

计算机操作系统

薛瑞尼

计算机科学与工程学院

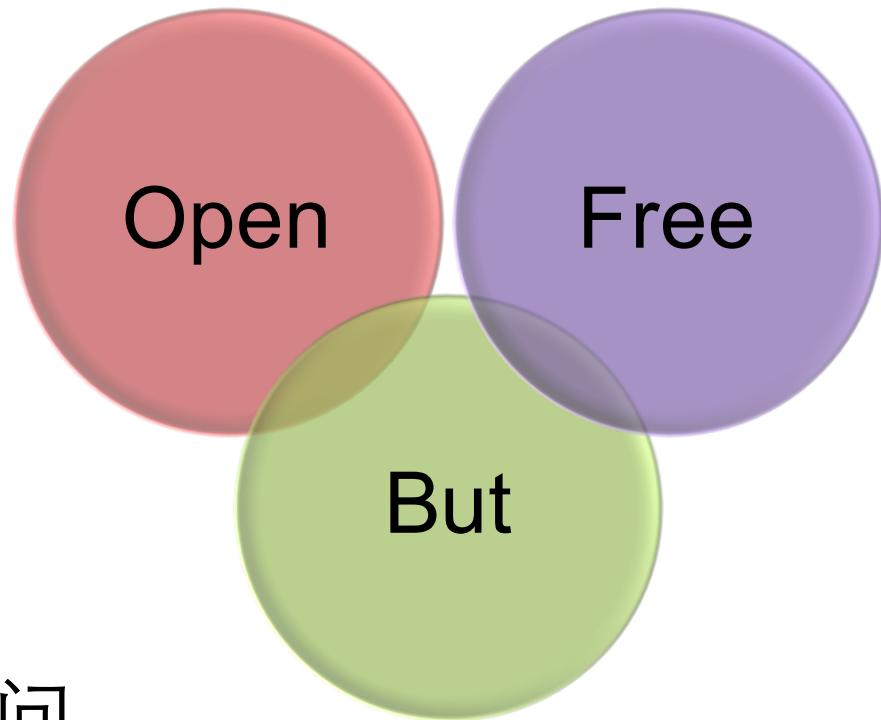
16/2/25

薛瑞尼

- xueruini@uestc.edu.cn
- 主楼B1-103、601
- 课堂群：489633484

课堂纪律

- 手机：关闭或者调成震动
- 不要影响其它同学
 - 说话
 - 睡觉
 - 进食
 - 走动
- 可随时打断老师提问

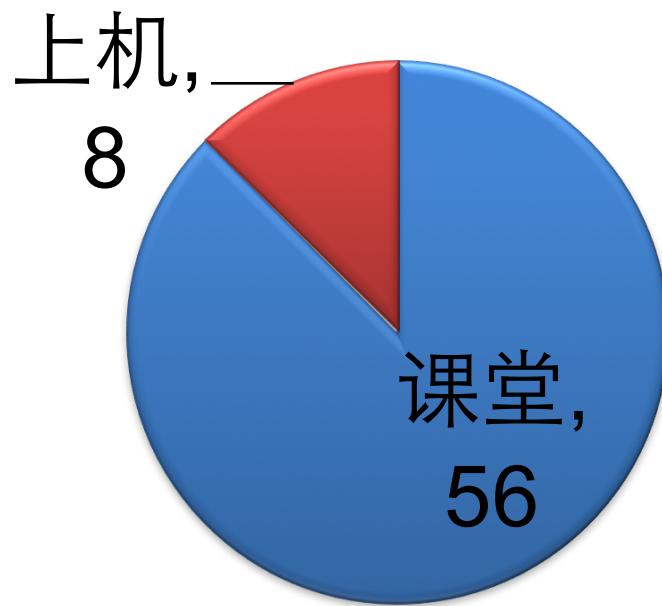


相关/先行课程

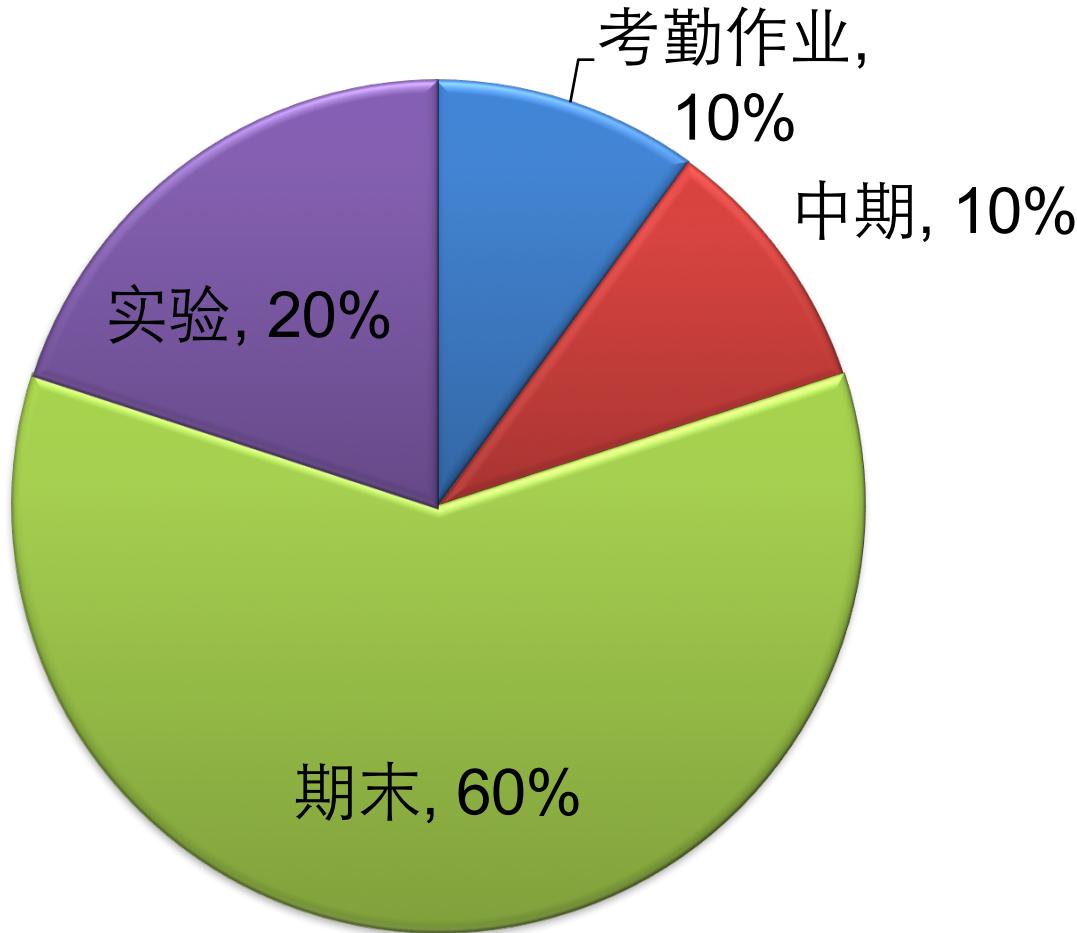
- 计算机组装原理
- 计算机系统结构
- 数据结构
- C和汇编语言

教学安排

- 总课时： 64
 - 1-16周
 - 周二 (3, 4)
 - 周四 (1, 2)



成绩组成



如何提交作业

- <https://github.com/os-uestc/assignments-and-labs-2016>
- 访问链接：
<https://classroom.github.com/assignment-invitations/7a168a9baa44fcfb9e1ed141a9f00208>
- [网页在线解释]

课堂形式

- 5次 “我来讲课” 的机会
 - 内容：课堂知识的扩展（允许超越课堂进度）
 - 时间：不超过一节课
 - 小组：不超过3人，可以1人讲也可以全上
- 先报名先得
 - 可以先占位置，再定题目
- 奖励
 - 所有人平时成绩满分（可以不做作业）
 - 最后一节课，全体同学QQ投票，前3组有奖品！

参考书

- Andrew S.Tanenbaum, *Modern Operating Systems (3rd Edition)*, Prentice Hall, 2007
- Abraham Silberschatz, Peter B. Galvin Greg Gagne, *Operating System Concepts (7th Edition)*, John Wiley & Sons, 2005
- William Stallings , *Operating Systems: Internals and Design Principles (7th Edition)*, Prentice Hall, 2011
- Richard McDougall, Jim Mauro. *Solaris Internals: Solaris 10 and OpenSolaris Kernel Architecture(2nd Edition)* Prentice Hall, 2006

参考书

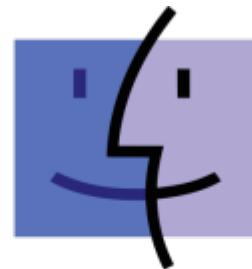
- Mark E. Russinovich, David A. Solomon.
Microsoft Windows Internals(5th Edition)
Microsoft Press, 2009
- Daniel P. Bovet. *Understanding the Linux Kernel(3rd Edition)*, O'Reilly Media, 2005
- www.wikipedia.org



参考课程

- MIT 6.828: Operating System Engineering
 - <http://pdos.csail.mit.edu/6.828/2011/>
- Tsinghua: Operating System
 - <http://os.cs.tsinghua.edu.cn/oscourse/>

认识哪些？



Mac[™] OS



X
Mac OS X

认识哪些？



iOS



android



BlackBerry™



Windows
phone

symbian

QNX®

认识哪些？



redhat



fedora



ubuntu



openSUSE



Mandriva



debian



gentoo linux



slackware
linux



KNOPPIX



CentOS

认识哪些？



为什么要学操作系统？

- 都有那么多操作系统了，我不大可能重写一个 (re-invent the wheel)
- 专业的标示
- 明了原理，理解问题
- For fun.

课程内容

操作系统介绍

进程管理

处理器调度

存储器管理

设备管理

文件管理

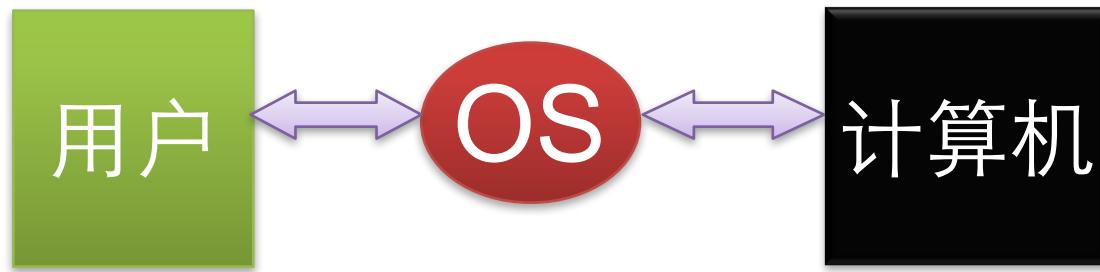
操作系统接口

1、操作系统概论

Operating System, OS

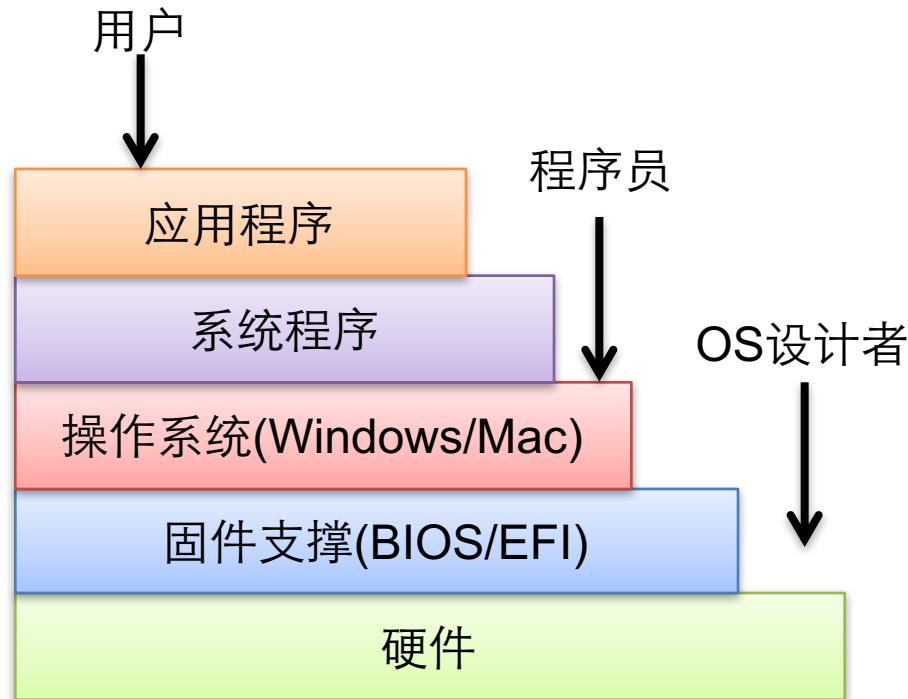
操作系统

- 裸机上的第一层软件，是对硬件系统功能的首次扩充，人与机器之间的界面。



计算机系统结构

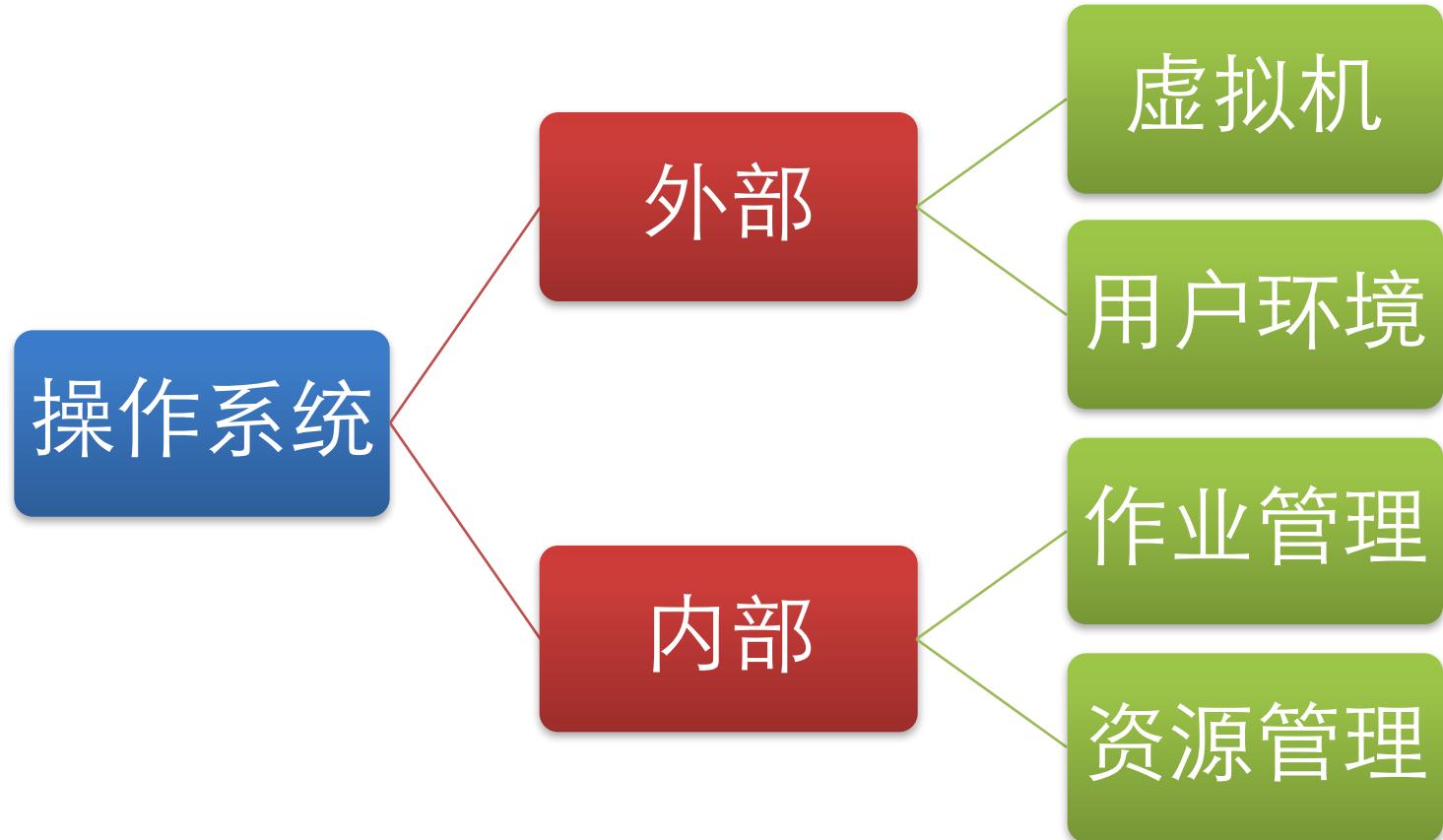
- 硬件
 - CPU, 内存, I/O设备
- 操作系统
 - 控制和协调硬件资源
- 系统程序
 - 编译器, 解释软件
- 应用程序
 - 用户如何使用系统资源解决实际问题
- 用户
 - 人, 或者其它机器



操作系统的抽象



什么是操作系统？



作业管理

- 系统**程序**
- 控制程序的执行，防止错误以及滥用
- 执行用户程序，使用户解决问题更容易
- 让计算机便于使用

资源管理

- 应用程序和硬件的接口
- 管理所有资源：协调资源的使用



定义之一

- 操作系统是一组控制和管理计算机硬件和软件资源，合理地对各类作业进行调度，以及方便用户使用的程序的集合。
- An operating system (OS) is a set of programs that manage computer hardware resources and provide common services for application software. ——wikipedia

操作系统的 目标

方便性

- 使计算机更易于使用

有效性

- 使用计算机资源更加有效

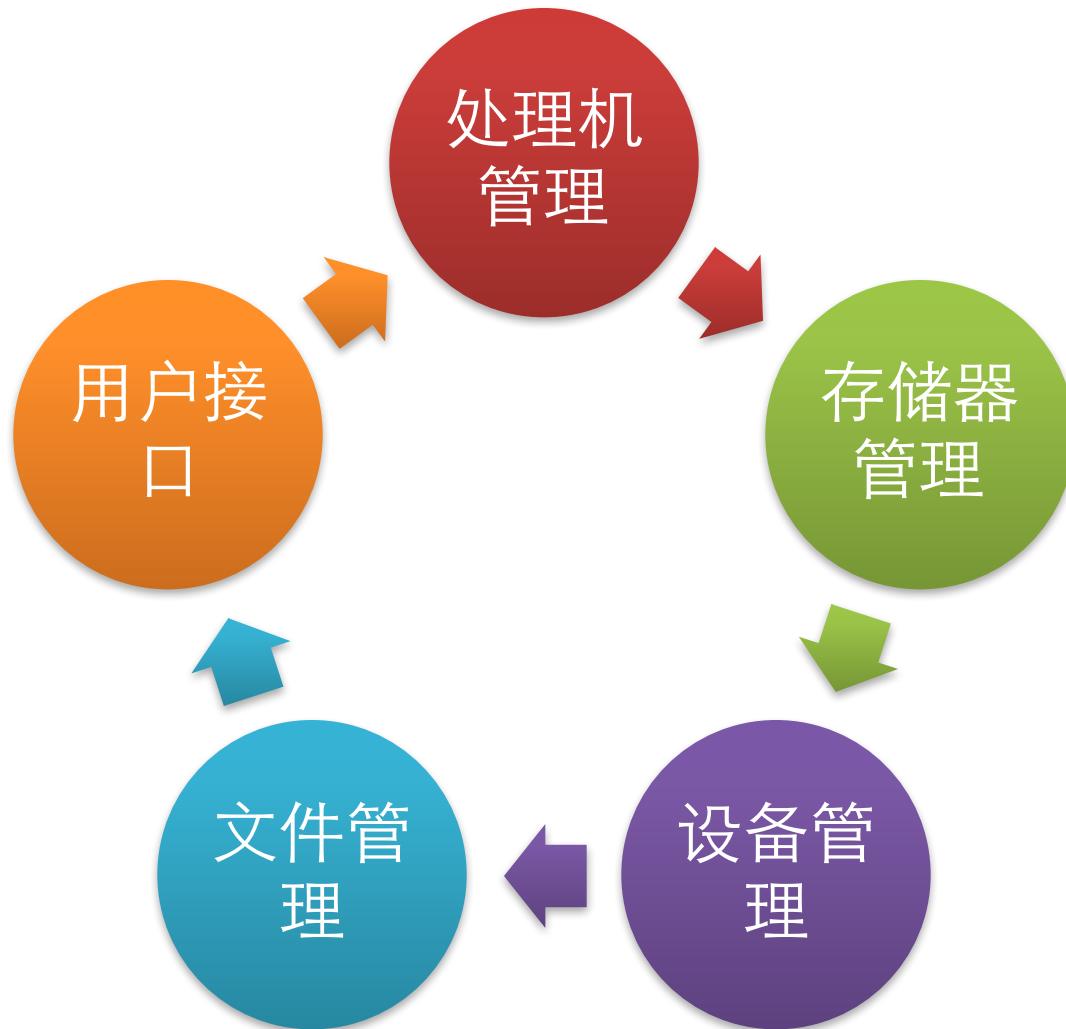
可扩展性

- 可开发、测试引入新功能

开放性

- 应用程序的可移植和互操作

操作系统的主要功能



处理机管理功能

- 按照一定的算法把处理机分配给进程（线程），并对其进行有效的管理和控制。



存储器管理功能

- 为多道程序提供运行环境，方便使用，提高存储器利用率以及能从逻辑上扩充内存。



设备管理功能

- 完成用户提出的I/O请求；为用户分配其所需的I/O设备；提高CPU和I/O设备的利用率；提高I/O速度；方便用户使用I/O设备。



文件管理功能

- 对用户文件和系统文件进行管理，以方便用户使用，并保证文件的安全性。



用户接口

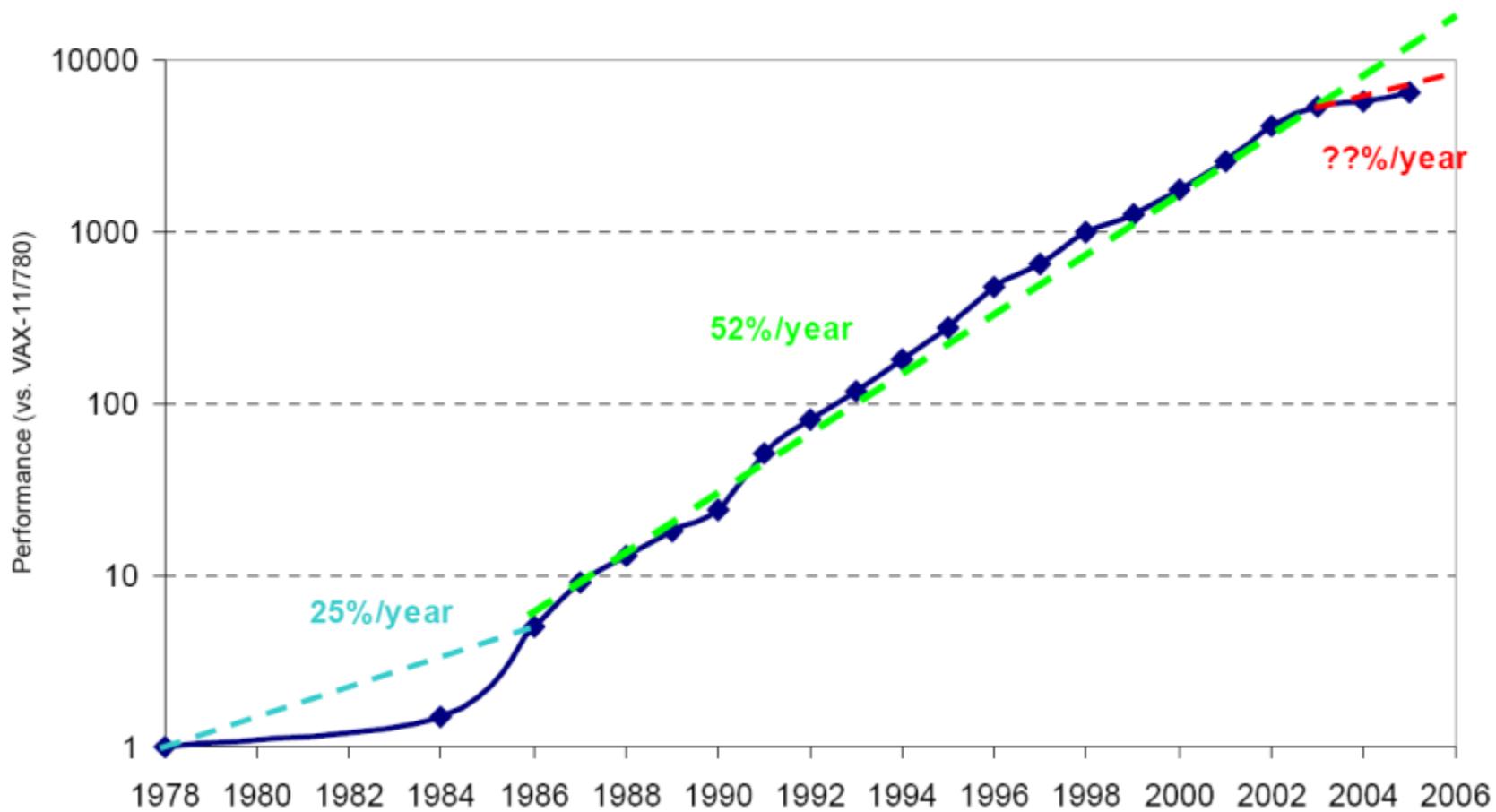
- OS提供给用户交互的命令集合
- 用户操作的方式：
 - 命令行接口CLI: Command Line Interface
 - 图形用户接口GUI: Graphic User Interface
- 应用程序编程接口
 - Application Programming Interface, API
 - 应用软件需要的系统调用 (System Call)

操作系统的形成与发展

- 操作系统能够不断发展，其原因：
 - 硬件升级和新型硬件的出现
 - 系统设计的平衡 (trade-off) 随着技术变化而变化
 - 新的用户需求

	1981	2006	比例
MIPS	1	3,600	3,600
DRAM	128KB	1GB	50,000
磁盘	10MB	500GB	51,200
网络带宽	9.6Kb/s	1Gb/s	100,000
地址空间	16位	64位	2^{48}
每电脑用户数	100	<1	100

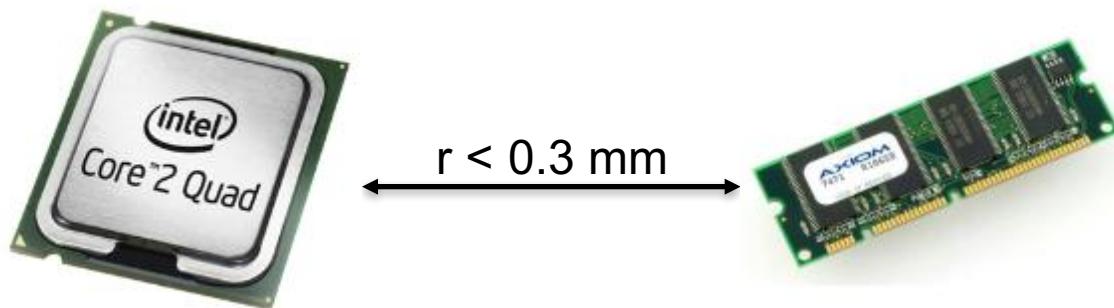
摩尔定律：CPU性能



From “Computer Architecture: A Quantitative Approach”, 4th edition, 2007

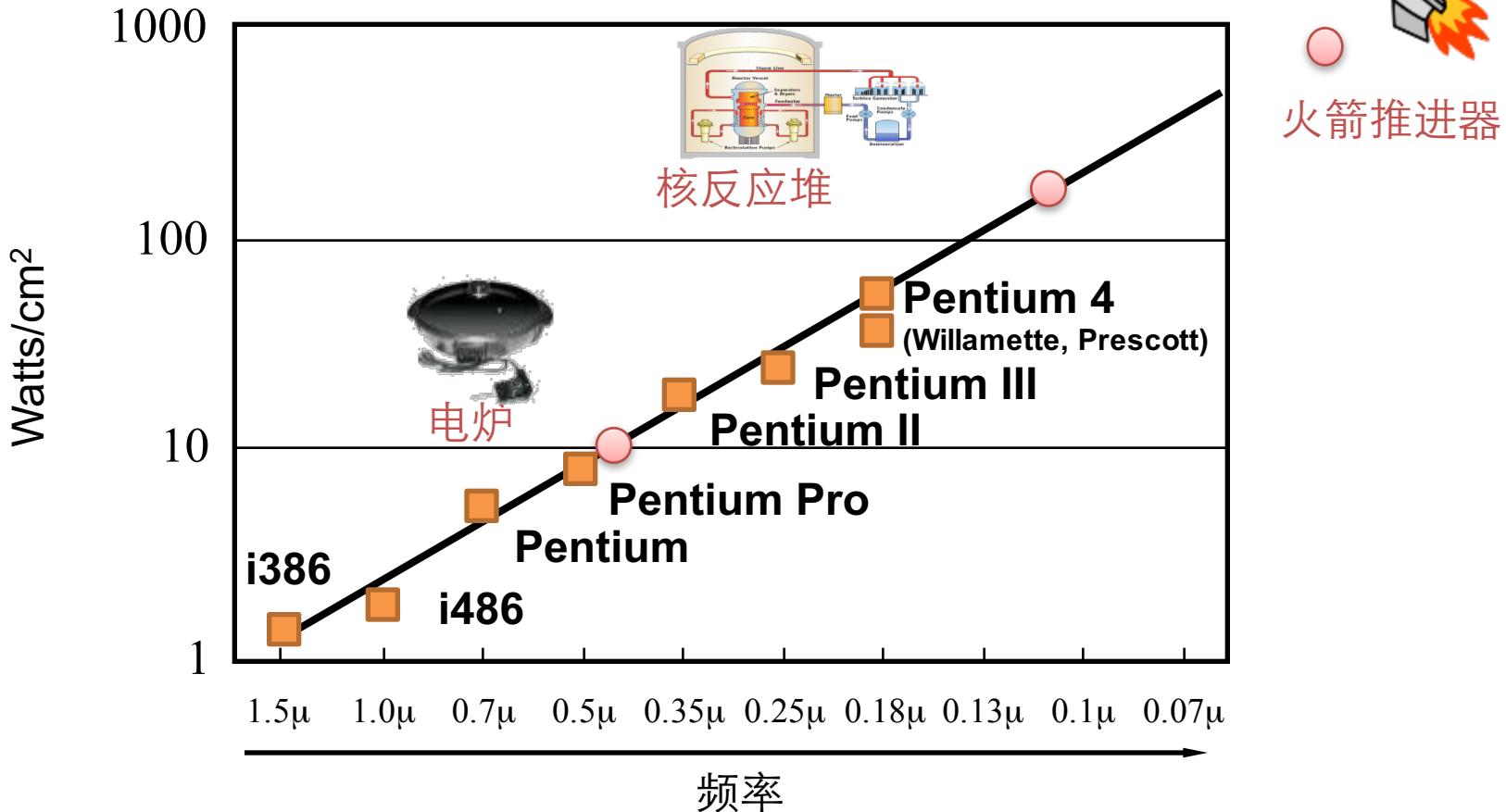
物理距离

- 开发1 Tflop/s的计算机
 - r : 内存→CPU数据传输距离
 - 每周期取1个单位数据→1s取 10^{12} 次
 - $c = 3 \times 10^8 \text{ m/s}$
 - $r < c/10^{12} = 0.3 \text{ mm}$

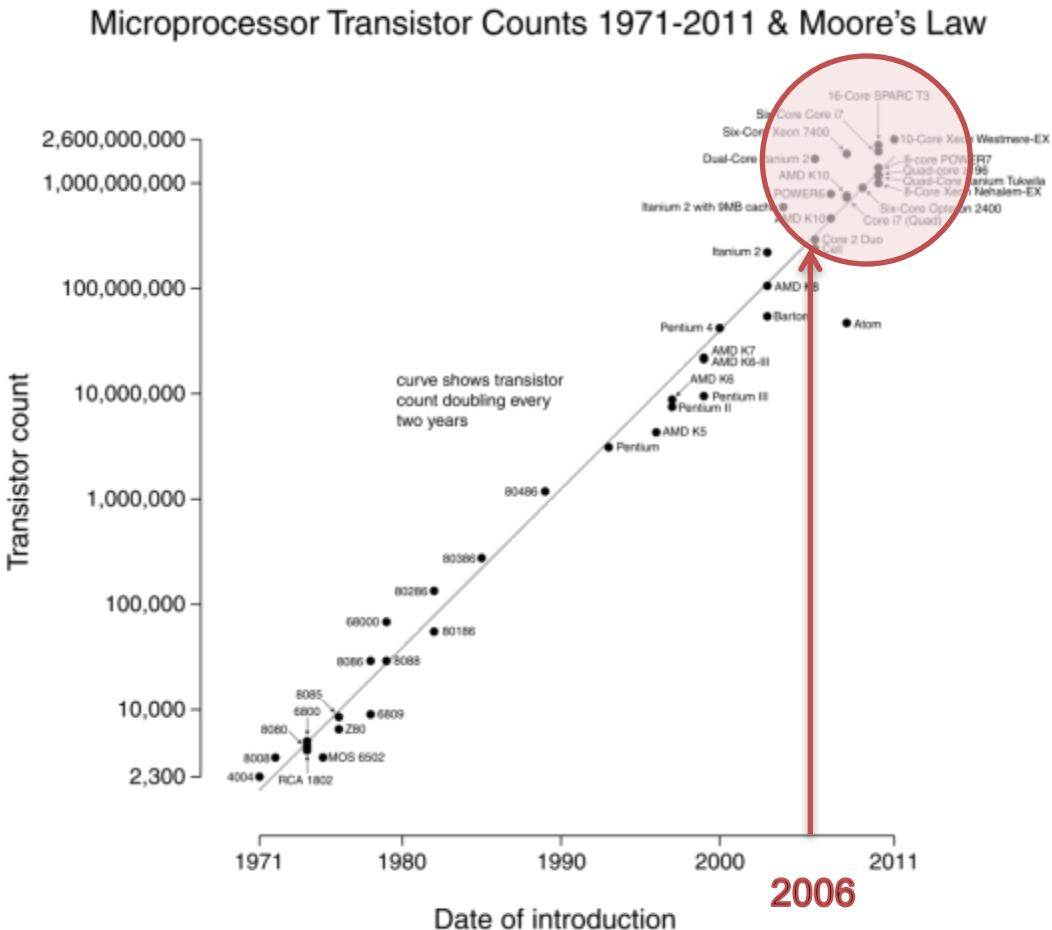


功耗

耗散功耗 $\sim CV^2f$



晶体管数量



Source: Wikipedia: http://en.wikipedia.org/wiki/Moore's_law

操作系统的发展

电子管
('46-'55)

晶体管
('55-'65)

集成电路
('65-'80)

超大规模
集成电路
('80-)

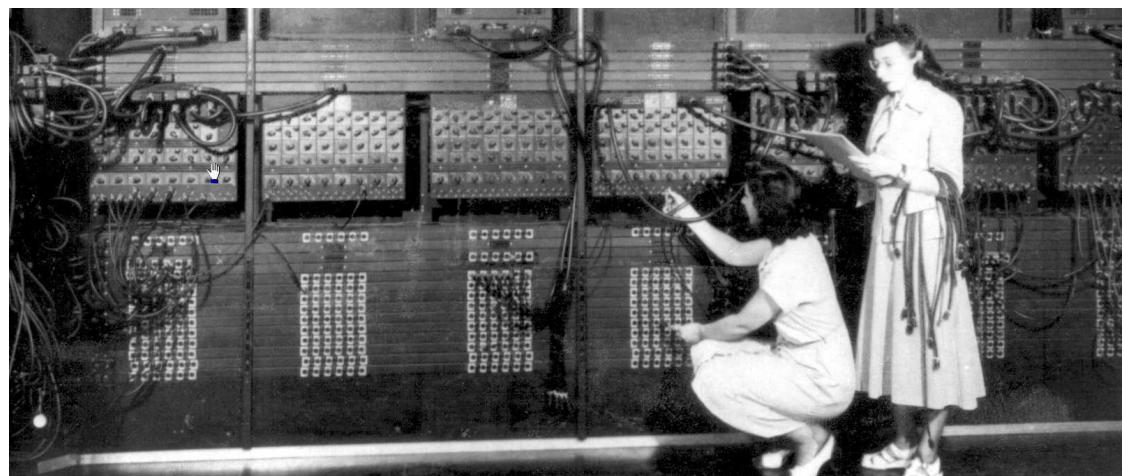
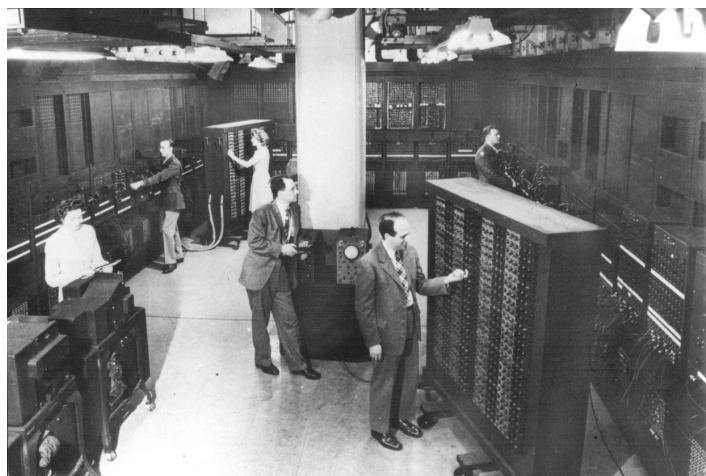
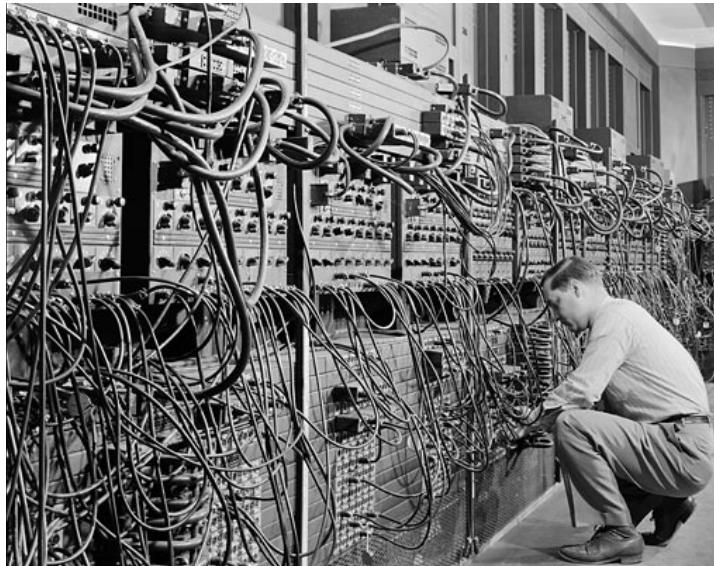
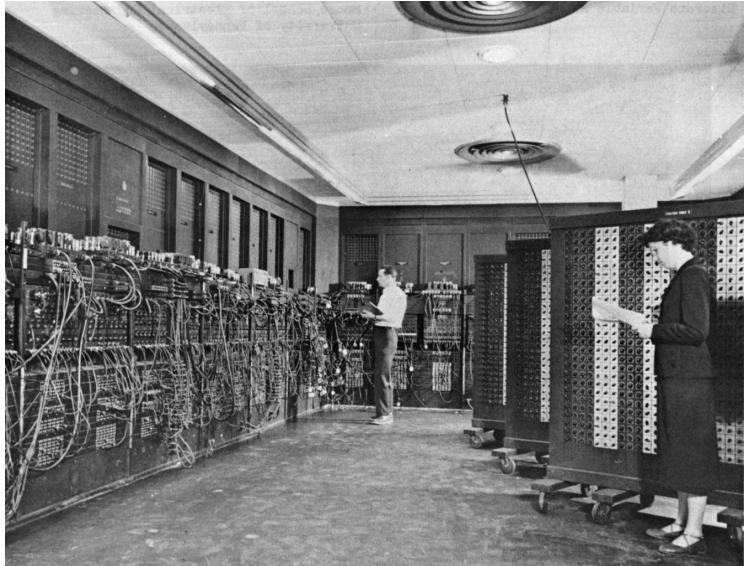
操作系统的发展

- 单用户系统（无操作系统）
- 批处理系统
 - 单道批处理系统
 - 多道批处理系统
- 分时系统
- 个人计算机
- 分布式计算系统
- 互联网时代（移动设备）
- http://en.wikipedia.org/wiki/Operating_system

单用户系统/无操作系统('45-'55)

- 人工操作方式
 - 一台计算机的所有资源由用户独占，降低了计算机资源利用率，人操作慢，出现了严重的人机矛盾。
- 脱机输入输出方式
 - 在外围计算机的控制下，实现输入输出。

世界第一台计算机



ENIAC@1946.2.14



- Electronic Numerical Integrator And Computer (电子数值积分计算机)
- 美国宾夕法尼亚大学，18000个电子管、1500个继电器及其它器件，约90立方米，30吨，170平方米，运算速度为每秒5000次加法或400次乘法。
- 由于耗电量大，据传ENIAC每次一开机，整个费城西区的电灯都为之黯然失色。

人物

- John Presper Eckert
(埃克特, 1919-1995)
- John William Mauchly
(莫奇利, 1907-1980)



历史上的预测

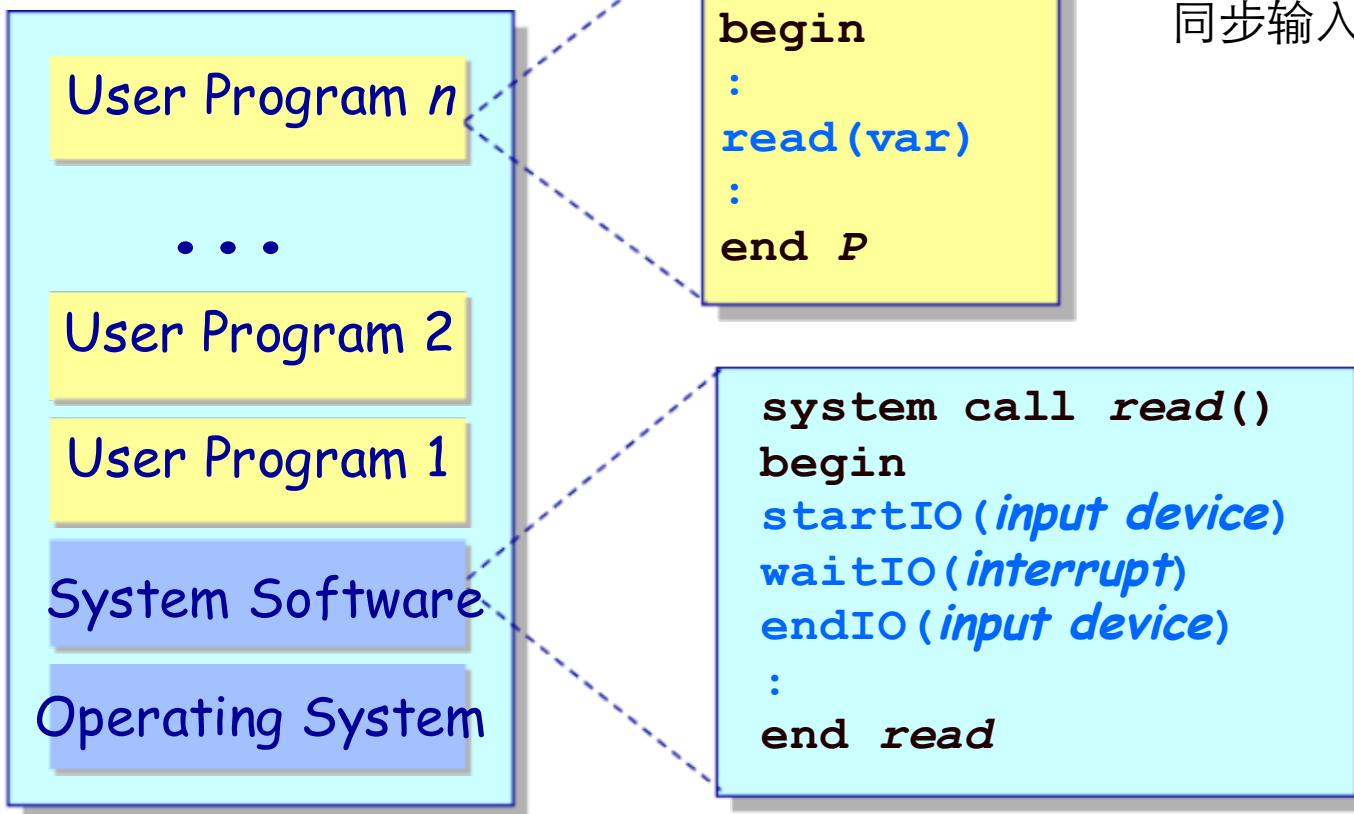
- 未来计算机的重量不会超过1.5吨。
——流行机械杂志，1949
- 我认为全世界只需要5台计算机。
——托马斯·沃特森，IBM董事会主席，1943（疑为误传）
- 没有理由让每个人在家里都安装一台计算机。
——肯·沃里森，DEC公司总裁，董事会主席，1977
- 无论对谁来说，640K内存已经足够。
——比尔·盖茨，1981（系误传）

单道批处理系统('55-'65)

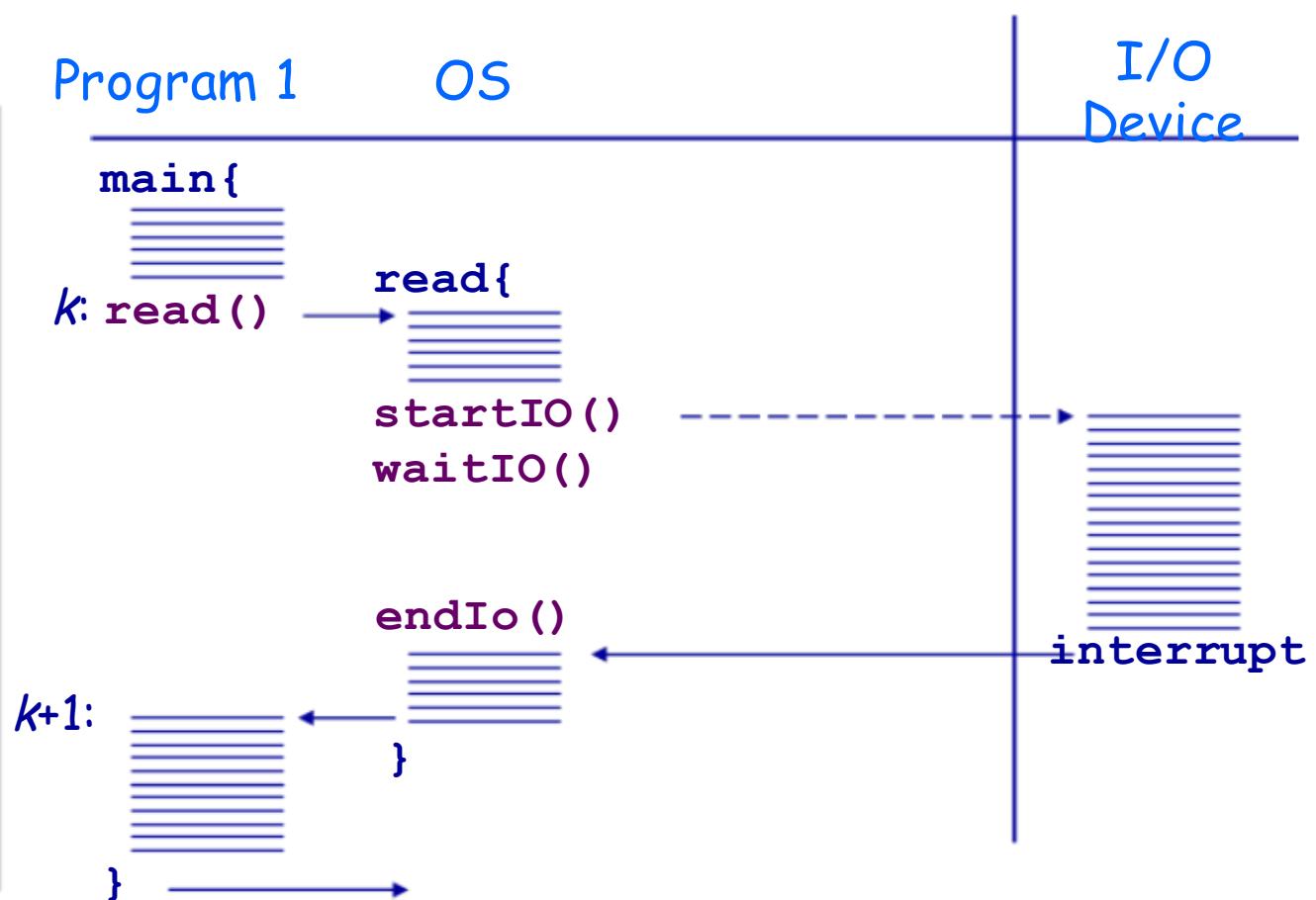
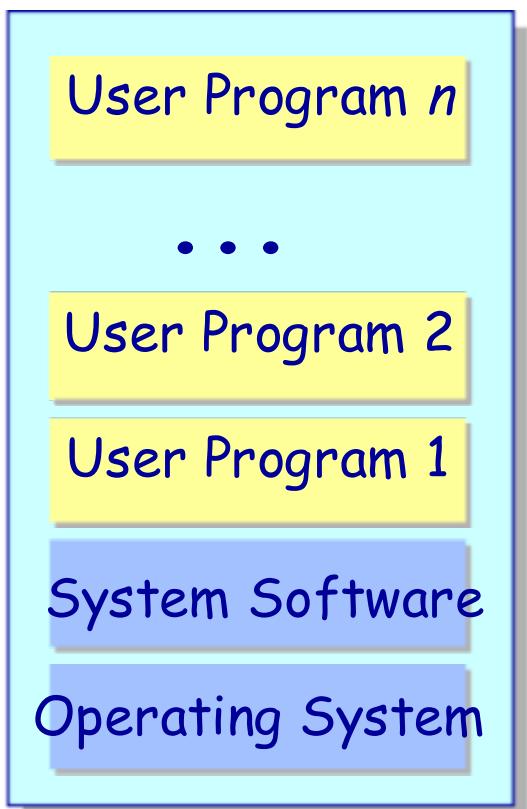
- 批处理：Batch Processing
- 在内存中仅存一道作业运行，运行结束或出错，才自动调另一道作业（磁带中）运行。
- 主要特征：自动性、顺序性、单道性。
- 优点：减少人工操作，解决了作业的自动接续。
- 缺点：平均周转时间长，没有交互能力。

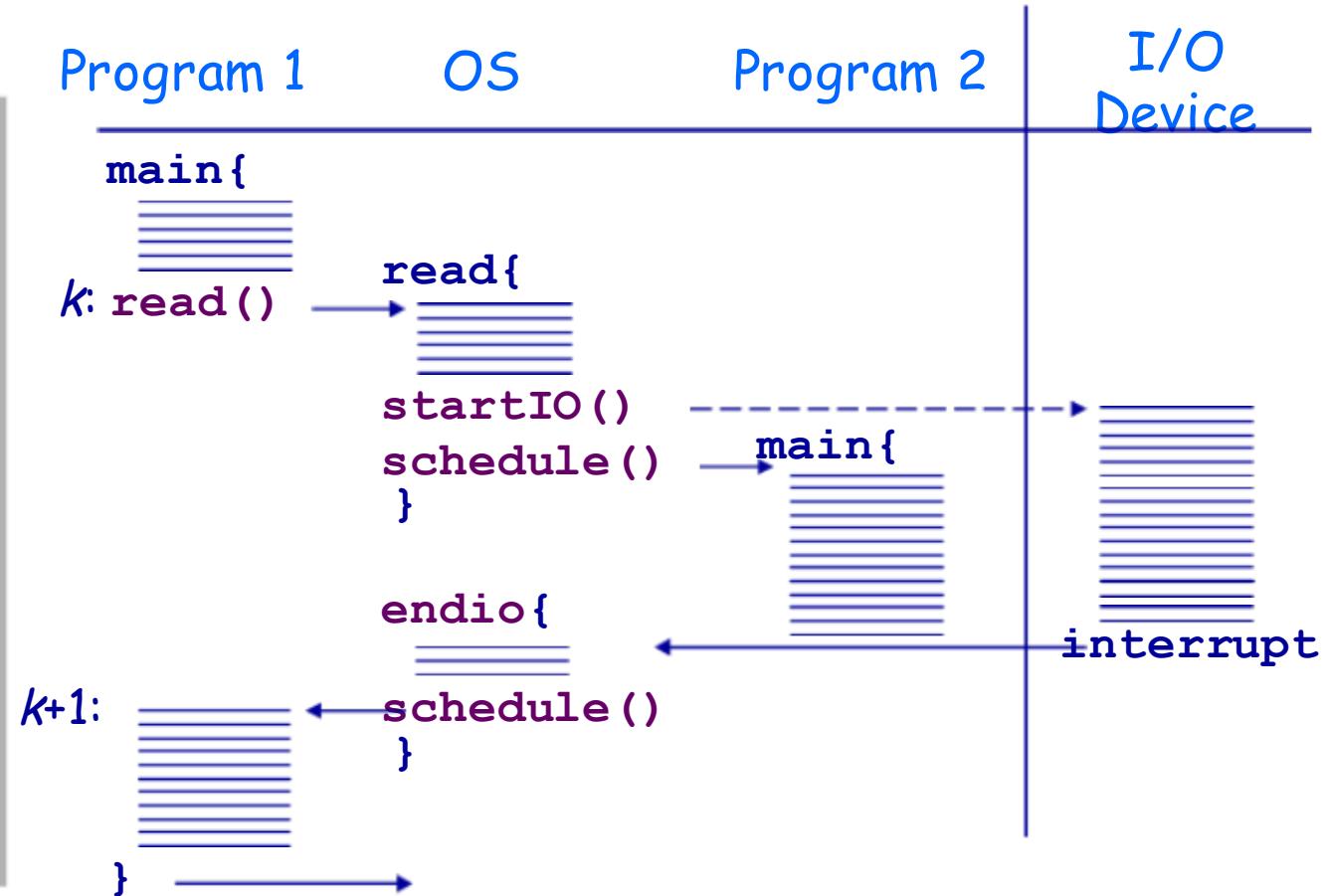
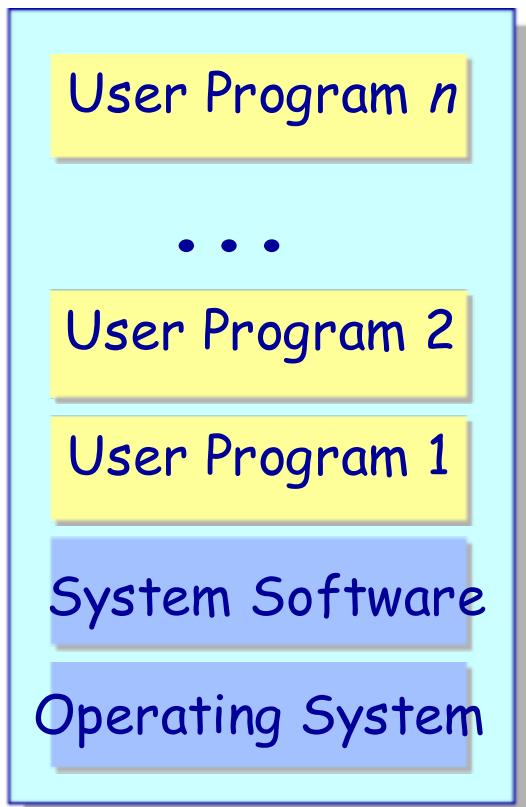
多道批处理系统('65-'80)

- 在内存中存放多道作业运行，I/O访问、运行结束或出错，自动调度内存中的另一道作业运行。
- 好处：
 - 提高CPU的利用率。
 - 提高内存和I/O设备利用率。
 - 增加系统吞吐率。



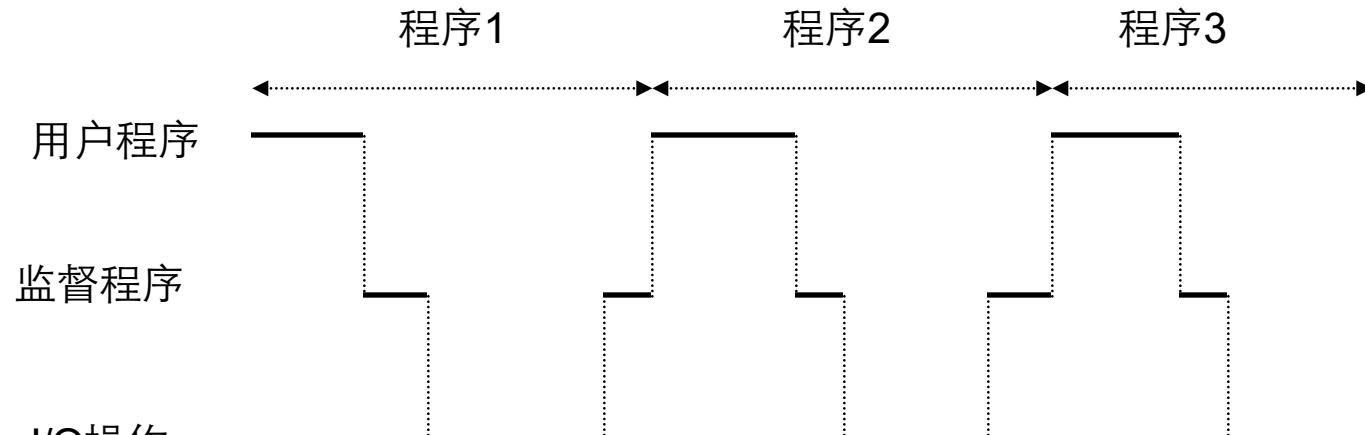
同步输入，等待I/O



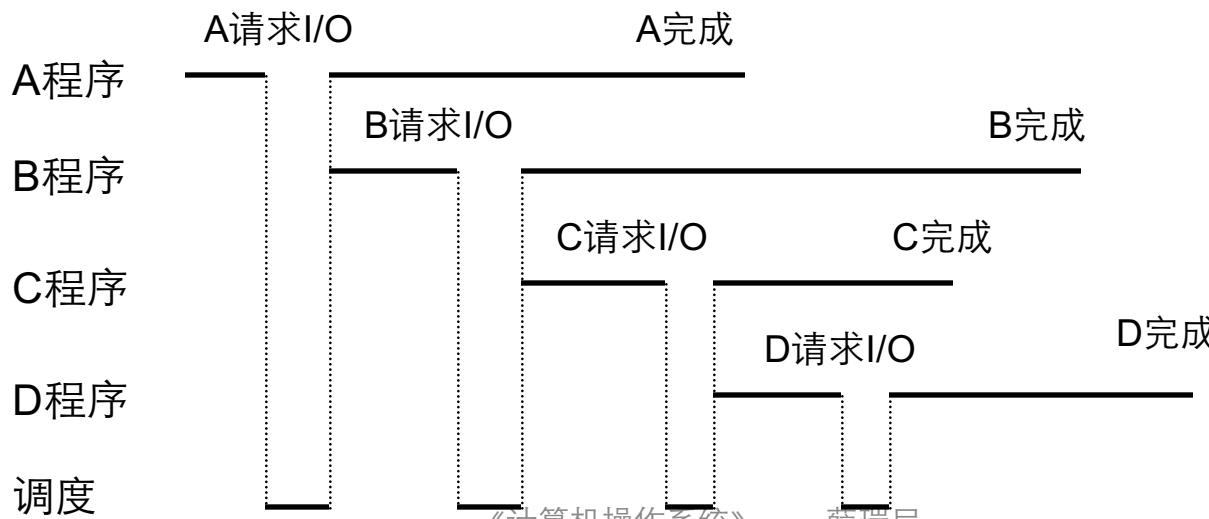


举例：多道执行情况

单道执行



多道执行



多道批处理系统主要特征

- 多道性、无序性、调度性（进程调度和作业调度）
- 优点：提高了资源利用率和吞吐能力。
- 缺点：没有交互能力。

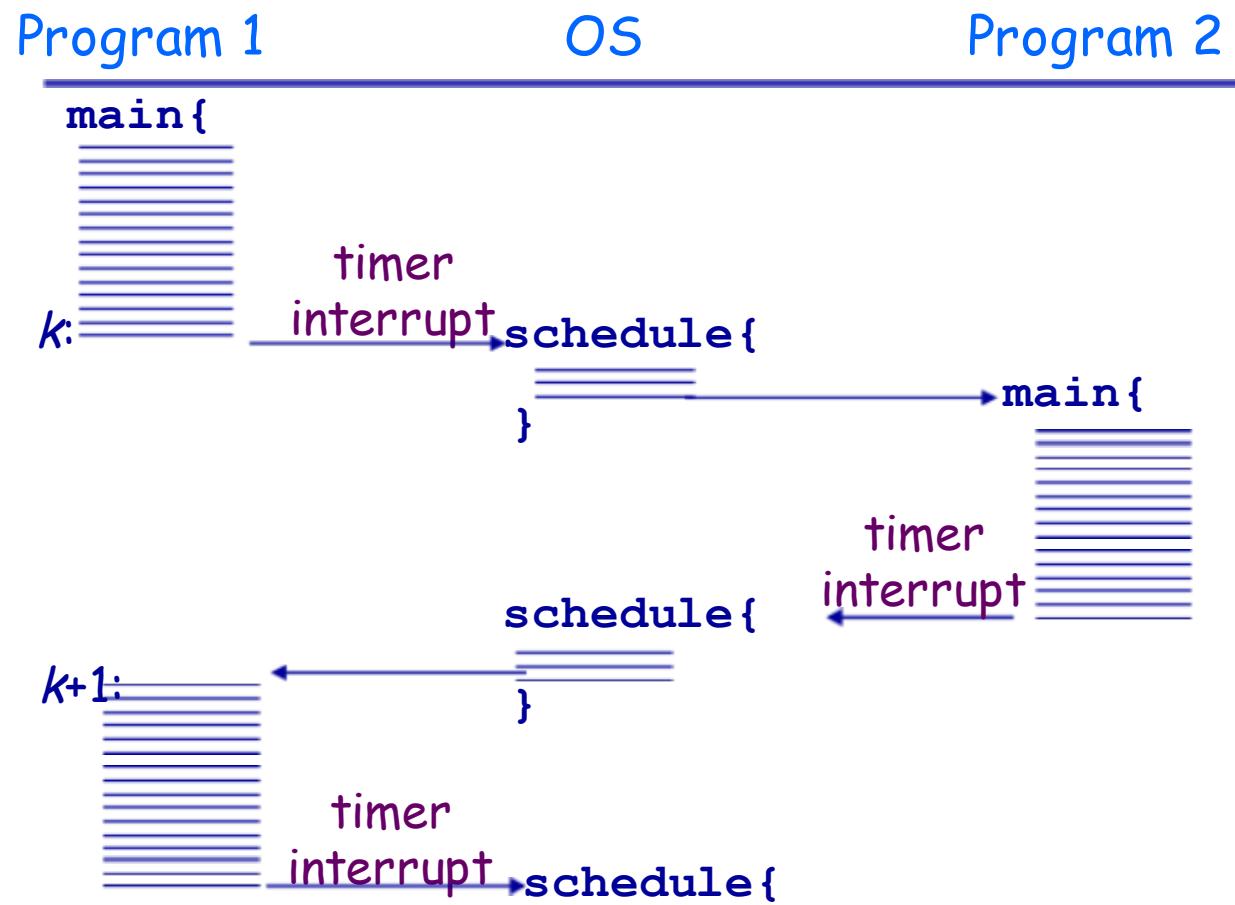
多道批处理系统需要解决的问题

- 处理机管理：分配和控制CPU
- 存储器管理：内存分配与回收
- I/O设备管理：I/O设备的分配与操纵
- 文件管理：文件的存取、共享和保护
- 作业管理：如何组织作业运行

分时系统('70-)——Timesharing

- 分时系统的产生
 - 用户需要：人机交互、共享主机、便于用户上机
- 分时系统实现的方法
 - 时钟中断(interrupt)：时间片(time slice)

时钟中断触发CPU切换



分时系统的特征

- 多路性：多个用户分时使用一台计算机。
- 独立性：独立运行，不混淆，不破坏。
- 及时性：系统能在很短的时间应答。
- 交互性：能实现人机对话。

个人电脑('80-): Personal Computer

- 个人电脑
 - 单用户
 - 利用率不是关键考虑
 - 关注用户界面和接口
 - 没有服务



分布式系统('90-)——Distributed System

- 支持分布式服务
- 系统之间数据共享
- 多处理器 (SMP)
- 高可用性和可靠性



中国天河

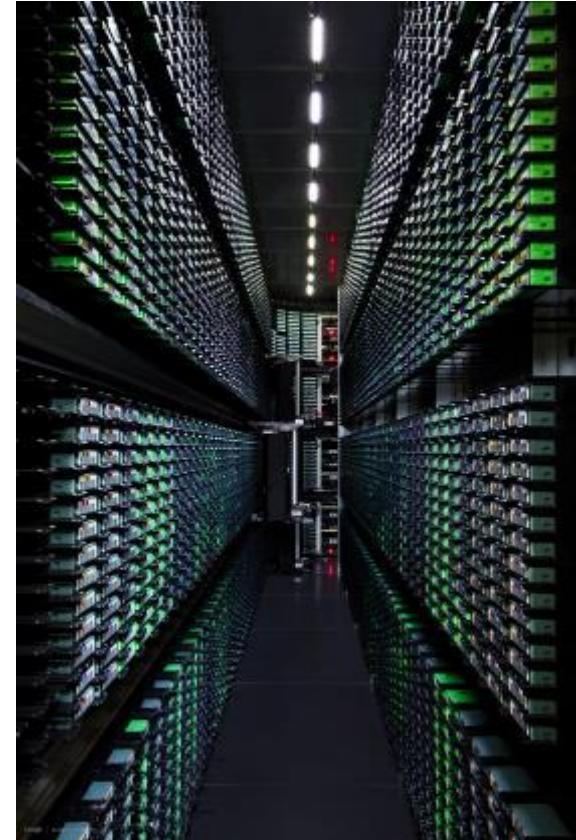


IBM Blue Gene/P



日本地球模拟器

互联网时代 ('05-) : Google数据中心



移动计算时代 ('10-) : 3G+

- 移动设备（手机，平板.....）
- 物联网（传感器）



Jawbone UP



Google Glass



Nike Fuelband



Fitbit腕带



Fitbit计步器

人物



Bill Gates



Jeff Dean

实时系统——Realtime System

- 计算机及时响应外部事件的请求，在规定的时间内完成对该事件的处理，并控制所有实时设备和实时任务协调一致的运行。
 - 航空航天
 - 军事
 - 工业控制



实时系统的特征

- 多路性：能对多个对象进行控制。
- 独立性：独立运行，不混淆，不破坏。
- 交互性：仅限于访问系统中某些特定的专用服务程序。
- 可靠性：高可靠性，应具有过载防护能力。
- 及时性：不同的系统要求不一样，控制对象必须在截止时间内完成。

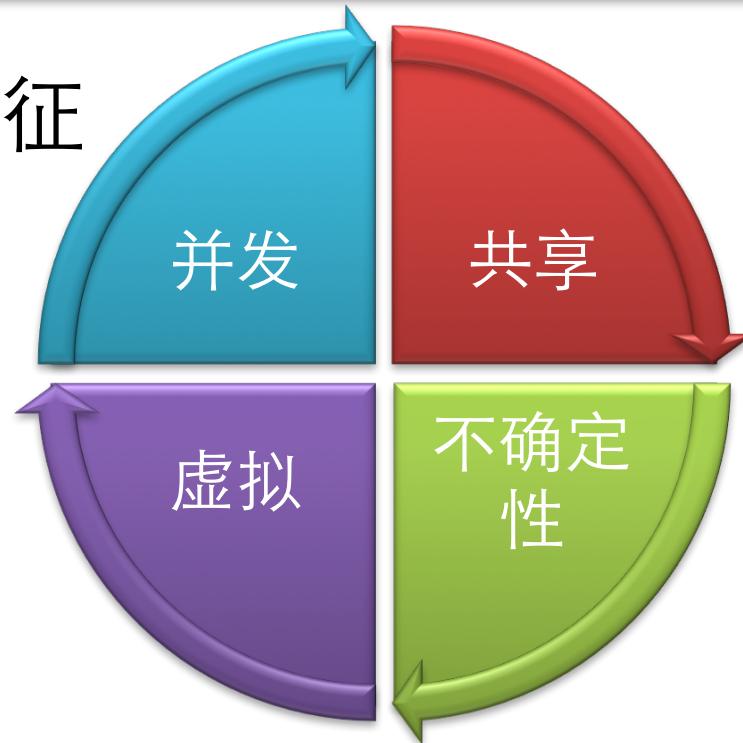
手机操作系统

- MediaTek (联发科) : 山寨鼻祖
 - Nucleus OS
- 匹夫无罪，怀璧其罪
 - 最完美的All-in-one解决方案



操作系统的基本特征

- 现代OS的四个基本特征
 - concurrency
 - sharing
 - virtualization
 - non-determinism
- 并发是最重要的特征，其它特征**基本都**以并发为前提。



同时性— Concurrency/Parallelism

- 并行性是指两个或多个事件在**同一时刻**发生。
- 并发性是指两个或多个事件在**同一时间间隔内**发生。
- 任务共行
 - 从宏观上看，指系统中有多个任务同时运行
 - 从微观上看
 - 单处理机系统中的任务并发（Task Concurrency）：多个任务在单个处理机上交替运行；
 - 或多处理机系统中的任务并行（Task Parallelism）：多个任务在多个处理机上同时运行）。

共享

- 系统中的资源可供内存中多个并发执行的
进程共同使用。
- 互斥共享方式
 - 把在一段时间内只允许一个进程访问的资源，
称为临界资源。
 - 系统中的临界资源可以提供给多个进程使用，
但一次仅允许一个进程使用，称为互斥共享方
式。

同时访问方式

- 从宏观上看，资源共享是指多个任务可以同时使用系统中的软硬件资源
- 从微观上看，资源共享是指多个任务可以交替互斥地使用系统中的某个资源。例如磁盘。

虚拟

- 通过某种技术把一个物理实体变为若干个逻辑上的对应物
 - 虚拟处理机：分时实现
 - 虚拟设备：SPOOLING技术
 - 虚拟存储器：虚拟存储管理实现

不确定性

- 同样的程序，同样的输入，未必同样的输出
 - 异步性——是指进程以异步的方式执行，进程是以不可预知的速度向前推进。
 - 同样的执行顺序也会产生不确定性
 - random()
 - gettimeofday()

操作系统的体系结构

- 操作系统是一个大型系统软件，其结构已经历了四代的变革：
 - 第一代：无结构；
 - 第二代：模块式结构；
 - 第三代：层次式结构；
 - 第四代：把工程学引入到软件开发的过程中，从而形成了软件工程学

传统的操作系统结构

- 操作系统中增加了越来越多的功能，并且随着底层硬件更高的性能，更加通用，操作系统的大小和复杂性也随着增加。
- 为了控制该软件的复杂性，在开发OS时，先后引入了分解、模块化、抽象和隐蔽等方法。
- 开发方法的不断发展，促进了OS结构的更新换代。

1、无结构操作系统

- 在早期开发操作系统时，设计者只是把他的注意力放在功能的实现和获得高的效率上，缺乏首尾一致的设计思想。
- OS是众多过程的集合，各过程之间可以相互调用，在操作系统内部不存在任何结构，因此，有人把它称为整体系统结构。
- 设计出的操作系统既庞大又杂乱，缺乏清晰的程序结构。
- 编制出的程序错误很多，给调试工作带来很多困难；增加了维护人员的负担。

2、模块化OS结构

- 使用分块结构的系统包含若干module（模块）；其中，每一块实现一组基本概念以及与其相关的基本属性。
- 块与块之间的相互关系：
 - 所有各块的实现均可以任意引用其它各块所提供的概念及属性。

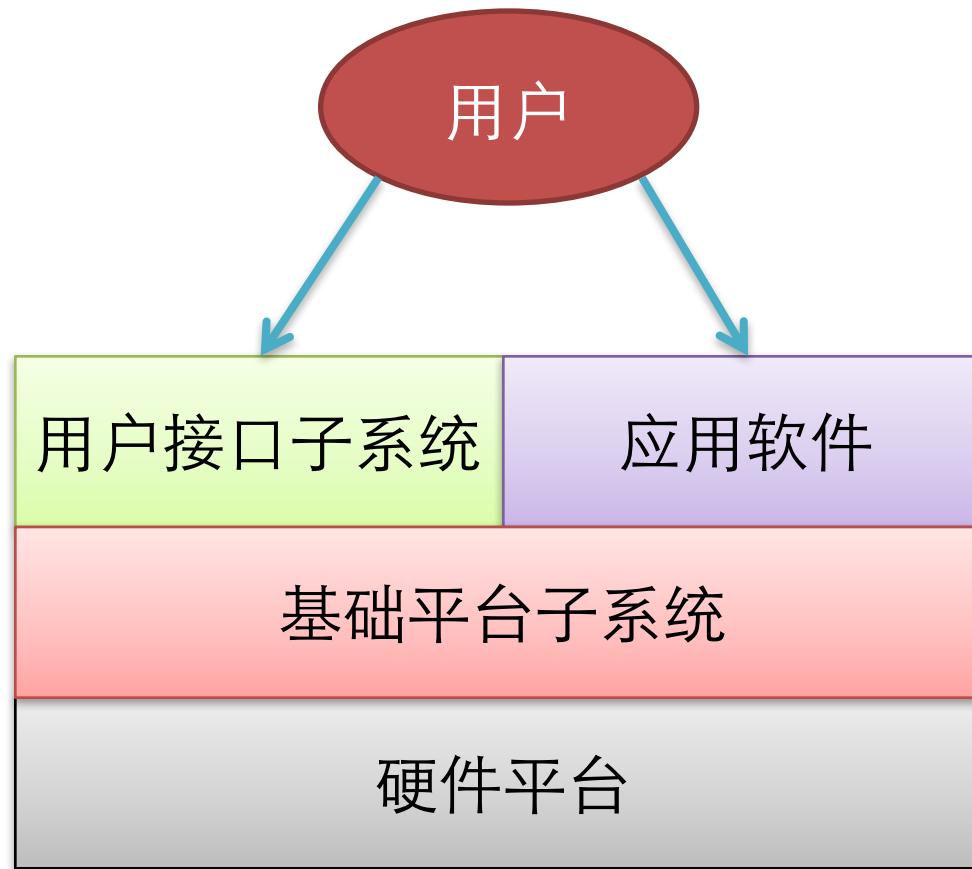
模块化OS的优缺点

- 优点：
 - 提高了OS设计的正确性、可理解性和可维护性。
 - 增强了OS的可适应性。
 - 加速了OS的开发过程。
- 缺点：
 - 对模块的划分及对接口的规定要精确描述很困难。
 - 从功能观点来划分模块时，未能将共享资源和独占资源加以区别；

3、分层式OS结构

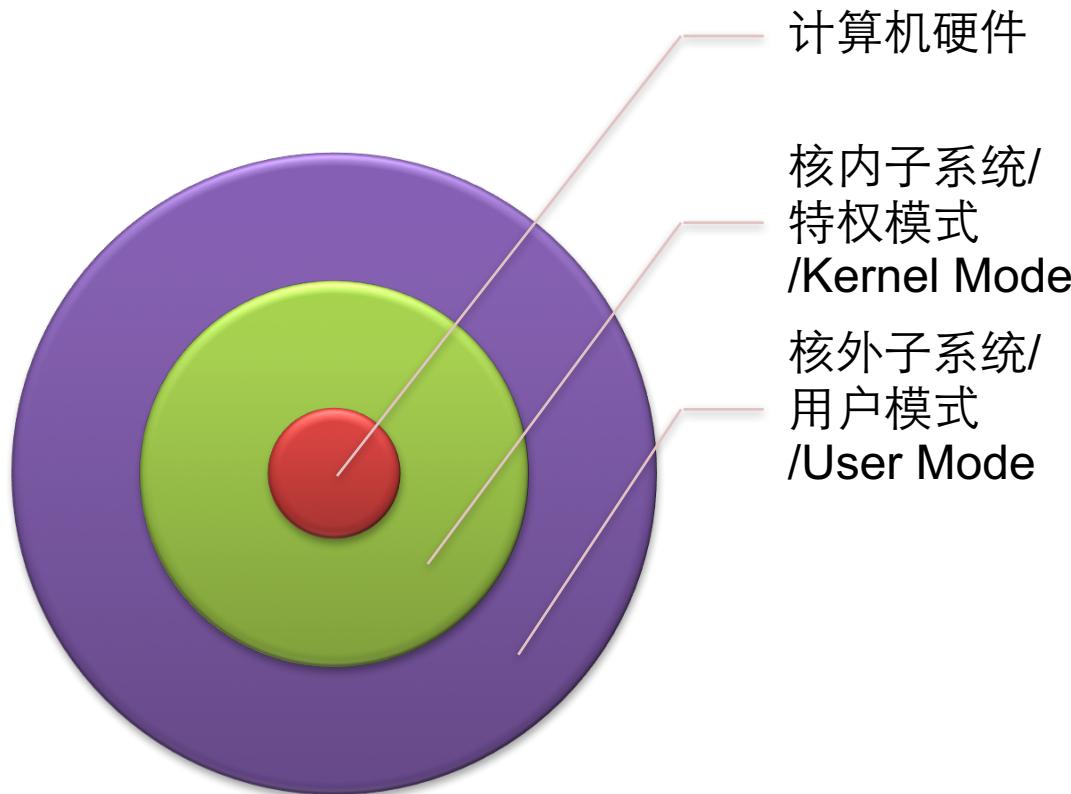
- 使用分层系统结构包含若干layer（层）；每一层实现一组基本概念以及与其相关的基本属性。
- 层与层之间的相互关系：
 - 各层的实现不依赖其以上各层所提供的概念及其属性，只依赖其直接下层所提供的概念及属性；
 - 每一层均对其上各层隐藏其下各层的存在。

常见OS总体结构



双模式平台系统

- 双模式基础平台子系统其总体结构包含两个模式模块



系统设计原则

- 策略（Policy）：做什么？
- 机制（Mechanism）：怎么做？
- 操作系统设计的基本原则

UNIX设计原则

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs to handle text streams, because that is a universal interface.
 - ——Douglas McIlroy

The Art of Unix Programming

- Rule of Modularity: Write simple parts connected by clean interfaces.
- Rule of Clarity: Clarity is better than cleverness.
- Rule of Composition: Design programs to be connected to other programs.
- Rule of Separation: Separate policy from mechanism; separate interfaces from engines.

The Art of Unix Programming

- Rule of Simplicity: Design for simplicity; add complexity only where you must.
- Rule of Parsimony: Write a big program only when it is clear by demonstration that nothing else will do.
- Rule of Transparency: Design for visibility to make inspection and debugging easier.

The Art of Unix Programming

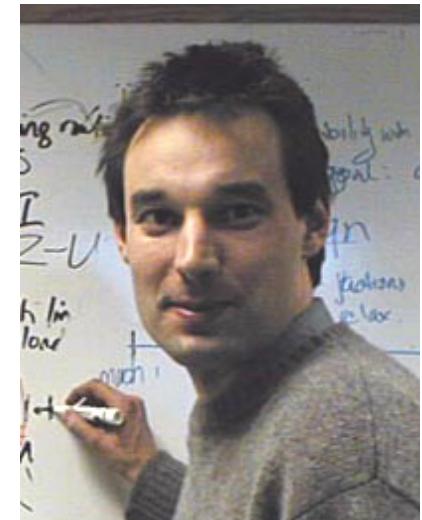
- Rule of Robustness: Robustness is the child of transparency and simplicity.
- Rule of Representation: Fold knowledge into data so program logic can be stupid and robust.
- Rule of Least Surprise: In interface design, always do the least surprising thing.
- Rule of Silence: When a program has nothing surprising to say, it should say nothing.
- Rule of Repair: When you must fail, fail noisily and as soon as possible.

The Art of Unix Programming

- Rule of Economy: Programmer time is expensive; conserve it in preference to machine time.
- Rule of Generation: Avoid hand-hacking; write programs to write programs when you can.
- Rule of Optimization: Prototype before polishing. Get it working before you optimize it.
- Rule of Diversity: Distrust all claims for “one true way”.
- Rule of Extensibility: Design for the future, because it will be here sooner than you think.

推荐读物

- Raymond, Eric S. *The art of Unix programming.*
Addison-Wesley Professional, 2003.
- *Principles of Computer System Design: An Introduction.* Jerome H. Saltzer, M. Frans Kaashoek, Morgan Kaufmann, 2009

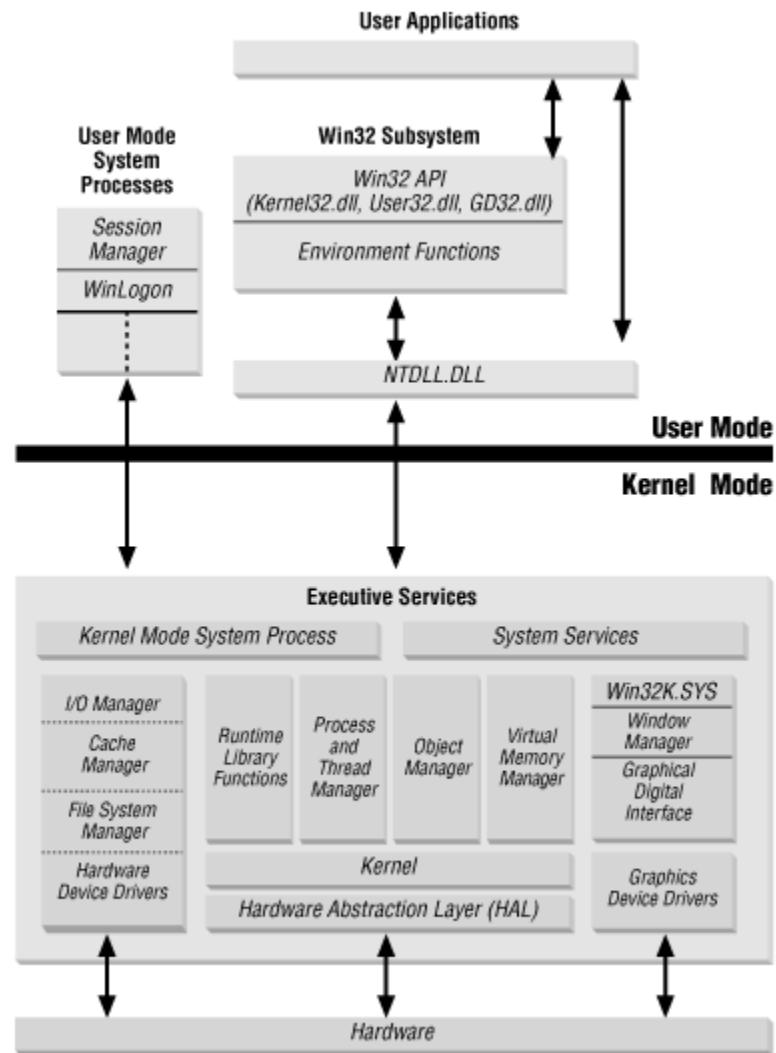


当前主流操作系统简介

- Unix系列
 - Unix/Solaris/FreeBSD/AIX/HP-UX
 - Linux/Android
- Windows
- Mac OS/iOS

Windows

- MS-DOS
- Windows 3.1
- Win 95/95/NT
- Windows Server
- XP
- Vista
- 7, 8, 10



UNIX

- Multics/MIT,BELL,GE@1960
- Unics/Kenneth Thompson & Dennis Ritchie @ Bell Labs, 1969.
- C, 1972
- 唯一因为工程贡献共享获得图灵奖



K. Thompson and D. Ritchie



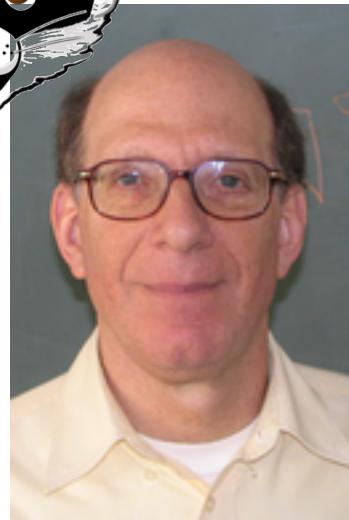
什么是内核？

- 宏内核/单内核 (monolithic kernel)
 - All-in-one
 - Linux, 早期操作系统
- 微内核 (micro kernel)
 - IPC: message passing 性能
 - Mach@CMU, MINIX@Vrije
 - 传奇的L4: Jochen Liedtke
- 现代操作系统：融合



Linux

- Linus Torvalds: 1991



Linus@2000

- Message passing as the fundamental operation of the operating system is just an exercise in computer science masturbation. It may feel good, but you don't actually get anything DONE. Nobody has ever shown that it made sense in the real world.

GNU

- GNU: **GNU is Not Unix**
- GNU/Linux

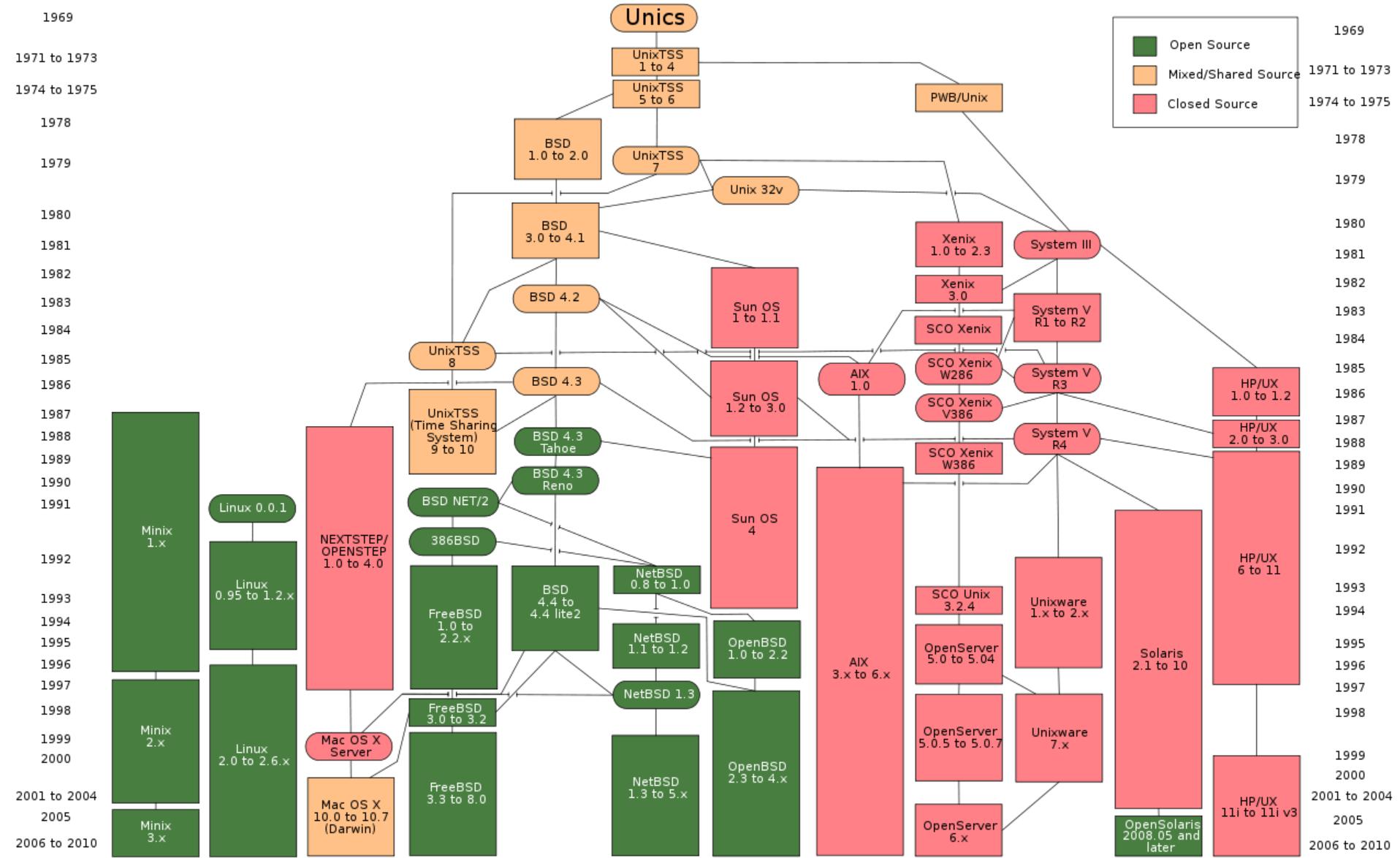


Richard Stallman



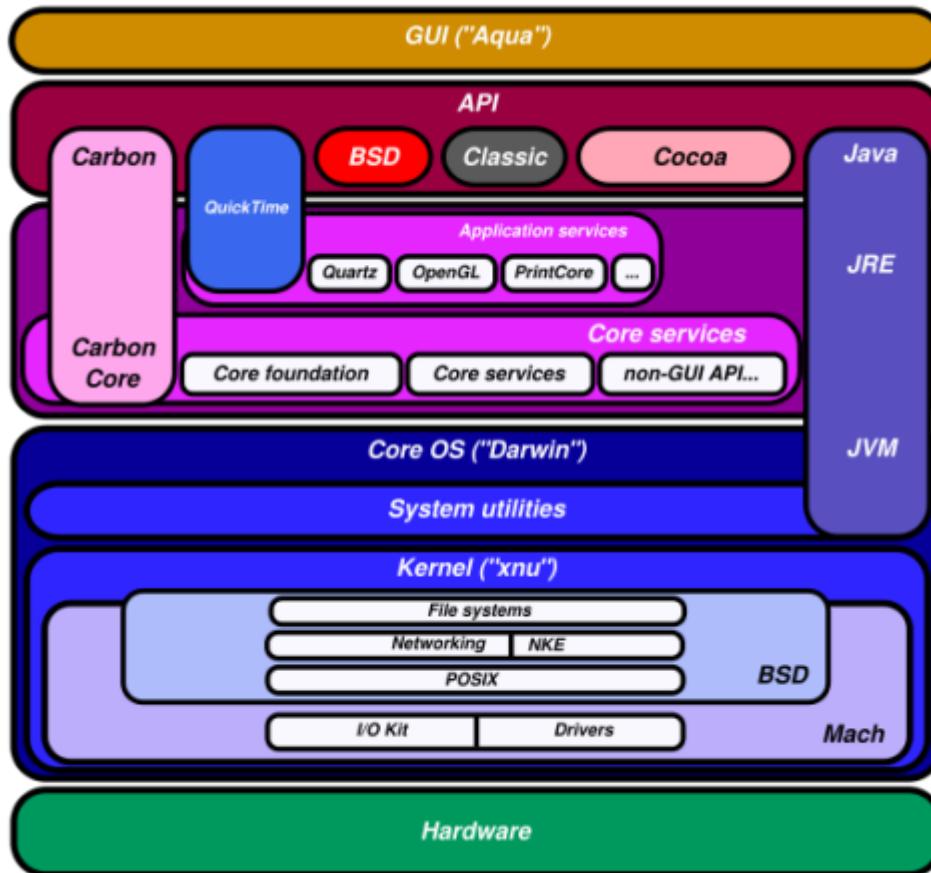
- 路漫漫其修远兮
- Hurd至今不可用
- 但是
 - GCC, GDB, glibc, coreutils, binutils, bash, Emacs, GNOME
- 开创了FSF/OpenSource潮流
- 被“诟病”的： GNU License





Mac OS

- Mach + FreeBSD=>Darwin



Macintosh @ 1984



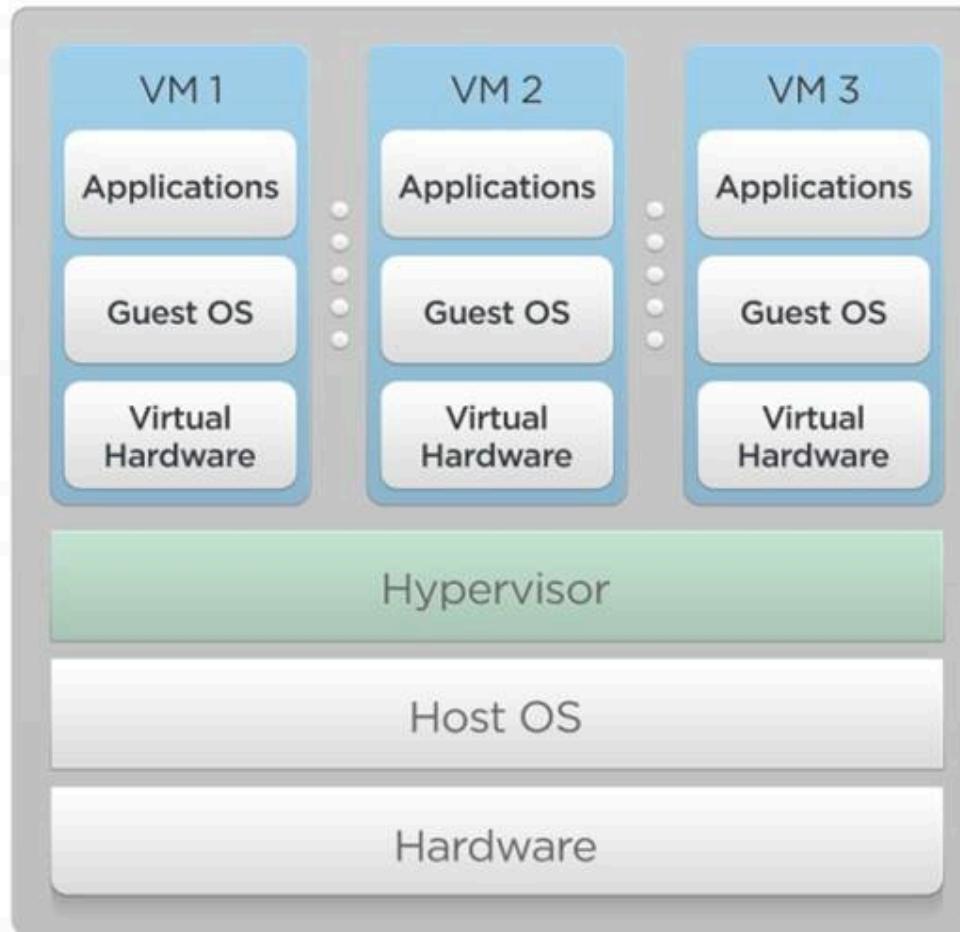
Mac OS X @ 2009

新的抽象

虚拟机 Virtual Machine



Hypervisor



Hypervisor

Type 1 hypervisor

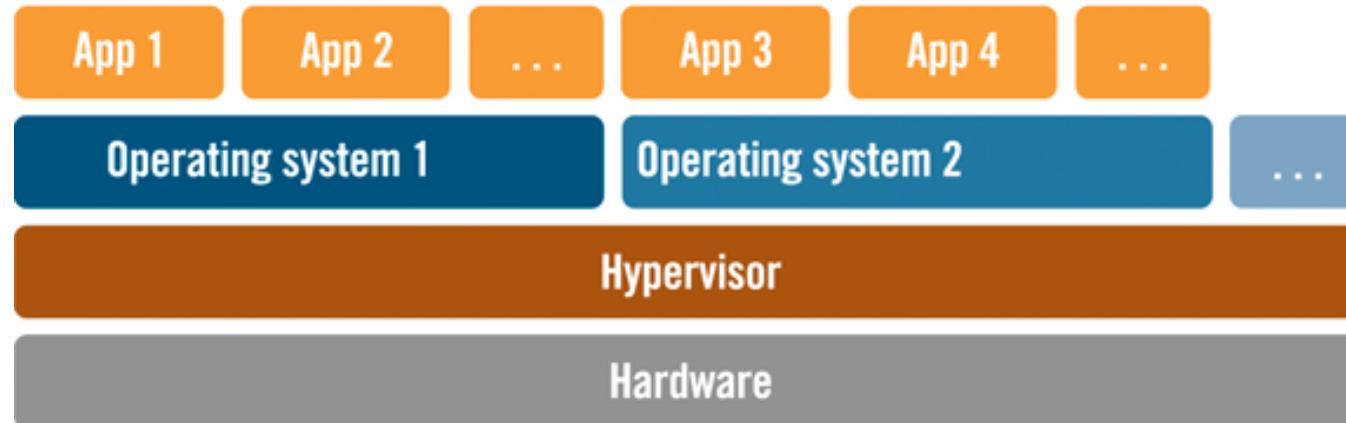
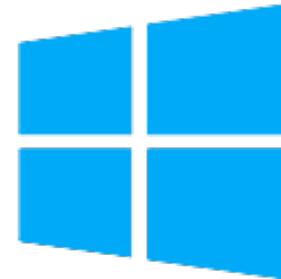


Figure 2. A Type 1 or bare-metal hypervisor sits directly on the host hardware.



Microsoft
Hyper-V

Hypervisor

Type 2 hypervisor

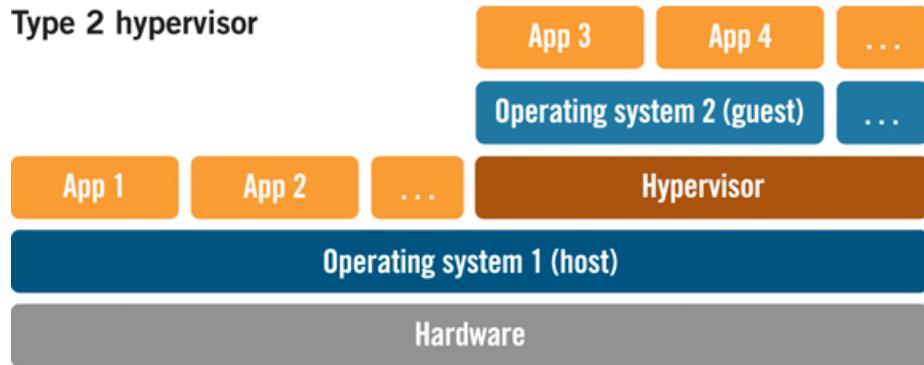


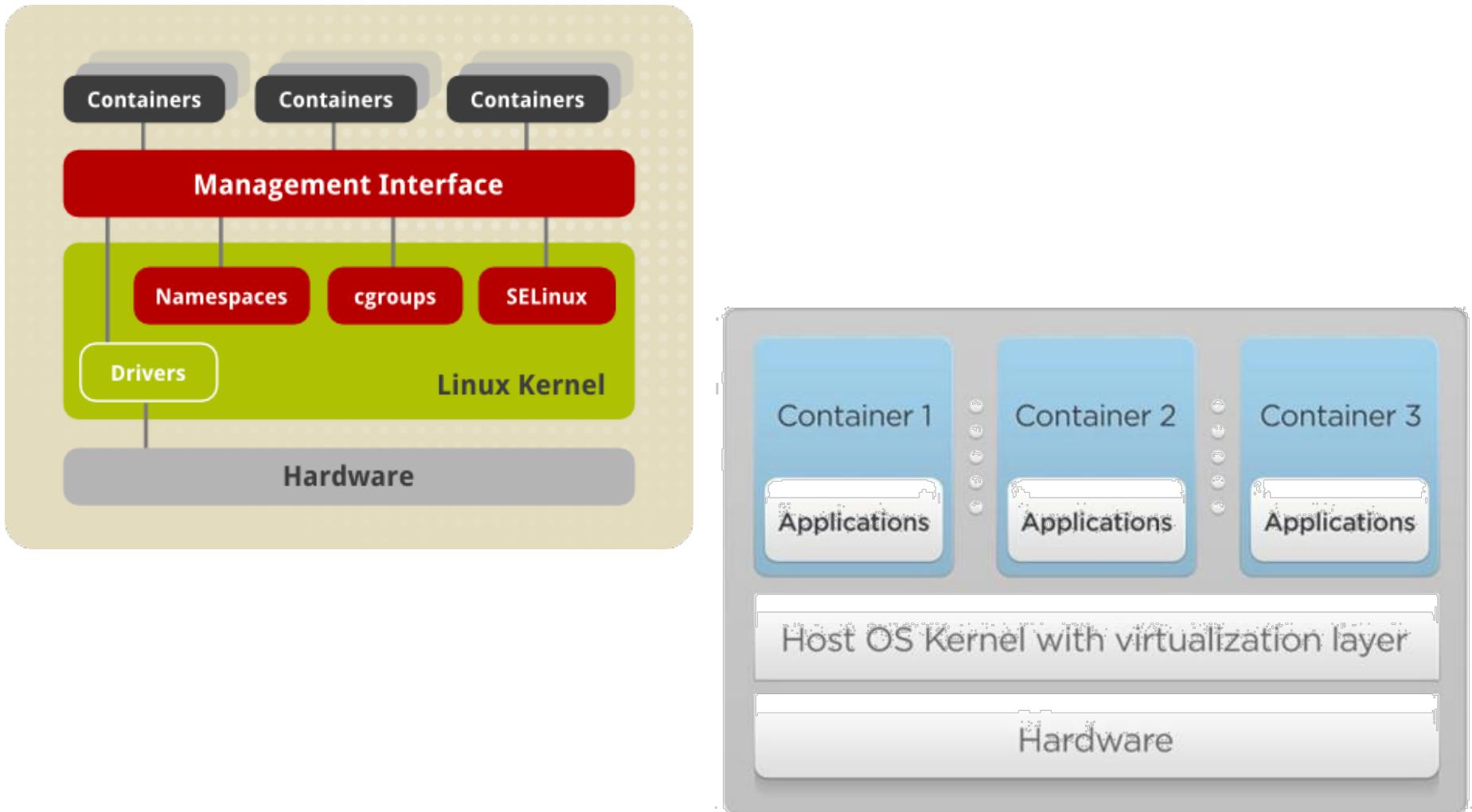
Figure 1. A Type 2 hypervisor runs as an application on a host operating system.



CPU Emulator



State of the art: Container

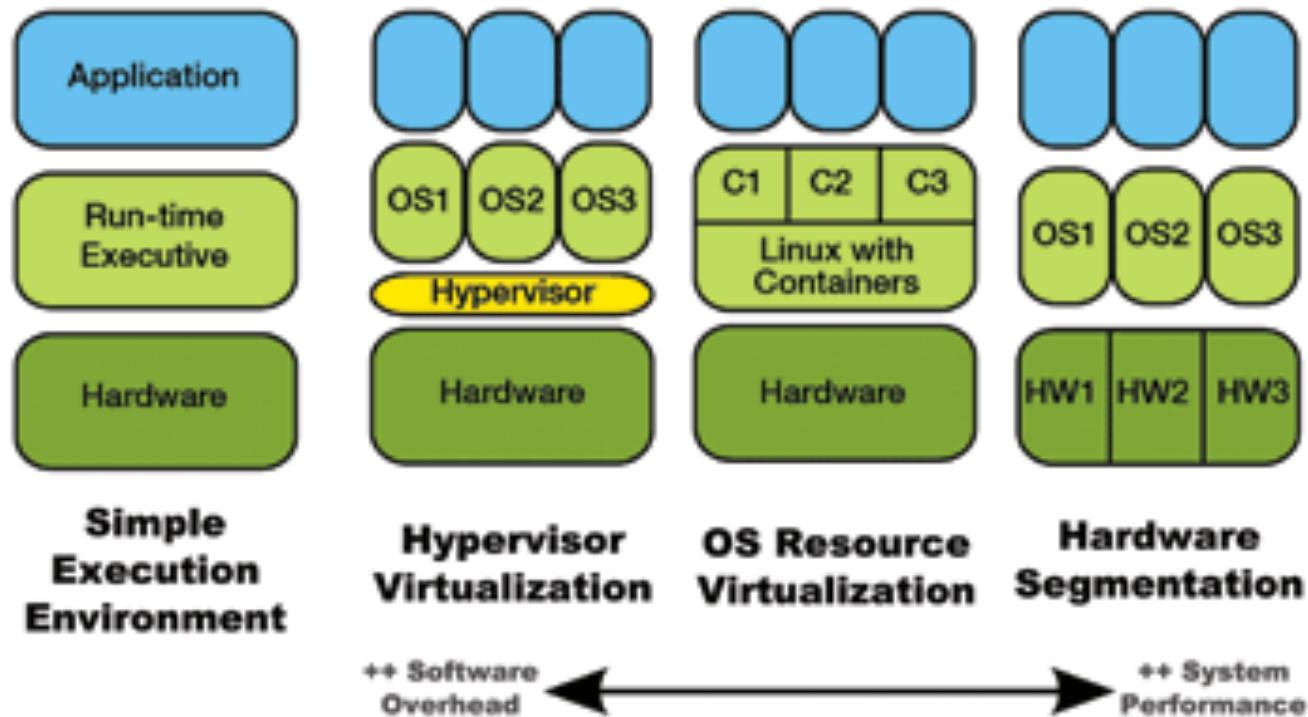


大明星



Core OS

比较



Even more

Unikernels are specialised, single-address-space machine images constructed by using library operating systems.

<https://www.wikiwand.com/en/Unikernel>

2/25/16

《计算机操作系统》——薛瑞尼

DOI:10.1145/2541883.2541895

Article development led by ICPP
quecs.acm.org

What if all the software layers in a virtual appliance were compiled within the same safe, high-level language framework?

BY ANIL MADHAVAPEDDY AND DAVID J. SCOTT

Unikernels: The Rise of the Virtual Library Operating System

CLOUD COMPUTING HAS been pioneering the business of renting computing resources in large data centers to multiple (and possibly competing) tenants. The basic enabling technology for the cloud is operating-system virtualization such as Xen¹ or VMWare, which allows customers to multiplex virtual machines (VMs) on a

shared cluster of physical machines. Each VM presents as a self-contained computer, booting a standard operating-system kernel and running unmodified applications just as if it were executing on a physical machine.

A key driver to the growth of cloud computing in the early days was server consolidation. Existing applications were often installed on physical hosts that were individually underutilized, and virtualization made it feasible to pack them onto fewer hosts without requiring any modifications or code recompilation. VMs are also managed via software APIs rather than physical

actions. They can be centrally backed up and migrated across different physical hosts without interrupting service. Today commercial providers such as Amazon and Rackspace maintain vast data centers that host millions of VMs. These cloud providers relieve their customers of the burden of managing data centers and achieve economies of scale, thereby lowering costs.

While operating-system virtualization is undeniably useful, it adds yet another layer to an already highly layered software stack now including support for old physical protocols (for example, disk standards developed

谢谢！

1、进程控制

- 进程控制的主要功能是为作业创建进程、撤销已结束的进程，以及控制进程在运行过程中的状态转换。

2、进程同步

- 进程同步的主要任务是为多个进程的运行进行协调。
- 两种协调方式：
 - 进程互斥方式，这是指诸进程（线程）在对临界资源进行访问时，应采用互斥方式；
 - 进程同步方式，指进程相互合作去完成共同的任务时，诸进程之间的协调。

3、进程通信

- 进程之间的信息交换：Inter Process Communication
- 机器内
- 机器间

4、调度

- 在操作系统中作业/进程运行需经调度才能执行完成。



1、内存分配

- 内存分配有两种方式
 - ①静态分配方式，每个作业运行之前分配好内存空间，在作业的整个运行期间不再改变。
 - ②动态分配方式中，每个作业在运行前或运行中，均可申请新的附加内存空间，以适应程序和数据的动态增涨。

结构和功能

- ①内存分配的数据结构，该结构用于记录内存空间的使用情况。
- ②内存分配功能——为用户程序分配内存空间；
- ③内存回收功能——当用户不再需要的内存时，系统能回收内存的功能。

2、内存保护

- 内存保护的主要任务：
 - 确保每道用户程序都只在自己的内存空间内运行，彼此互不干扰。
- 内存保护机制：
 - 设置两个界限寄存器，越界检查都由硬件实现

3、地址映射

- 地址空间——目标程序或装入程序限定的空间，称为“地址空间”。单元的编号称为逻辑地址，又称为相对地址。
- 内存空间——由内存中的一系列单元所限定的地址范围称为“内存空间”，其中的地址称为“物理地址”。
- 地址映射——运行时，将地址空间中的逻辑地址转换为内存空间中与之对应的物理地址，称为地址映射。

4、内存扩充

- 借助于虚拟存储技术
 - 从逻辑上去扩充内存容量，使用户所感觉到的内存容量比实际内存容量大得多；
- 扩充内存必须具有内存扩充机制：
 - 请求调入功能。在程序运行过程中，若所需的程序和数据尚未装入内存，可由OS从磁盘中将所需部分调入内存，继续运行。
 - 置换功能。将内存中的一部分暂时不用的程序和数据调出到磁盘上，然后再将所需调入的部分装入内存。

1、缓冲管理

- 有效地缓和CPU和I/O设备速度不匹配的矛盾，提高CPU的利用率。
- 对于不同的系统，可以采用不同的缓冲区机制

2、设备分配

- ①设备分配的基本任务，是根据用户进程的I/O请求，按照某种设备分配策略，为之分配其所需的设备。
- ②为了实现设备分配，系统中应设置设备控制表、控制器控制表等数据结构，用于记录设备及控制器的标识符和状态，以供进行设备分配时参考。
- ③不同的设备类型（独占、共享）而采用不同的设备分配方式。

3、设备处理

- 设备处理程序又称为设备驱动程序。
- 设备处理其基本任务：是用于实现CPU和设备控制器之间的通信，即由CPU向设备控制器发出I/O命令，要求它完成指定的I/O操作；反之由CPU接收从控制器发来的中断请求，并给予迅速的响应和相应的处理。
- 处理过程：检查请求的合法性→设备空闲否？→向控制器发I/O命令→启动I/O执行。

1、文件存储空间的管理

- 对诸多文件及文件的存储空间，实施统一的管理。基于数据结构（MCB）对存储空间进行分配和回收的功能。

2、目录管理

- 为每个文件建立目录项，并对众多的目录项加以有效的组织与管理（例如，按名存取，文件共享）。

3、文件的读／写管理和保护

- 文件的读／写管理
 - 根据用户的请求，从外存中读取数据或将数据写入外存。
- 文件保护：即存取控制功能：
 - 防止未经核准的用户存取文件；
 - 防止冒名顶替存取文件；
 - 防止以不正确的方式使用文件。

用户接口：命令接口

- 用户可通过该接口向作业发出命令以控制作业的运行。
 - 联机用户接口：这是为联机用户提供的，它由一组键盘操作命令及命令解释程序所组成。
 - 脱机用户接口：用户用JCL把需要对作业进行的控制和干预，事先写在作业说明书上，然后将作业连同作业说明书一起提供给系统。当系统调度到该作业运行时，再调用命令解释程序，对作业说明书上的命令，逐条地解释执行。该接口即为批处理接口。

2、程序接口

- 该接口是为用户程序在执行中访问系统资源而设置的，是用户程序取得操作系统服务的惟一途径。
- 它是由一组系统调用组成，每一个系统调用都是一个能完成特定功能的子程序，每当应用程序要求OS提供某种服务（功能）时，便调用具有相应功能的系统调用。
- 不同的系统其调用形式不同。

3. 图形接口

- GUI: Graphic User Interface
- 图形用户接口采用了图形化的操作界面，用非常容易识别的各种图标（icon）来将系统的各项功能、各种应用程序和文件，直观、逼真地表示出来。用户可用鼠标或通过菜单和对话框，来完成对应用程序和文件的操作。