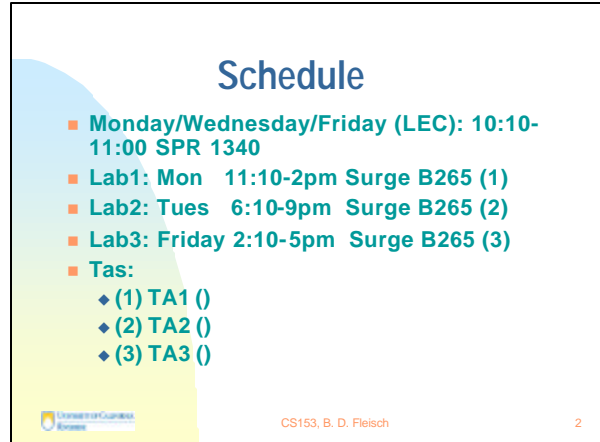


# CS153 Operating Systems

Prof. Brett D. Fleisch  
University of California, Riverside  
Surge 322, Office Hours: MWF 11-12 or by appointment  
[brett@cs.ucr.edu](mailto:brett@cs.ucr.edu)

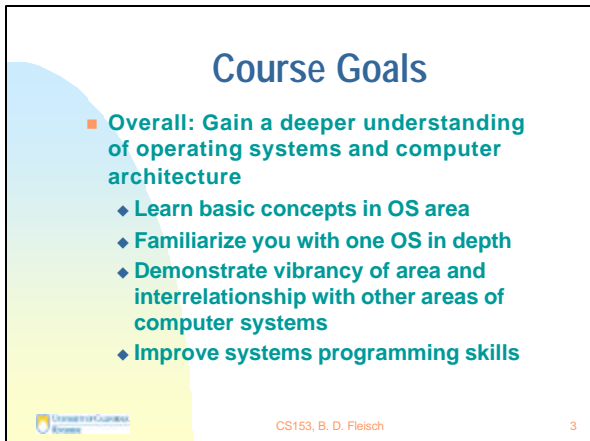
University of California, Riverside  
CS153, B. D. Fleisch 1



## Schedule

- Monday/Wednesday/Friday (LEC): 10:10-11:00 SPR 1340
- Lab1: Mon 11:10-2pm Surge B265 (1)
- Lab2: Tues 6:10-9pm Surge B265 (2)
- Lab3: Friday 2:10-5pm Surge B265 (3)
- Tas:
  - ◆ (1) TA1 ()
  - ◆ (2) TA2 ()
  - ◆ (3) TA3 ()

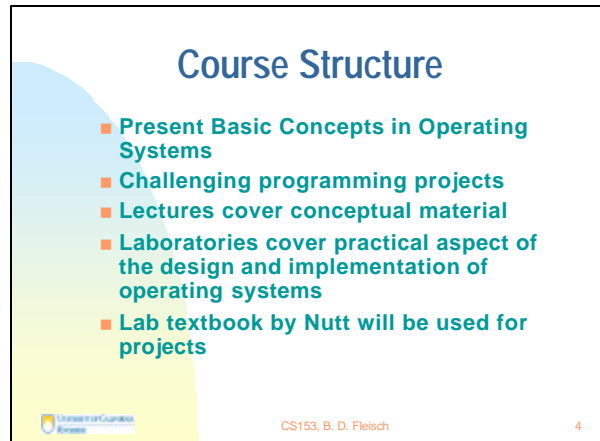
University of California, Riverside  
CS153, B. D. Fleisch 2



## Course Goals

- Overall: Gain a deeper understanding of operating systems and computer architecture
  - ◆ Learn basic concepts in OS area
  - ◆ Familiarize you with one OS in depth
  - ◆ Demonstrate vibrancy of area and interrelationship with other areas of computer systems
  - ◆ Improve systems programming skills

University of California, Riverside  
CS153, B. D. Fleisch 3




## Course Structure

- Present Basic Concepts in Operating Systems
- Challenging programming projects
- Lectures cover conceptual material
- Laboratories cover practical aspect of the design and implementation of operating systems
- Lab textbook by Nutt will be used for projects

University of California, Riverside  
CS153, B. D. Fleisch 4

## Topics to be Covered

- Operating system overview and history
- Architecture component overview
- Process description and control
- Threads
- Concurrency: mutual exclusion and synchronization
- Concurrency: deadlock and starvation
- Memory management
- Virtual memory
- Uniprocessor scheduling
- I/O management and disk scheduling
- File management

 CS153, B. D. Fleisch 5


## Course Requirements

**Prerequisites**

- Good programming skills in C and some familiarity with UNIX
- Ability to work independently and as a member of a team

**Expected work**

- Midterm exam (Feb 15)
- Textbook Reading, attend lectures and labs, review lab textbook (Nutt) prior to projects
- Homeworks (3-4 total) on assigned reading and past lectures
- 3-4 programming projects in Linux
- Bonus Work assigned in class
- Final exam (March 18-23 TBA)


 CS153, B. D. Fleisch 6

## Books

- Stallings, W., *Operating Systems: Internals and Design Principles*, Prentice-Hall, 4/e, 2000
- Nutt, Gary, *Linux Kernel Projects*, Addison Wesley, 1999. Text used primarily for background for project assignments and in lab.

**Linux Resources:**


- Beck, M., Bohme, et. al., *Linux Kernel Internals*, Addison Wesley, 1998
- Rubini, A., *Linux Device Drivers*, O Reilly, 1998.
- Card, R., Dumas, E., Mevel, Franck, *The Linux Kernel Book*, John Wiley and Sons, 1997.

 CS153, B. D. Fleisch 7

## Grading

Grade Component	Points Available
Midterm Exam	150
Programming Projects (4)	400
Homeworks (3-4)	100
Bonus Credit Work or Homework (24) (assigned for extra credit in class, no credit if not in class when assigned)	99
Final Exam	250

Grade Range	Points
A	>= 900
B	800-899
C	700-799
D	600-699
F	< 600

 CS153, B. D. Fleisch 8


## Projects

Goals

Structure

- Improve systems programming skills
- Learn components of operating systems and how to design and implement these components

- Some Individual projects
- Some partnered projects as the quarter progresses
- Projects increase in difficulty as the quarter progresses




CS153, B. D. Fleisch

9

## Linux

- Open source version of UNIX developed by Linus Torvalds and developers from around the world
- Several popular versions of Linux that run on PCs including Red Hat Linux, Debian and Slackware
- Very popular version of UNIX due to architecture independence and portability




CS153, B. D. Fleisch

10

## Linux Popularity

- Rebellion against Microsoft
  - ◆ Judge determines that Microsoft is a predatory monopoly boosting systems such as Linux
- Linux has been ported to many popular architectures: DEC Alpha, Sparc, Intel architecture, PowerPC, Amiga
- Conforms to POSIX 1003.1 standard
- Open source code under GNU source license: freely available, you can copy, use, modify the source code free of charge
- Avoids intellectual property rights issues associated with closed source code




CS153, B. D. Fleisch

11

## Linux Popularity (2)

- According to market researcher IDC, Linux is the fastest-growing operating system for servers
  - Linux's worldwide market share of new and upgraded operating systems for servers was 27 percent in 2000.
  - It was second only to Microsoft, which stood at 42 percent
- Source: 12-17-2001 Salt Lake Tribune



CS153, B. D. Fleisch

12

## Linux Goals

- Learn some Linux internals and to appreciate the quality (and problems) in the underlying source code that fundamentally drive these businesses
- Learn about robustness, portability and configurability of Linux systems
- Learn how to build, modify and use the source code base
- Learn how to build device drivers and improve Linux's ability to support new devices

University of Cambridge  
CS153, B. D. Fleisch 13

## Linux Main Characteristics (1)

- Multitasking
- Multi-user access
- Multiprocessing
- Architecture Independence
- Demand load executables
- Paging
- Dynamic Cache for hard disk
- Shared Libraries

University of Cambridge  
CS153, B. D. Fleisch 14

## Linux Main Characteristics (2)

- Support for POSIX 1003.1 and in part for System V and BSD
- Support for various types of executable files
- Memory protected mode
- Support for variety of keyboards and fonts
- Support for different types of filesystems
- Support for TCP/IP, SLIP and PPP

University of Cambridge  
CS153, B. D. Fleisch 15

## VMware

- A thin software layer between the Intel architecture and the operating system
- Virtualizes the hardware and managing all hardware resources including the network
- The OS can't tell the difference between operating in a VM or in a "real" machine.
- A Virtual Machine (VM) is defined by Popek and Goldberg (in their paper "Formal requirements for virtualizable third generation architectures," Communications of the ACM, Vol 17, July 1974) as an "efficient, isolated duplicate of a real machine".

University of Cambridge  
CS153, B. D. Fleisch 16

## VMWare (2)

- With VMware Workstation, operating systems and applications run inside virtual machines.
- You can create a whole set of computers - whether you operate under Linux, Windows NT or Windows 2000. And forget about dual booting. You can run them all at the same time
- Will save enormous time for you in debugging the OS and rebooting, just do it under VMware



CS153, B. D. Fleisch

17

## Lateness Policy

- Projects are planned with extra time in mind for system outages *so begin early*
- *There will be no extensions for project assignments*
- Late projects penalized 25% per calendar day late
- Points deducted for lateness are not differentiated from points deducted for other reasons in my records
- *All other work (non-project) is not accepted late*
- *Contact me concerning illness, sickness in family, and other personal problems that could delay work*



CS153, B. D. Fleisch

18

## Lateness Implications

- Implications of being late (assumes no bonus points from extra credit problems):
  - ◆ 2 days late on 3 projects
    - Reduces project score to 300 points max yielding a B range grade at best
  - ◆ 2 days late on all 4 projects
    - Reduces your project score to a maximum of 200 points yielding a C range grade at best
  - ◆ 3 days late on all 4 projects
    - Reduces your project score to a maximum of 100 points putting you in the D range grade at best, if no bonus points an F



CS153, B. D. Fleisch

19

## Planned Project Returns

- We plan to return projects two weeks after they are handed in
- Late projects will **not be** returned with on-time assignments. Late assignments will be returned with the next batch of returns for the subsequent assignment usually.
- Expectations concerning the quality of the project will be discussed in the lab including commenting standards and code quality issues




CS153, B. D. Fleisch

20


## Honor Code, Academic Honesty

- Everything you turn in should be your own work
- Write down who you discussed assignments with, all external sources you used, reference textbooks you used
- Don't collaborate unless authorized to do so
- Be honest – you know what cheating is and isn't
- All programs are run through the cheat checker
- A student found cheating will be assigned the grade F for the course


CS153, B. D. Fleisch
21


## Course Mechanics

- Course homepage:
  - ◆ [http://www.cs.ucr.edu/~brett/cs153\\_w02/cs153.html](http://www.cs.ucr.edu/~brett/cs153_w02/cs153.html)
- Course notes will be made available on-line before each class *no later* than 9am on the day of the lecture
- Announcements will be posted to the *course web page*, as needed
- Newsgroup for discussion: [cs153@lists.cs.ucr.edu](mailto:cs153@lists.cs.ucr.edu)


CS153, B. D. Fleisch
22

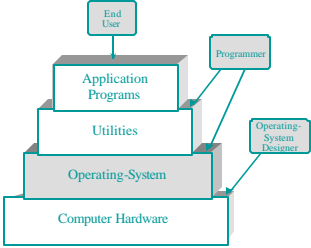
## Topics for Lectures 1-5


- What is an operating System?
  - ◆ Stallings Chapter 2
- Computer Architecture Overview
  - ◆ Stallings Chapter 1
- Read chapters 1 and 2


CS153, B. D. Fleisch
23

## What is an Operating System?


- A software layer between the hardware and the application programs/users
  - ◆ provides its own interface (which is a so-called virtual machine interface): ease of use, safety, evolution
- A resource manager that
  - ◆ allows programs/user to share the hardware resources: fairness and efficiency




CS153, B. D. Fleisch
24

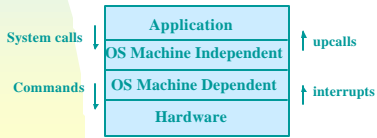
## Operating System


- Is a **program** that controls the execution of application programs
  - ◆ OS must relinquish control to user programs and regain it safely and efficiently
  - ◆ Tells the CPU **when** to execute programs
- An interface between the user and hardware
- Masks the details of the hardware to application programs
  - ◆ Hence OS must deal with hardware


CS153, B. D. Fleisch
25

## OS Flow of Control


- Users access OS via interface calls called *system calls*
- Operating system issues internal commands to manage devices and resources
- Internal command completion signalled by *interrupts*
- OS signals users or applications by *upcalls*
- OS complexity arises from managing the concurrency internally and possibly from the applications




CS153, B. D. Fleisch
26

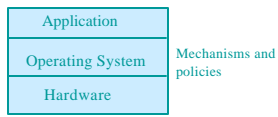
## Services Provided by OS


<ul style="list-style-type: none"> <li>■ Program creation                             <ul style="list-style-type: none"> <li>◆ editors and debuggers</li> </ul> </li> <li>■ Program execution</li> <li>■ Access to I/O devices</li> <li>■ Controlled access to files</li> <li>■ System access</li> <li>■ Accounting                             <ul style="list-style-type: none"> <li>◆ Collect Statistics</li> <li>◆ Monitor Performance</li> <li>◆ Anticipate Enhancements</li> <li>◆ Billing users</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>■ Error detection and response                             <ul style="list-style-type: none"> <li>◆ internal and external hardware errors                                     <ul style="list-style-type: none"> <li>■ memory error</li> <li>■ device failure</li> </ul> </li> <li>◆ software errors                                     <ul style="list-style-type: none"> <li>■ arithmetic overflow</li> <li>■ access forbidden memory locations</li> </ul> </li> <li>◆ Denial of application request</li> </ul> </li> </ul>
---	---


CS153, B. D. Fleisch
27

## OS Separates Mechanisms and Policies


- **Mechanism:** Data structures and operations that implement some abstraction e.g. priority queue
- **Policy:** procedure used to select the ways to accomplish certain objectives from alternatives e.g. least remaining time first, first come first serve, round robin, etc




CS153, B. D. Fleisch
28

## Difficulties in OS Design

- **Improper synchronization**
  - ◆ E.g. ensure a program waiting for an I/O device receives the signal
- **Failed mutual exclusion**
  - ◆ Permit only one program at a time to perform an action on a portion of data
- **Deadlock**
  - ◆ Two or more programs wait endlessly for each other to perform an operation which they can never perform due to their waiting
- **Maintainability**
  - ◆ Well written and commented code is essential for maintainability

 CS153, B. D. Fleisch 29