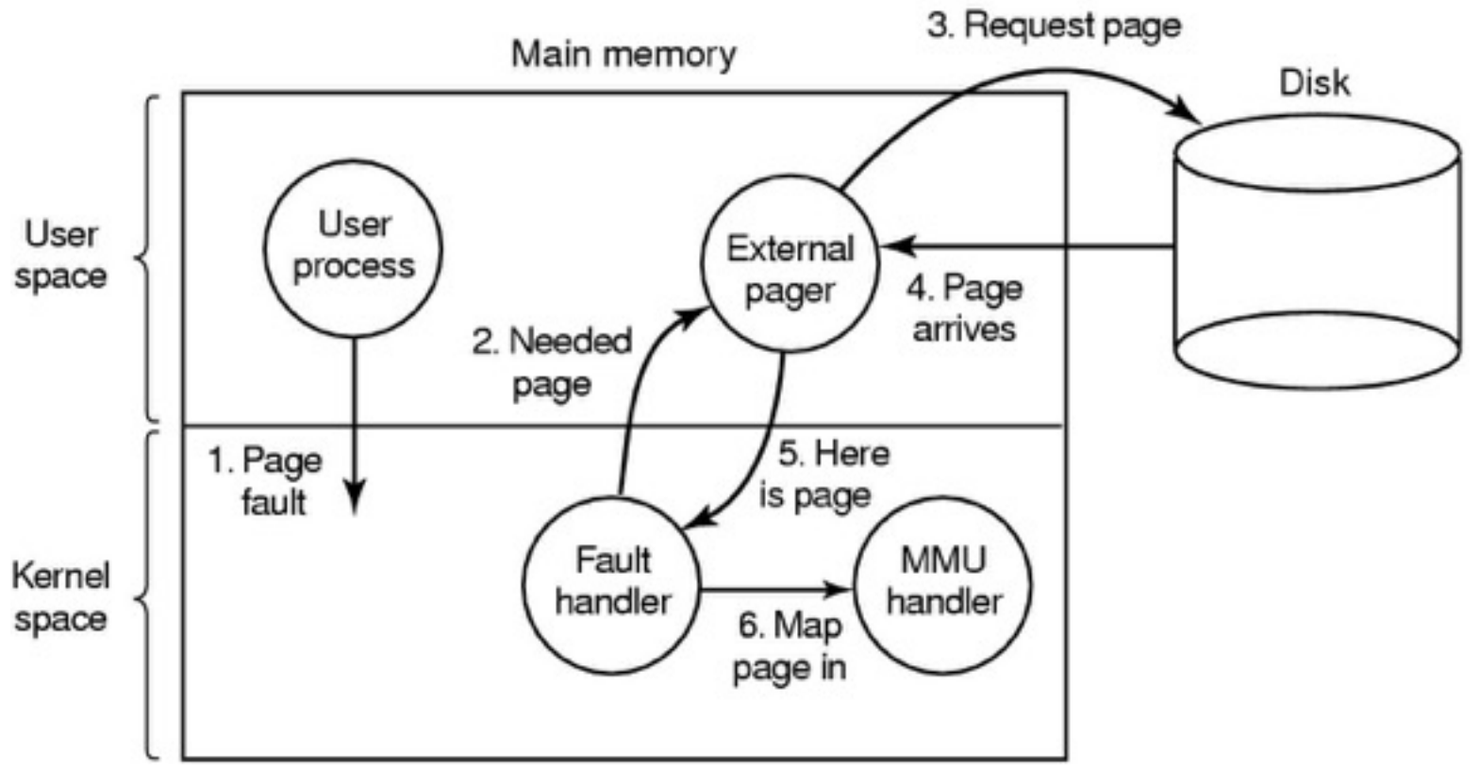
- Reliability must be designed and built
  - File system process crash still catastrophic
    - Restart of critical processes must be designed and built
- Security is still an issue
  - Exploited file system process can access any file
  - Exploited network process can read all packets

Mach

- Developed at Carnegie Mellon University in the 80's
- Memory management design influenced modern OS design
- Goal: separate policy from mechanism

Example: Mach

- User-level memory management
  - trusted/protected by the kernel
  - kernel provides the basic protection mechanism
  - user-level memory manager handles page loading; decides replacement policy



Microkernel Failures

- Windows NT family
  - Original Windows NT had microkernel design
  - By Windows 2000, functionality moved into kernel
- Mac OS X
  - Based on NextStep which is based on Mach + 4.4BSD
  - BSD sub-systems moved into kernel; live alongside Mach
  - Essentially two kernels living in the same space

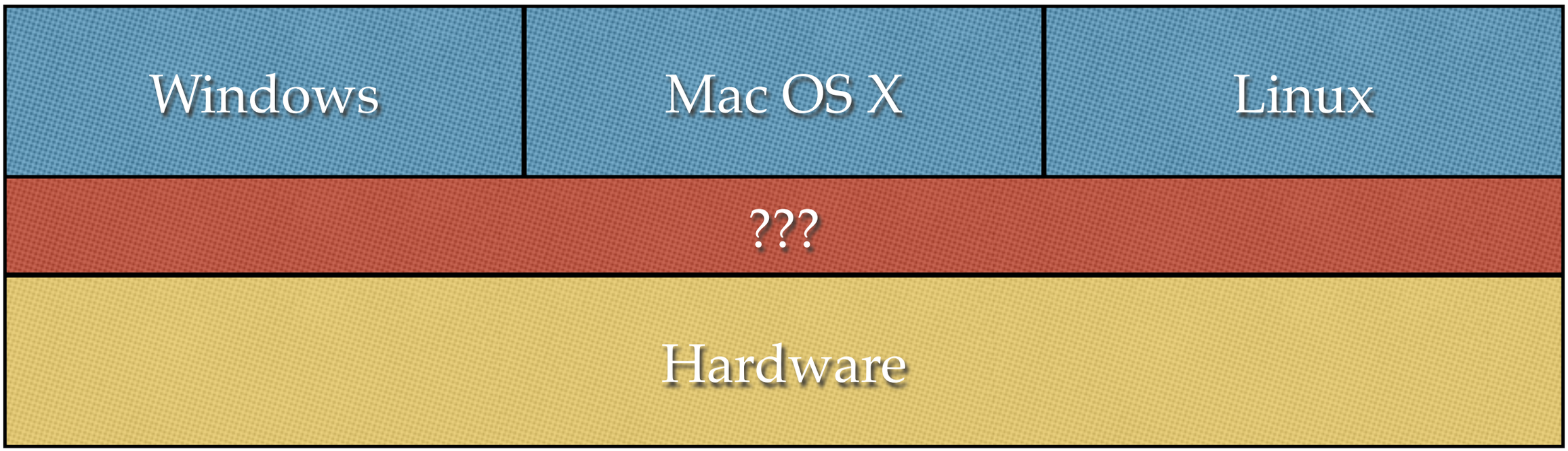
Microkernel Successes

- QNX (real-time operating system kernel)
- Symbian (mobile operating system)
- L4
  - Major work to reduce microkernel overheads
  - Can run Linux with L4Linux
  - seL4: Fully verified variant

Hypervisors and Virtual Machines

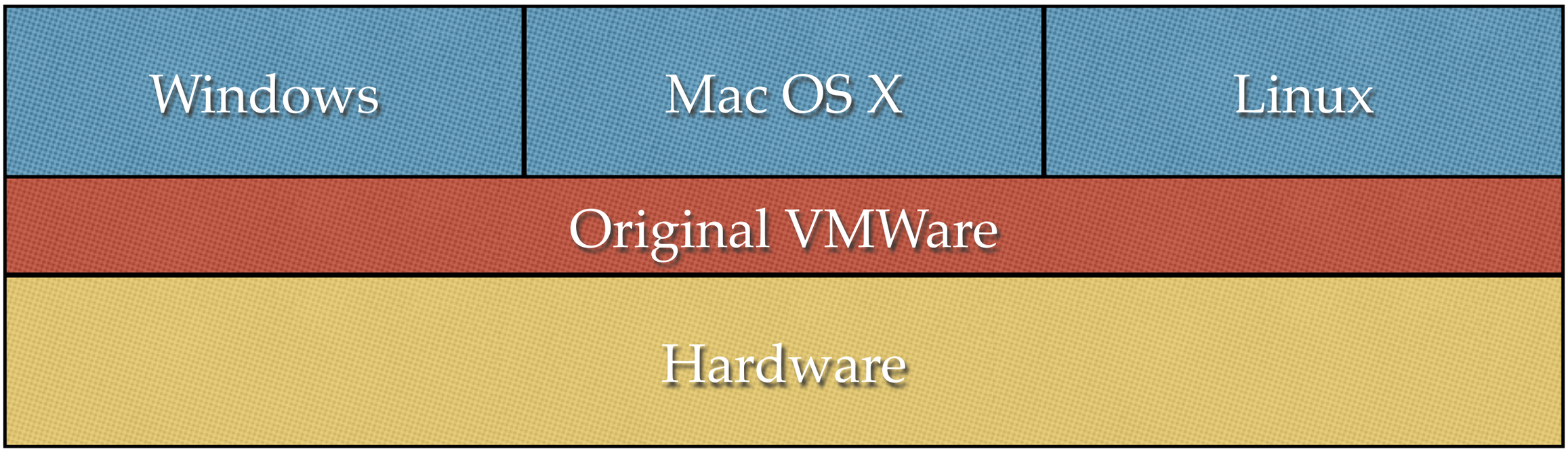
Virtual Machines

- Run multiple OS instances
- Migrate OS instances from one machine to another
- Software compatibility when hardware changes



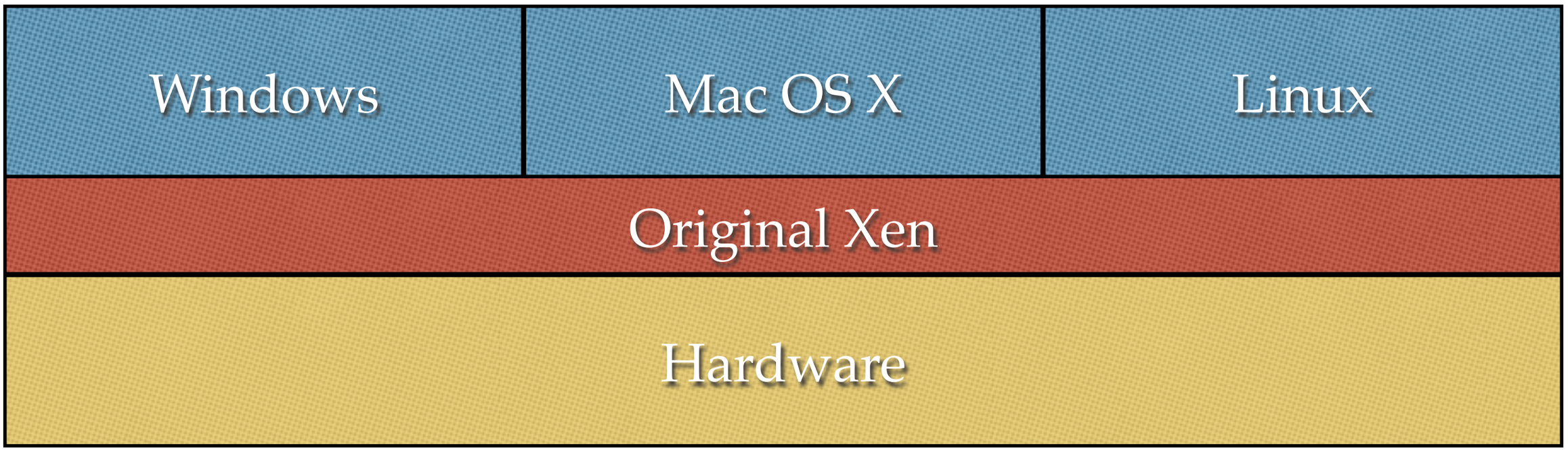
Compiler Translation

- Translate binary code (Original VMWare)
- Translate virtual code (JVM, OS/360)



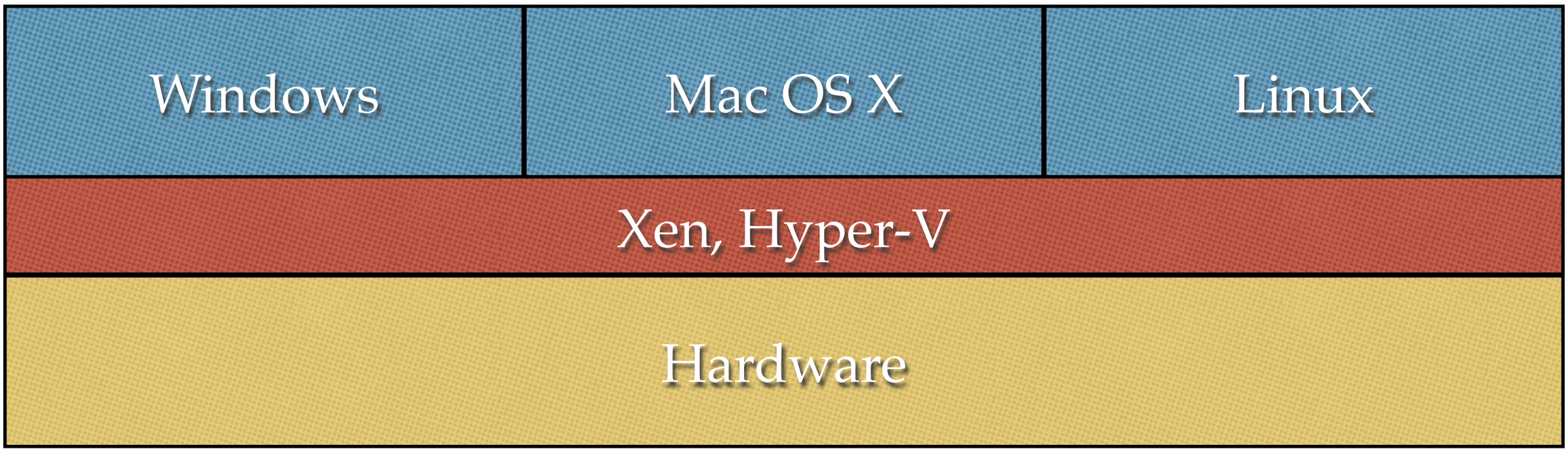
Para-Virtualizaiton

- Modify OS to interface with lower-level hypervisor
- Efficient but requires OS changes



Hardware Virtualization

- Hardware provide new privilege layer under OS
- Efficient
- Compatible
- Requires new hardware



Credits

- Some slides based on slides from previous year
- Slides only to be used for instruction at the University of Rochester