Operating System Projects

2016 Spring

Joonwon Lee

Bon Keun Seo, Sangwook Kim, Sunghun Kim

Objective

- Understanding the architecture of practical OS
  - ***Based on Linux kernel***
  - Process and thread
  - System call and interrupt
  - Memory management
  - Storage management
  - …

Material

- **Linux kernel versions later than 3.0**
  - Recommends ubuntu 14.04 (linux-4.2)

- **Books**
  - Robert Love,
    Linux Kernel Development, 3rd edition, Addision-Wesley.
  - Wolfgang Mauerer,
    Professional Linux Kernel Architecture, Wrox.
  - Daniel P. Bovet and Marco Cesati,
    Understanding the Linux Kernel, 3rd edition, O'reilly.

Schedule

Week	Title	Projects (due)
1	Introduction to Linux kernel	
2	Tasks, scheduler, system call and interrupt	0. Environment setup
3-7	Memory management and virtual memory	0.5. Start up (module)
8	(Mid-term exam)	1. Virtual memory
9-12	Block device and device mapper	2. Block device
13-15	Virtual file system and ext4fs	
16	(Final exam)	3. File system

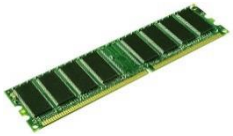
Grading

- **Assignment**: 90%
  - Project 0, 0.5: 20% (person)
  - Project 1~3: 70% (team)
- **Presentation**: 10% (person)
  - Progress reports
  - Final presentation
- **Bonus points**
  - Brilliant ideas in projects or presentations

Introduction to Linux

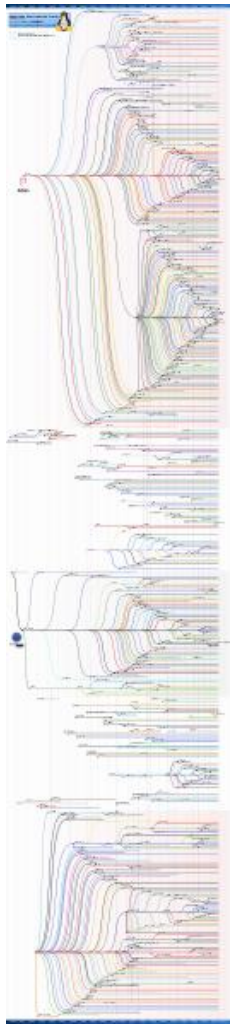
Operating system

- What is an "operating system"?
  - A software that application software operates on



Linux operating system

- An UNIX-like operating system by Linus Torvalds



fedora

debian

ubuntu

redhat

gentoo linux™

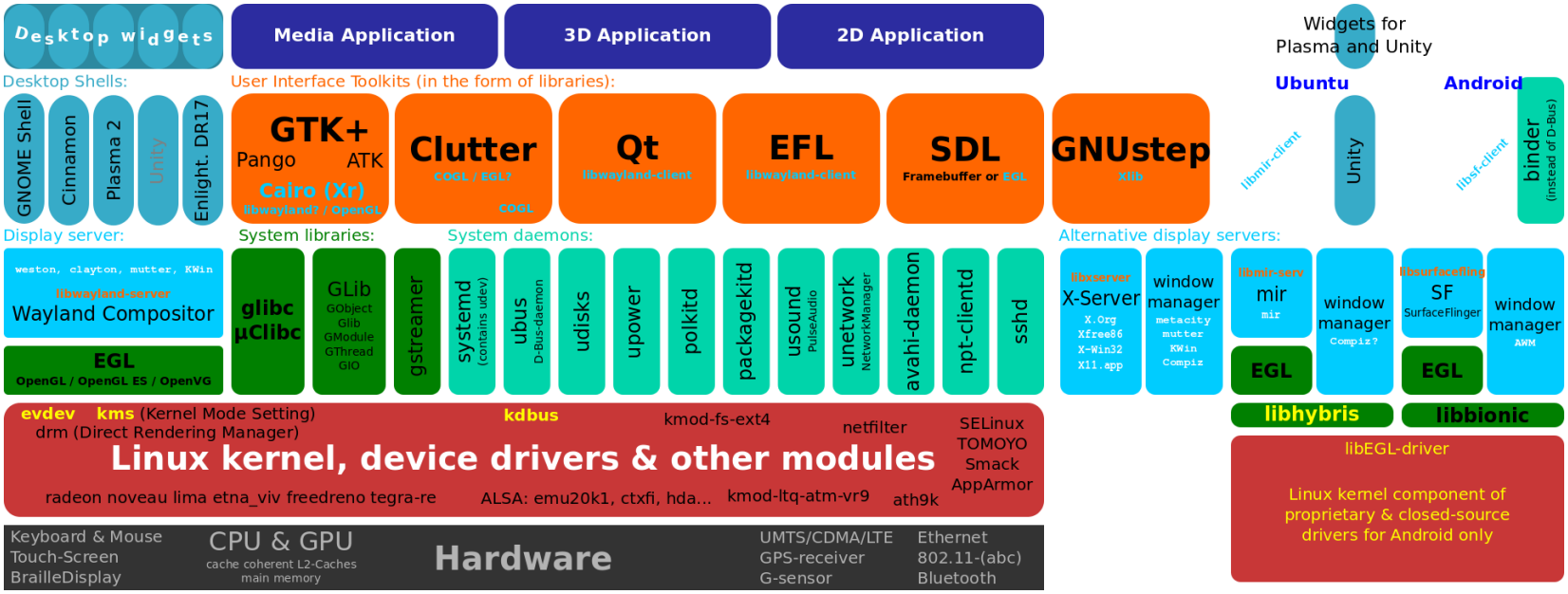
suse

archlinux

Linux distributions

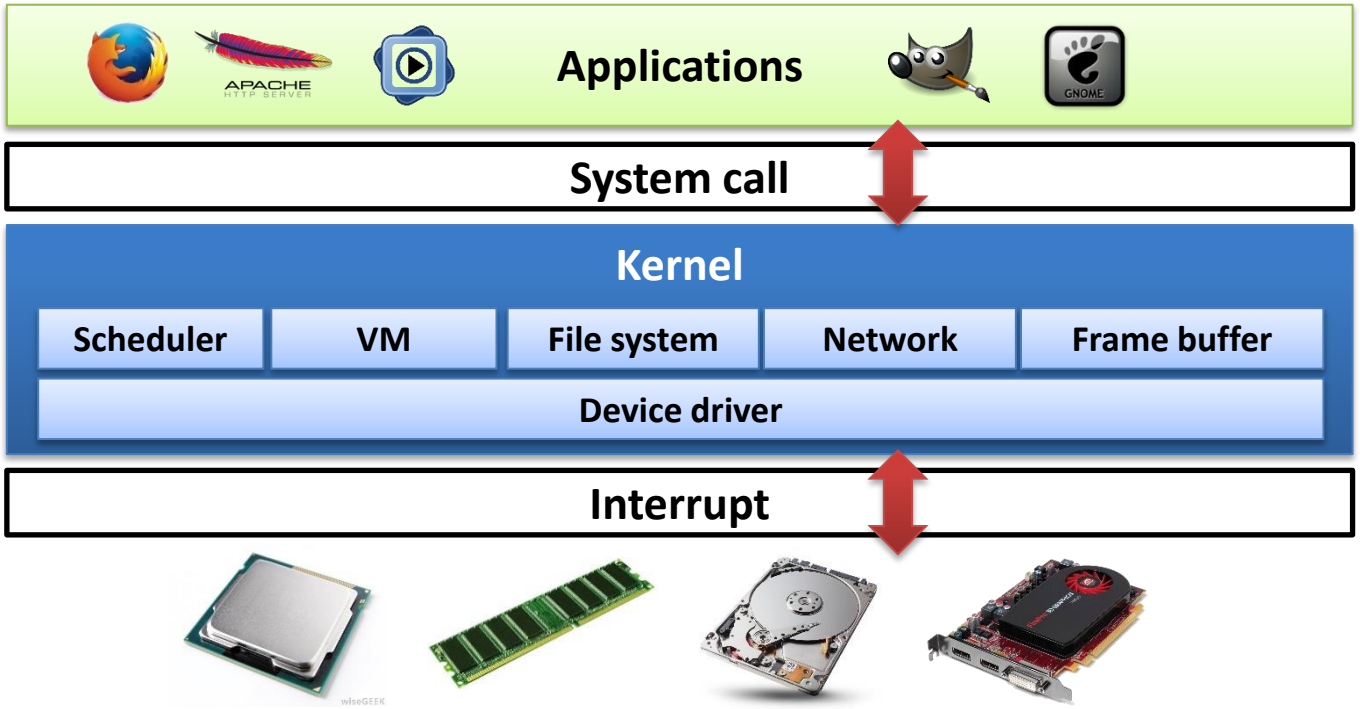
GNU/Linux

- **Linux distribution =**
  - **Applications**
  - **GNU** (standard C library + system utilities)
  - ***Linux*** (kernel)

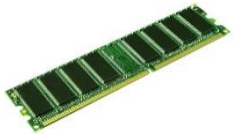


Linux kernel

- A pre-emptive multi-process monolithic kernel
  - Event driven architecture
  - Supports multiple architectures
  - Kernel modules to extend



Kernel abstraction

-  Process, thread
-  Virtual memory
-  Block device, file  Character device
-  Frame buffer device
-  Socket

Kernel feature (1)

- Compatible to **POSIX**
- **Multi-architecture**
  - x86, ARM, MIPS, …
  - Multi-processor, NUMA, …
- **Multi-process, multi-thread**
  - Fair, time sharing scheduler
- **Synchronization primitives**
  - Semaphore, spinlock, RCU, futex, …

Kernel feature (2)

- **Device mapper**
  - LVM, software RAID, flash caching, …
- **File system**
  - ext4, btrfs, f2fs, …
  - FUSE
- **OSS sound**
- **Kernel Virtual Machine**
- **Wide varieties of device drivers**
  - Block, network, graphics, sound, tty, …

Kernel source tree

- **Documentation**
- **arch** : architecture dependent codes
  - Boot, interrupt, system call and memory management
- **kernel** : scheduler and synchronization
- **mm** : memory allocation and page caching
- **block** : block device abstraction
- **net** : network stack
- **fs** : virtual file system and file systems
- **drivers** : device drivers (physical and virtual)

Kernel development

- Coding style / indentation
  - Refer to Documentation/CodingStyle
  - Written in C, but with an object-oriented programming style
- The best documentation is the code itself
  - Most of codes are not documented
  - Or, the documents are stale
- Beware of race conditions and the interrupt context