montavista VISION 2007

EMBEDDED LINUX DEVELOPERS CONFERENCE

Linux 101

Joe Green

MontaVista Software

montavista

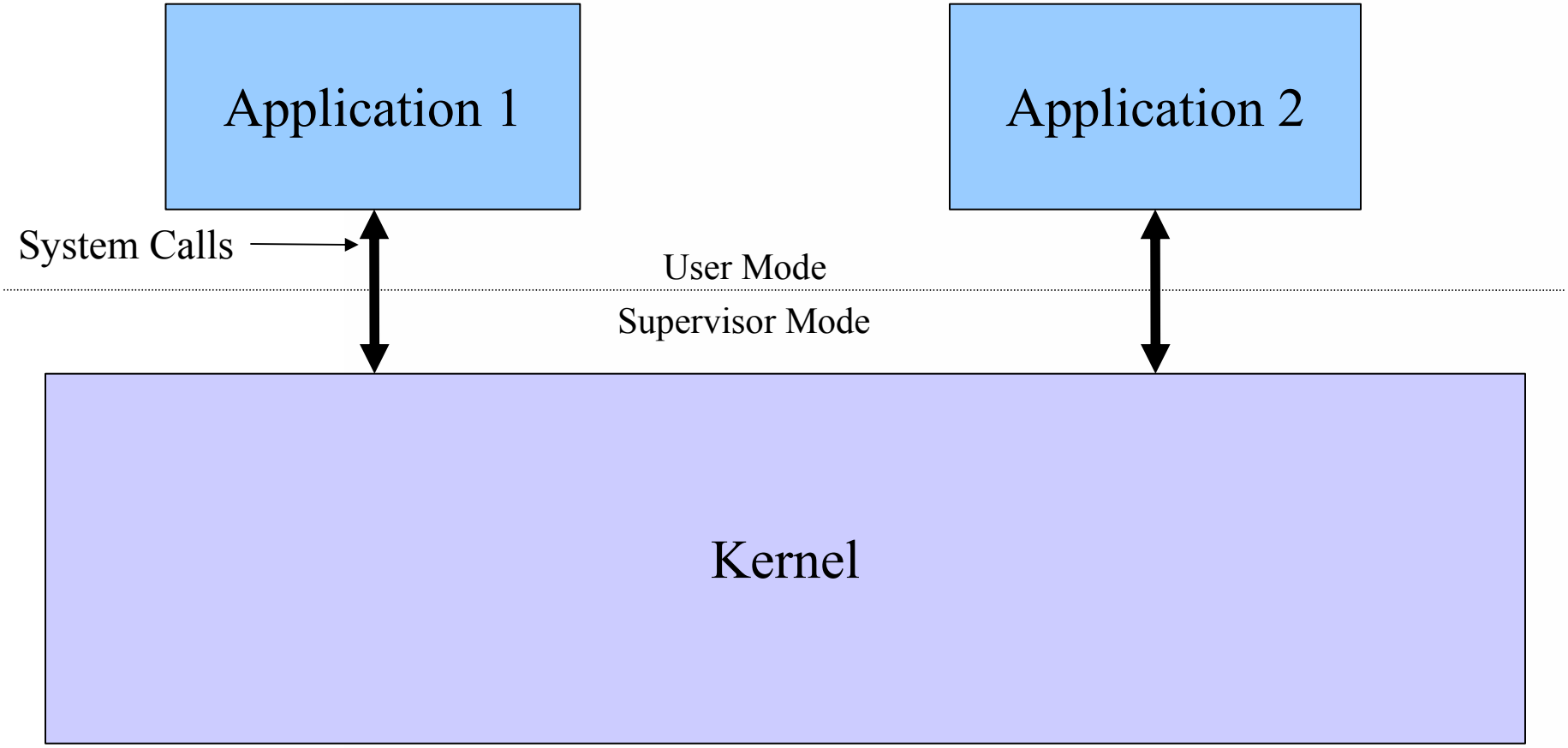
Linux 101

- Kernel
- User Space
- RPM Package Manager
- Community Development

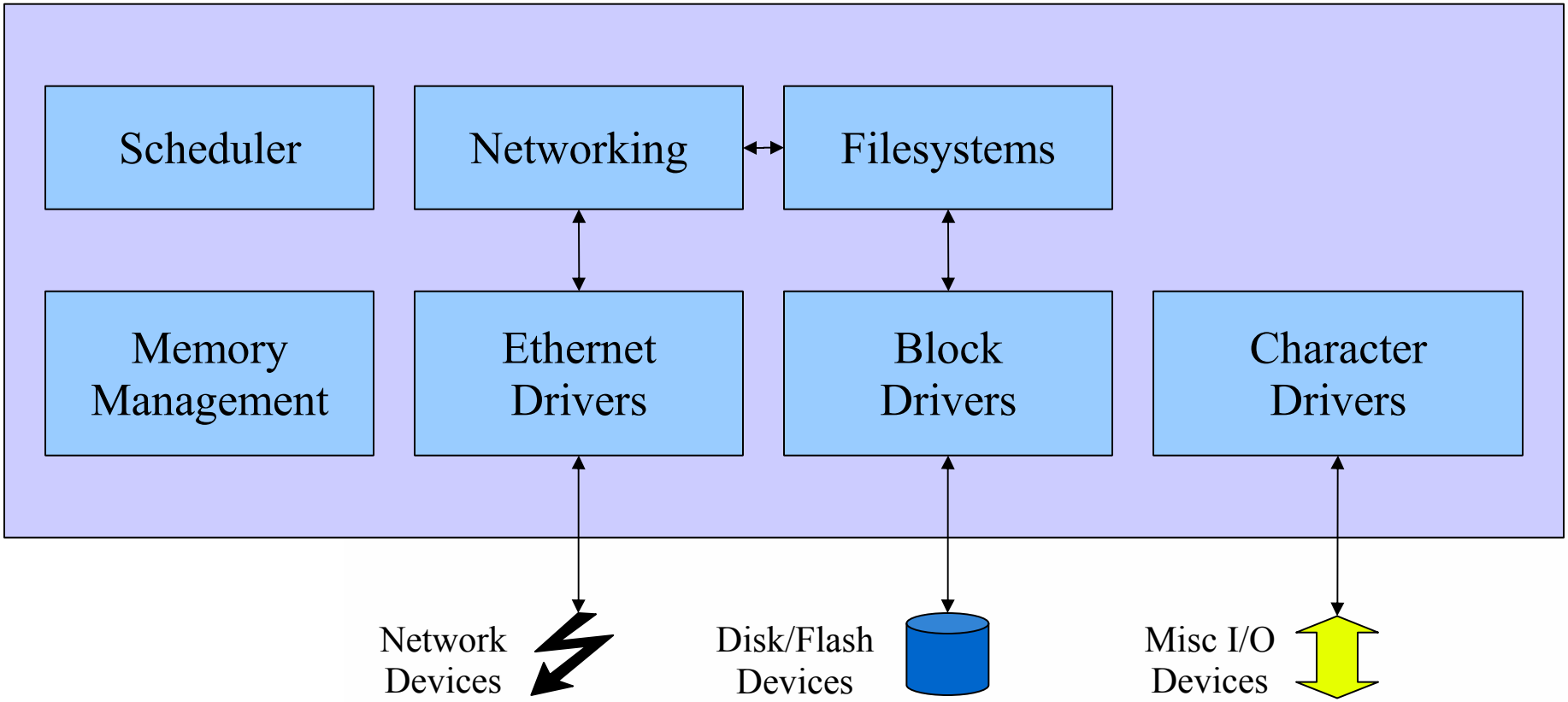
What is Linux?

- A UNIX-like operating system
  - Kernel independently developed by Linus Torvalds and others
  - Community development
  - Open source
  - Nominally compliant with POSIX, Single UNIX Specification
- Free Software Foundation (FSF)/GNU
  - Compilers and tools
  - Utilities
  - GNU/Linux

Kernel mode vs. User Mode



Kernel Subsystems



Device Drivers

- System calls
  - open
  - close
  - read
  - write
  - mmap
  - ioctl vs. /sys
- Device Frameworks
  - Networking
  - USB
  - Disk devices
- Filesystem interfaces
  - /dev
  - /sys
  - /proc
- Static linking vs. modules

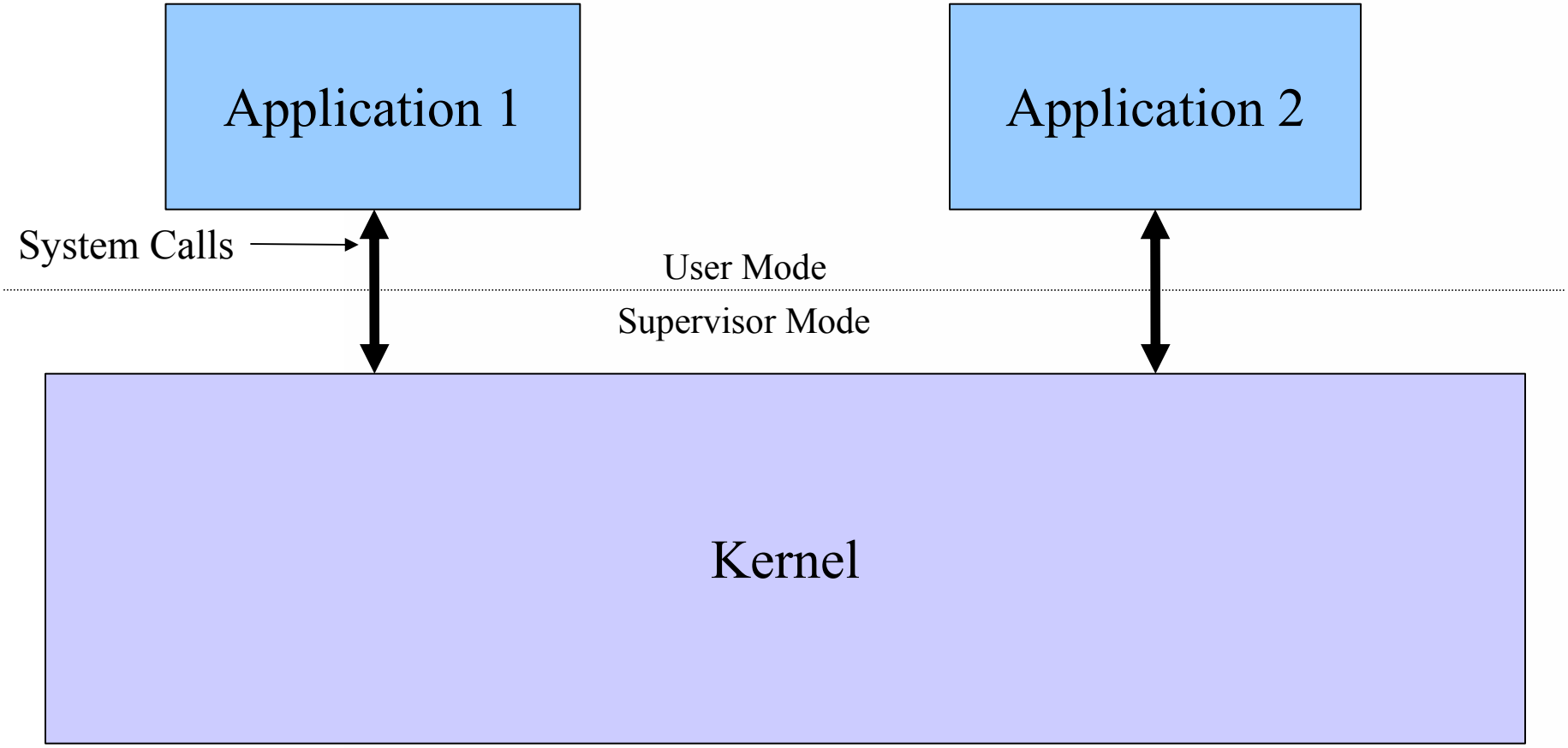
Kernel Licensing Issues

- I am not a lawyer. Don't sue me.
- GNU Public License (GPL)
- Binary modules
  - Derived work
  - Community HATES binary modules
  - Some interfaces declared to require GPL modules
- User space drivers
  - *Memory-mapped I/O can be mapped by mmap()*
  - *Interrupt handling requires a simple driver*
  - *Hooks to circumvent GPL frowned upon*
  - *Supported framework under development*

Kernel Documentation

- Kernel source tree Documentation directory
- Web sites
  - kernel.org
  - kernelnewbies.org
  - LWN.net
- Books
  - *Linux Kernel Development, Robert Love*
  - *Linux Device Drivers, Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman*

User Space

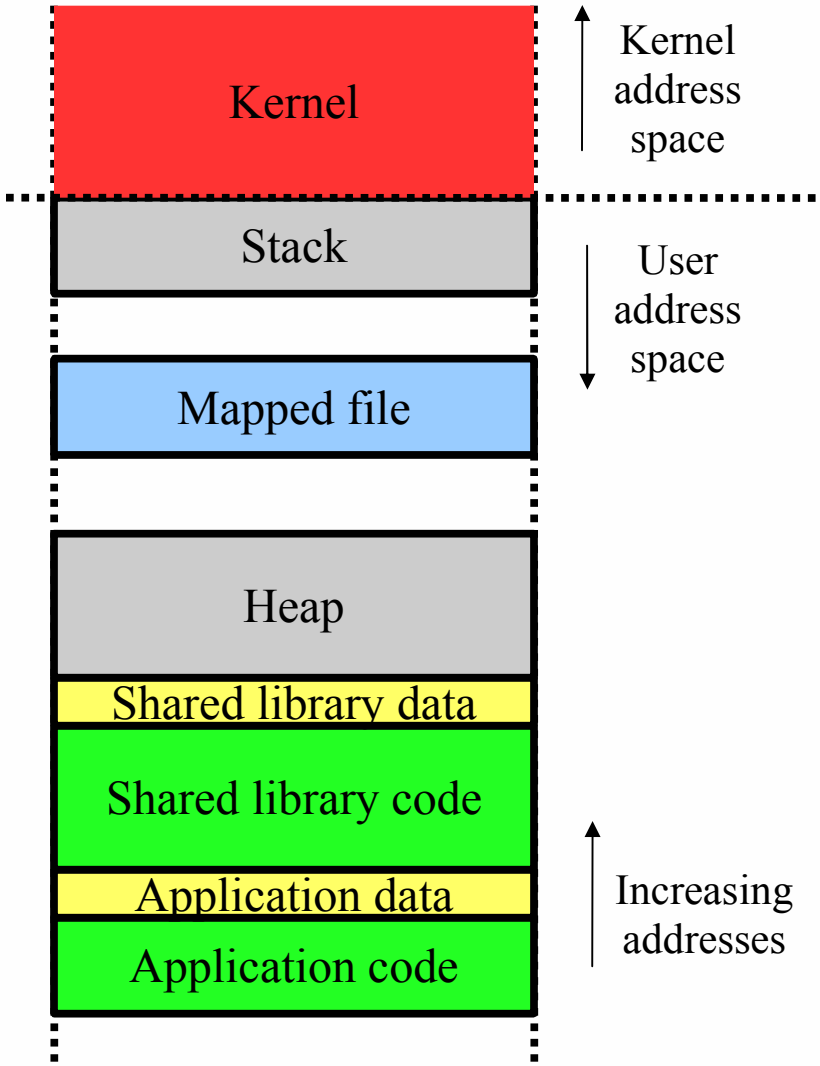

Application 1
Application 2
System Calls
User Mode
Supervisor Mode
Kernel

Processes

- Each application occupies a separate *process*
- *Each process has a unique virtual address space*
  - *Virtual addresses are mapped to physical memory by a hardware memory management unit (MMU)*
- *Each process has a set of file descriptors*
  - *A file descriptor is an integer handle which represents an open file or device*
  - *All I/O operations are performed on file descriptors*
  - *Well-known file descriptors represent standard input, output and error devices*
  - *The contents of an object represented by a file descriptor can be mapped to a portion of the virtual address space*

Process Virtual Memory

- Memory space consists of a set of mapped files
- “Anonymous” memory (stack, heap) is special zero-filled mapping
- By default, operating system chooses where to create mappings
- Kernel occupies top of address space; not accessible to user
- Initial stack grows down from top of user address space
- Heap grows up from end of program
- Program and shared library code and data are mapped directly from the binary files



Virtual Memory Mappings

- mmap() can be used to map an open file or device to memory
- Pages are faulted in as needed
  - Only a small part of a program, library or file need be in memory at any time
  - Less used pages may be discarded from memory at any time
- Copy-on-Write
  - First write to a page faults and creates a private copy
  - Useful for data in forked processes, initialized data in programs, libraries

Processes and Threads

- Processes
  - Separate virtual memory, file descriptors
  - fork() creates duplicate of current process
  - exec() overlays new program over current process
- Threads
  - Share virtual memory, file descriptors
  - POSIX threads (pthreads)
  - LinuxThreads
    - Deprecated
    - No inter-process synchronization
  - Native POSIX Thread Library (NPTL)

Inter-Process Communication (IPC)

- Shared memory
  - Mapped shared files
- pthreads
  - Mutexes, semaphores, condition variables
  - Accessed through shared memory
- Signals
- Sockets, Pipes
- System V IPC
  - Shared memory
  - Message queues
  - Semaphores

Security and Access Control

- Access control determines which data and device files can be accessed by an application
- Basic Linux access control
  - File ownership (user, group, other)
  - Permissions (read, write, execute)
  - “root” user all-powerful
- Capabilities
- Advanced security models
  - Access Control Lists (ACLs)
  - Security-Enhanced Linux (SELinux)
  - AppArmor, etc.

Application Programming Documentation

- Online documentation
  - man pages
  - info pages
  - /usr/share
- Single UNIX Specification (http://www.unix.org/)
- Linux Standard Base (http://www.linuxbase.org/)
- Many books on Linux and UNIX programming

RPM Package Manager

- Most Linux distributions provide each software component as an archive of files called an RPM package
- Originally called the RedHat Package Manager, developed by RedHat
- Administrative tool
  - Manages installation, removal, update of software
  - Queries package information
- rpmbuild
  - Controls build procedure
  - Specifies package information
  - Defines package dependencies and conflicts
- Books (Red Hat RPM Guide, Maximum RPM)
- http://www.rpm.org/

Community Development

http://www.globalvoicesonline.org/2006/04/29/man-and-anthill-guyana/

UTSL = “Use the source, Luke!”