

CSE291

Virtualization and Cloud Computing

- Course Overview

Yiying Zhang

Before we start

- This course is about virtualization and cloud computing. If you feel that this is not for you, e.g., if you don't like paper reading, if you don't like doing one big open-ended group project per quarter, if you have no software development experience, if you have no knowledge of **undergrad OS+architecture**, please feel free to leave now.
- Class website: <https://cseweb.ucsd.edu/~yiying/cse291-spring24/>
- Class time: MW 9:30pm - 10:50pm in Room CSE 2154

Who Am I

- Associate prof at UCSD CSE
- Working on datacenter systems
 - OS, systems and ML, computer architecture, networking, security, programming language, etc.
- Leads WukLab (wuklab.io)

Course Structure

- Most are lectures given by me or paper discussion led by me
- Rest are paper discussion led by student volunteers (with bonus points)
- Most classes require paper reading, rest are online material learning
 - Need to submit paper summaries and answer questions before class
- One group project (open-ended, research oriented)
- 3 quizzes throughout the quarter
- No exam

Tentative Grading Scheme

- 25% Paper reading and class participation/attendance (attendance tracked from next week)
- 15% Quizzes
- 60% Project
 - Project proposal 15% April 18
 - Project progress report 5% May 9
 - Final project presentation 7% (about 10 min per group) June 5
 - Final project report 33% (3-6 pages) June 13
- Up to 5pts Bonus (discussion lead)

Sample Turn-in of Paper Summary

Only PDF is accepted

Name your PDF as [month]-[day]-[your first name]-[your last name].pdf

1-10-yiying-zhang.pdf

A Comparison of Software and Hardware Techniques for x86 Virtualization

Summary and your overall feeling of the paper: 2-5 sentences

- Q1: Why is x86 un-virtualizable with trap-and-emulate? Give one example.
- A: 1-3 sentences

- Q2: How are jump instructions translated?
- A: 1-3 sentences

- Q3: With hardware virtualization extensions (e.g., Intel VT), do we still need binary translation? Why or why not?
- A: 1-3 sentences

Paper Discussion Lead

To gain bonus points, you can volunteer to lead or co-lead paper discussion. Here are what you need to do for discussion lead:

- Prepare slides that you will use to lead discussion. You can use slides that are available online, build slides yourself, or extend existing online slides with your own content.
- You should not only just explain the paper (using existing online slides) but also prepare questions that will stimulate class discussion.
- Send draft slides to me at least 48 hours before the corresponding class start time. I will make comments and suggestions, which you should incorporate in the final version. The sooner you send your draft slides to me, the more helpful feedback I can give you.

Course Project

- One research-oriented, open-ended project in groups of 1-4
- Be prepared to do substantial programming and self-learning, start early!
- Most projects have assigned topics
 - But you need to formulate your own problem and solution
 - Allow self-defined project topics, discuss with me ASAP if you choose one
- Form your group by next Wed (4/10)!

Project Topics

- Build: Container migration
- Study: Cloud service comparison
- ML for systems: SQL/Spark query classification
- Systems for ML: Debugger for compound LLM
- Build/Fix: Solving Github issues for related open-source projects
- Propose your own (inform me by 4/10 if you want to choose this option)

More Exciting Research at WukLab!

Tentative Schedule

- 1 week: intro and virtualization overview
- 1.5 - 2 weeks: virtualizing CPU, memory, and I/O
- 0.5 week: cloud computing overview
- 1 week: container and container orchestration
- 1 week: serverless and resource disaggregation
- 0.5 - 1 week: systems and machine learning
- 0.5 weeks: hardware support
- 0.5 week: security
- 0.5 week: the future of cloud computing
- 0.5 week: course summary and project presentation

Diversity and Inclusion

- Respect for Diversity: It is my intent that students from all diverse backgrounds and perspectives be well served by this course, that students' learning needs be addressed both in and out of class, and that the diversity that students bring to this class be viewed as a resource, strength and benefit.
- All people have the right to be addressed and referred to in accordance with their personal identity. In this class, we will have the chance to indicate the name that we prefer to be called and, if we choose, to identify pronouns with which we would like to be addressed...I will do my best to address and refer to all students accordingly and support classmates in doing so as well.

Academic Integrity

As a student at UCSD you are subject to the UCSD POLICY ON INTEGRITY OF SCHOLARSHIP, which enjoins you to respect the highest standards of honesty and integrity. All work that you submit in this course must be your own; unauthorized group efforts are considered academic dishonesty. Academic dishonesty is a serious offense which may result in suspension or expulsion from the university. Students are encouraged to report academic dishonesty to the instructor directly, or to the Academic Integrity Office.

Another Chance To Leave Now

- If not, then your first (and very important) assignment is
 - First Day Survey: Prior Knowledge #FinAid
 - Available as a quiz (ungraded survey) on Canvas
 - Should only take a couple of minute
 - Due **April 12th**

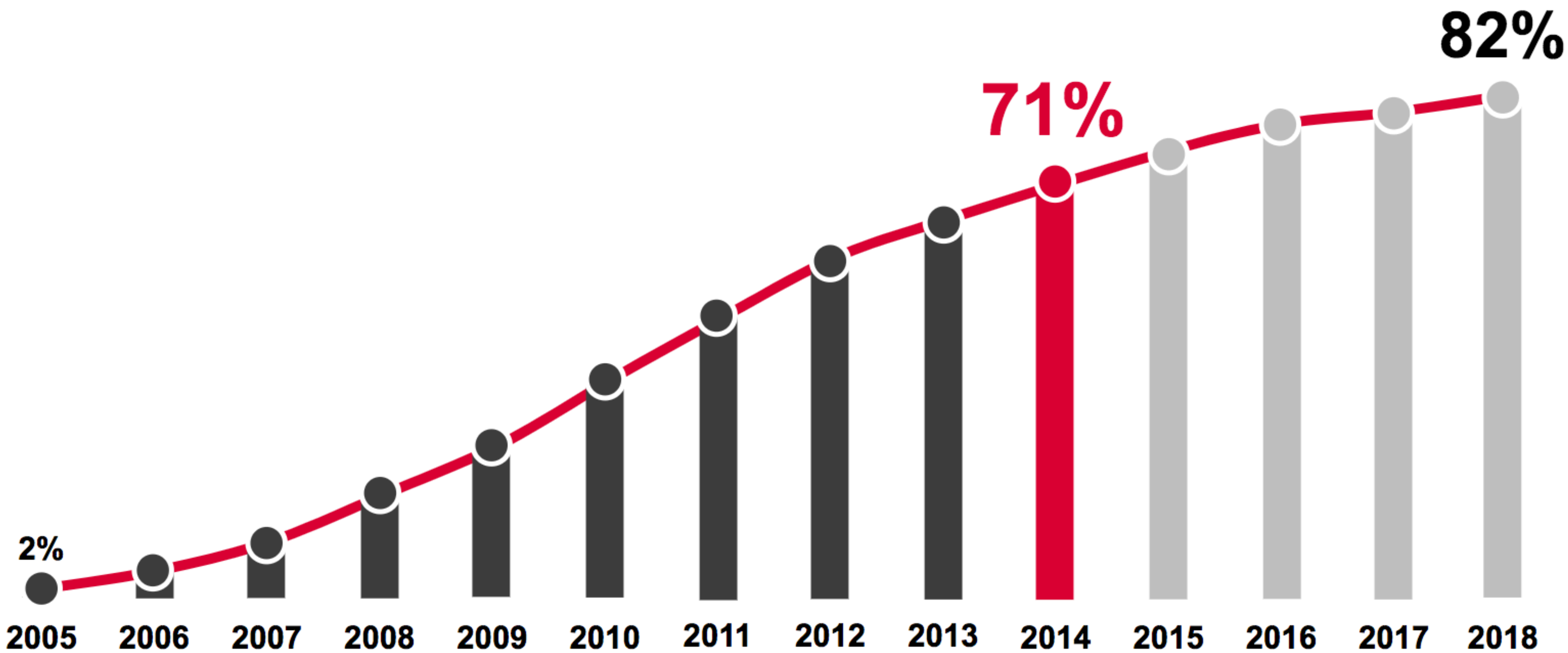
Why Studying Virtualization and Cloud Computing?

- Cloud computing dominates IT spending: over \$500 billion in 2023, over \$1 trillion by 2028
- Almost all cloud applications run in the virtualization environment
- Most IT infrastructures run in the cloud or on-prem virtualization env
- Understanding virtualization and cloud computing *internals* is key to building cloud and data-center infrastructures
- Understanding virtualization and cloud computing will help application design

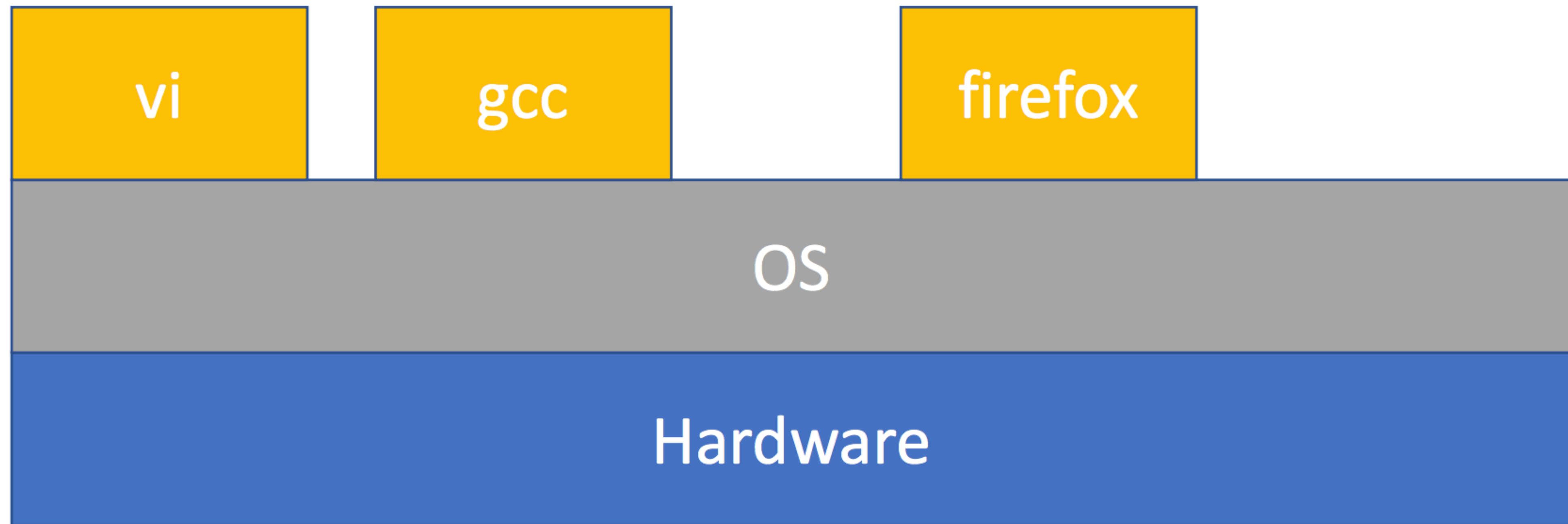
CLOUD COMPUTING MARKET SIZE, 2021 TO 2030 (USD BILLION)



Adoption of Virtualization



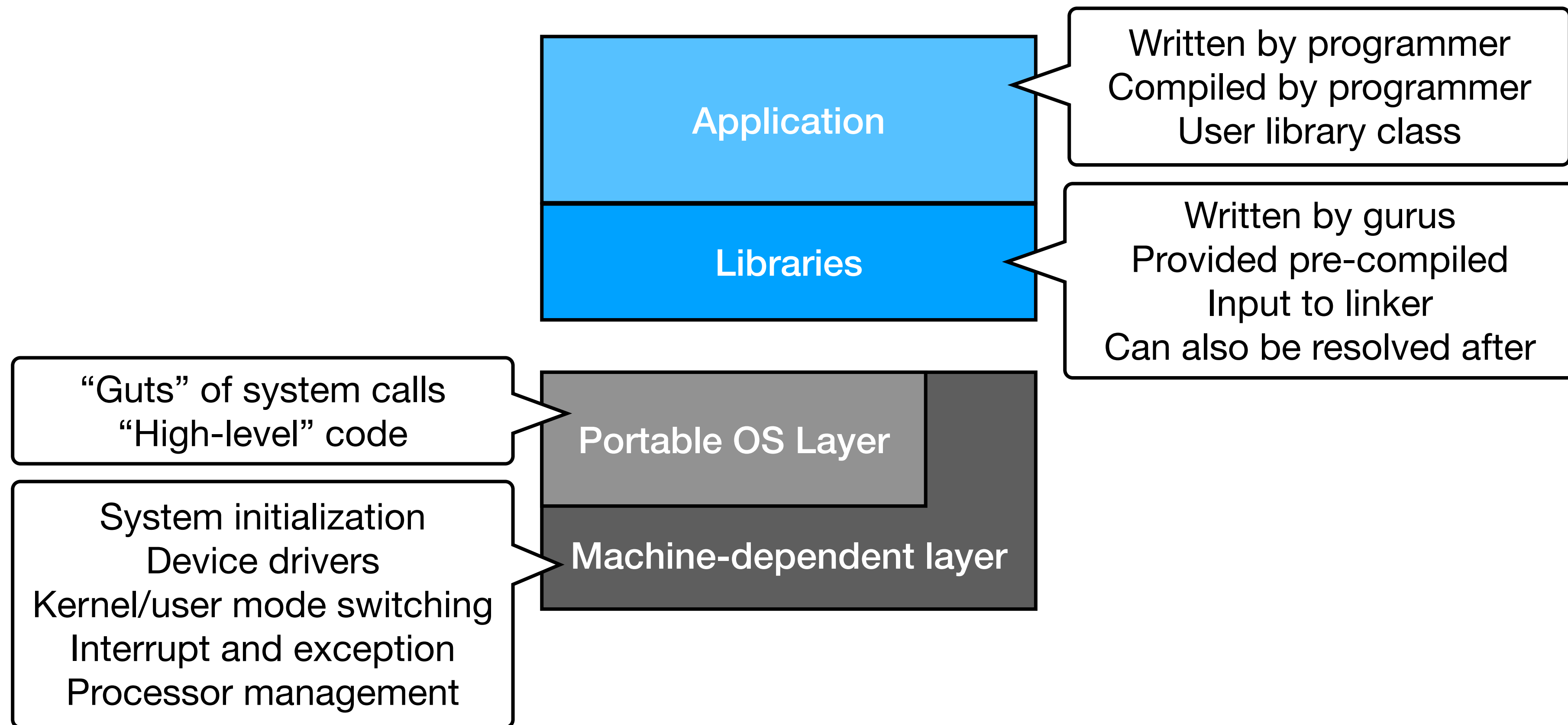
Regular Machine Stack



What is an OS?

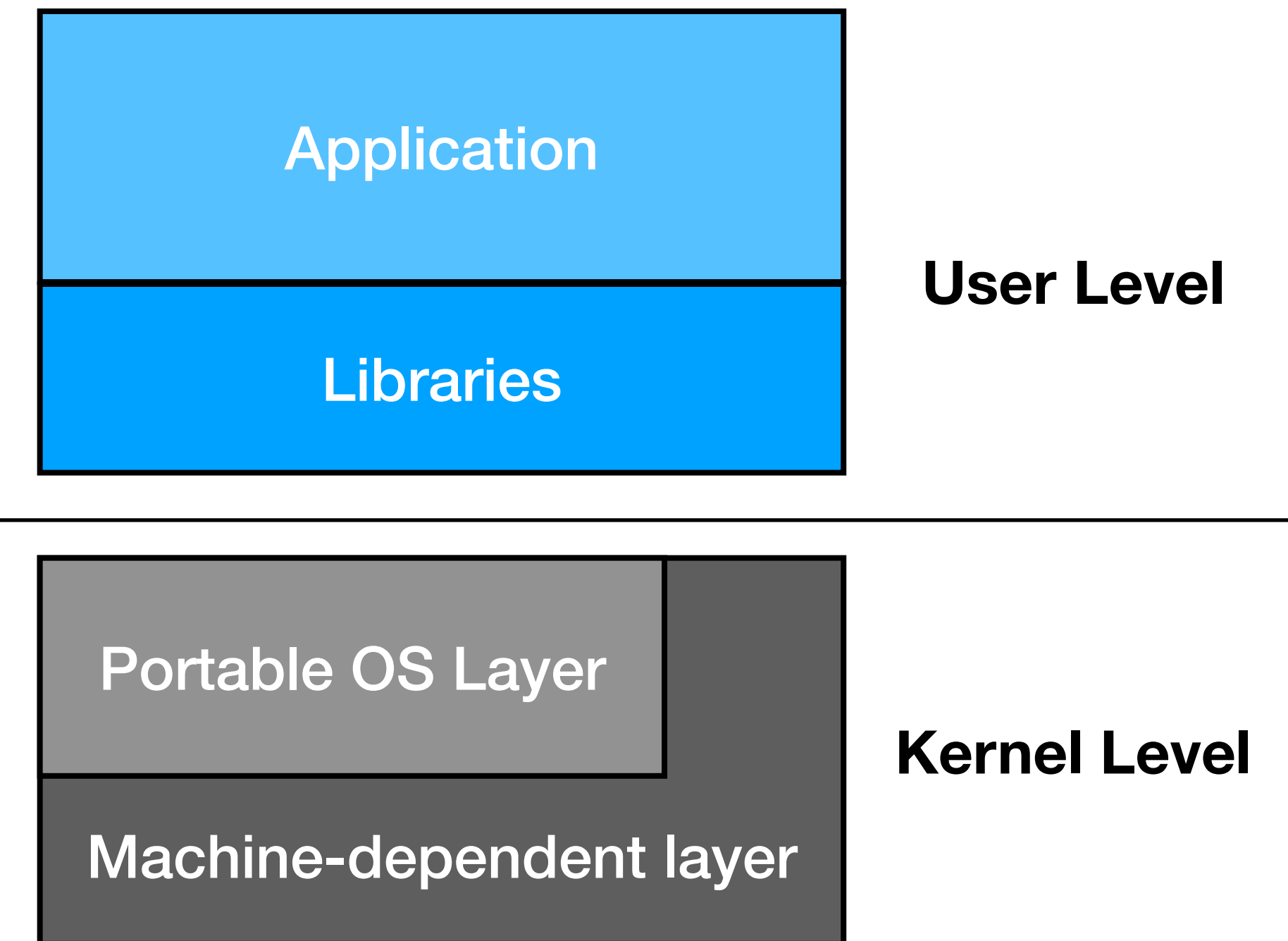
- A piece of software that manages and virtualizes hardware for applications
 - An indirection layer between applications and hardware
 - Provides a high-level interface to applications,
 - while interacting with hardware devices with low-level interfaces
 - Runs privileged instructions to interact with hardware devices
 - In a sense, a giant “interrupt handler”
- Applications
 - Can only execute unprivileged instructions
 - Perform system calls or faults to “*trap*” into OS
 - OS protect applications from each other (to some extent) (e.g., address space)

Typical OS Structure

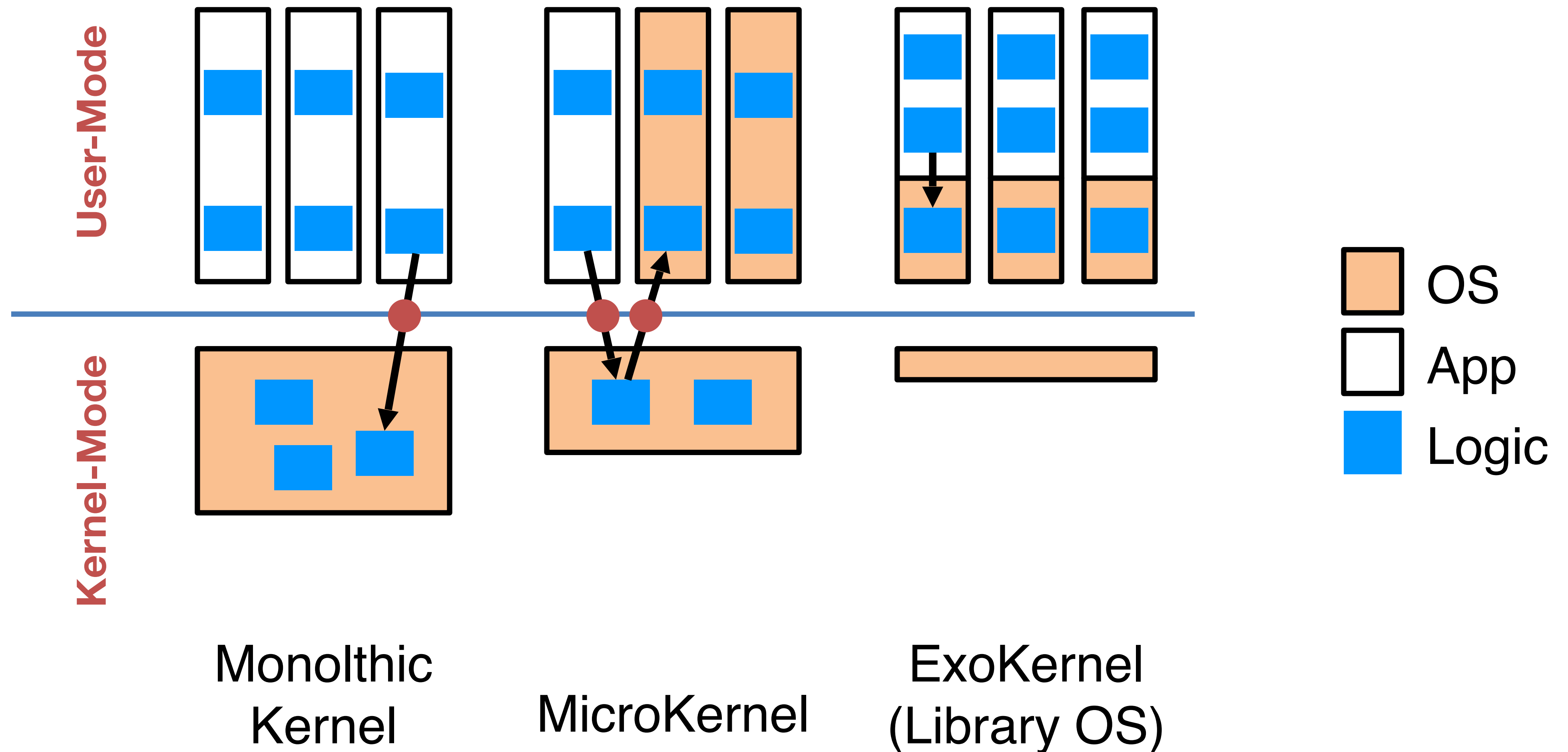


Dual-Mode Operation

- OS manages shared resources
 - OS protects programs from other programs
 - OS needs to be “privileged”
-
- Dual-mode operation of hardware
 - Kernel mode – can run *privileged* instructions
 - User mode – can only run *non-privileged* instructions



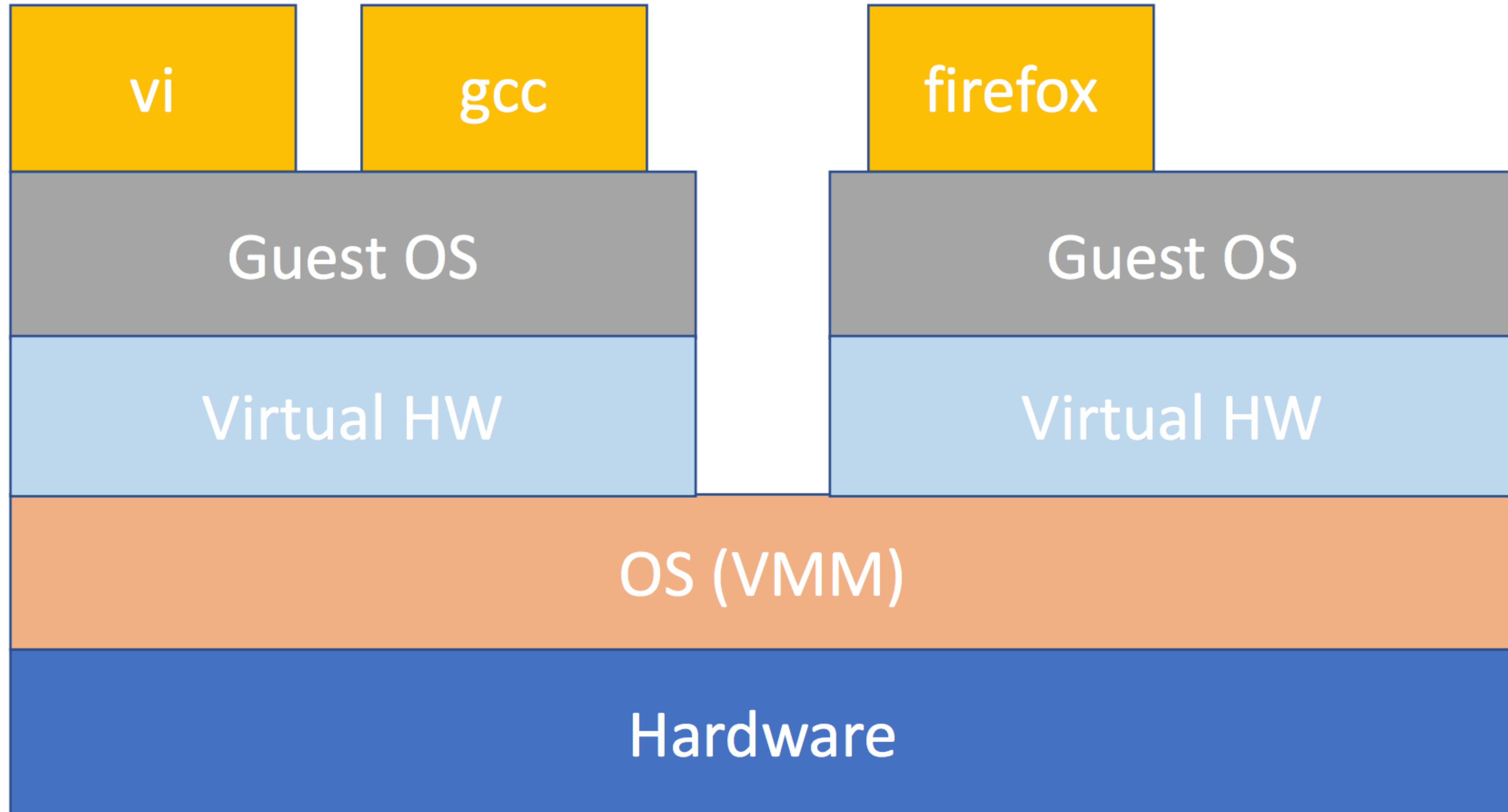
Different OS Structures



Virtualization

- Adding another level of indirection (e.g., run another OS on an abstraction of hardware)
- Virtual Machine (Guest OS)
 - OS that runs on virtualized hardware resources
 - Managed by another software (VMM/Hypervisor)
- Virtual Machine Monitor (Hypervisor)
 - The software that creates and manages the execution of virtual machines
 - Runs on bare-metal hardware

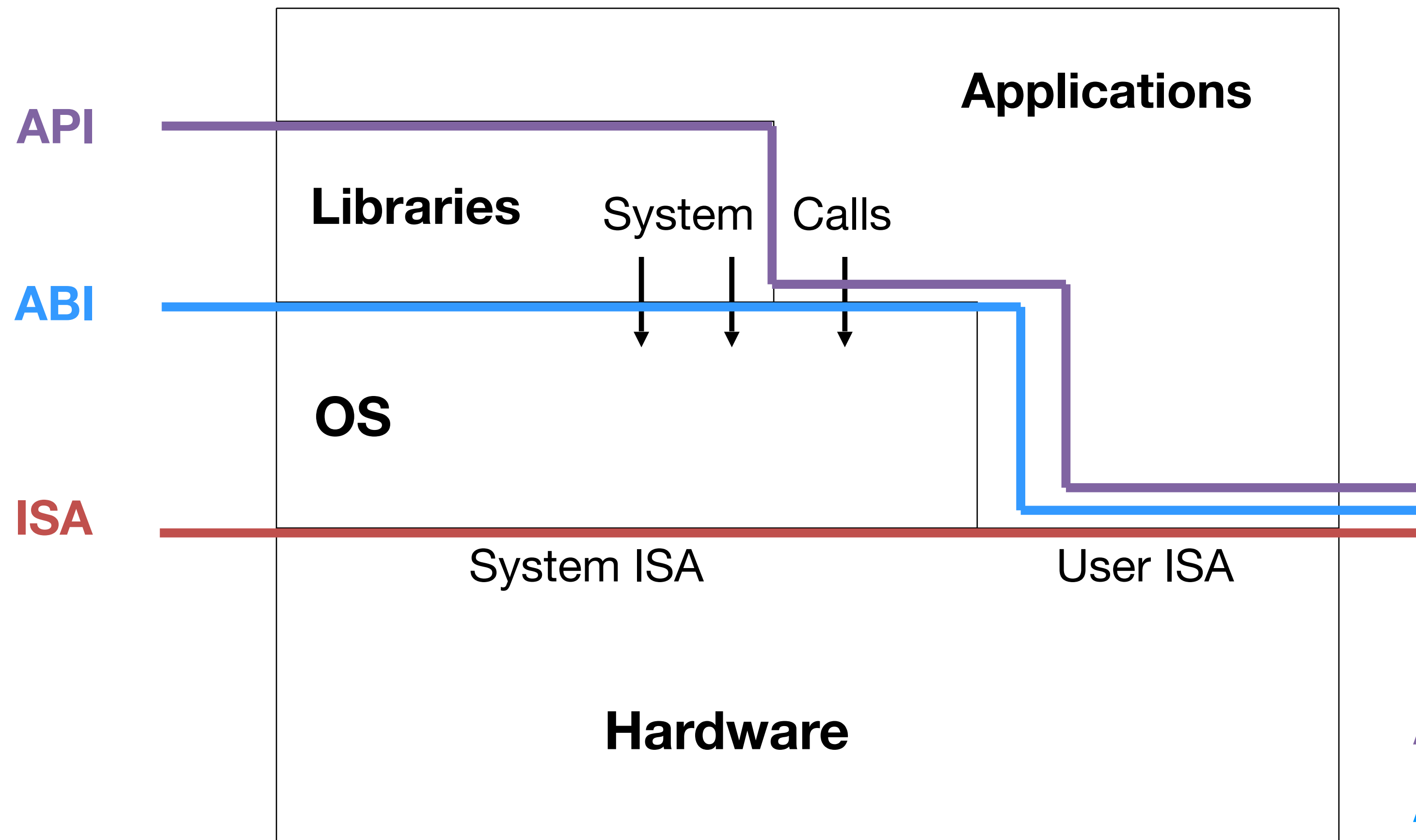
Virtualization



What Is An ISA?

- ISA (instruction set architecture)
 - A well-defined hardware/software interface
- The **“contract”** between software and hardware
 - **Functional definition** of operations, modes, and storage locations supported by hardware
 - **Precise description** of how to invoke, and access them
- No guarantees regarding
 - How operations are implemented
 - Which operations are fast and which are slow and when
 - Which operations take more power and which take less

Interaction between Different Layers



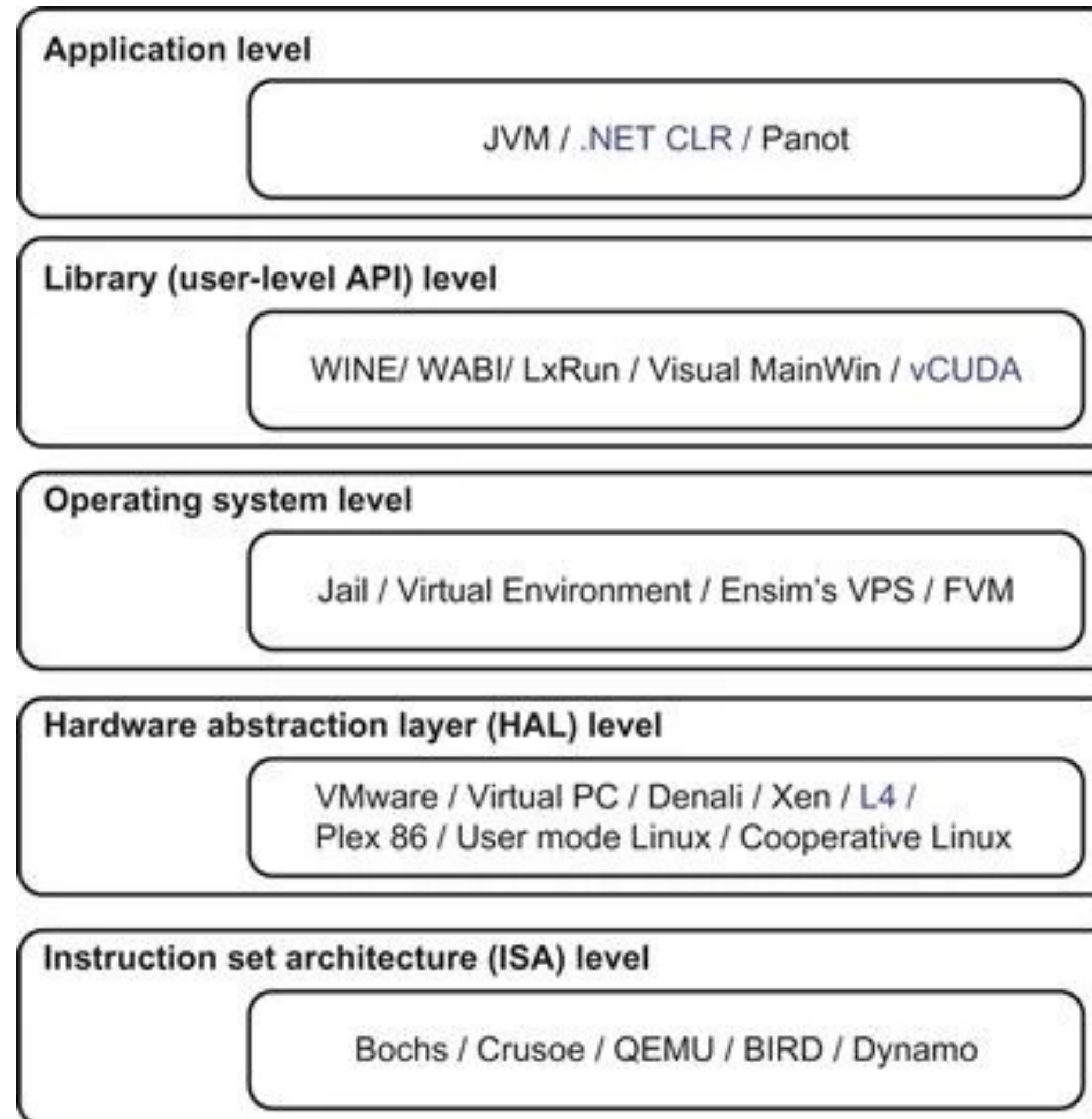
Which layer should virtualization be at?

API – application programming interface

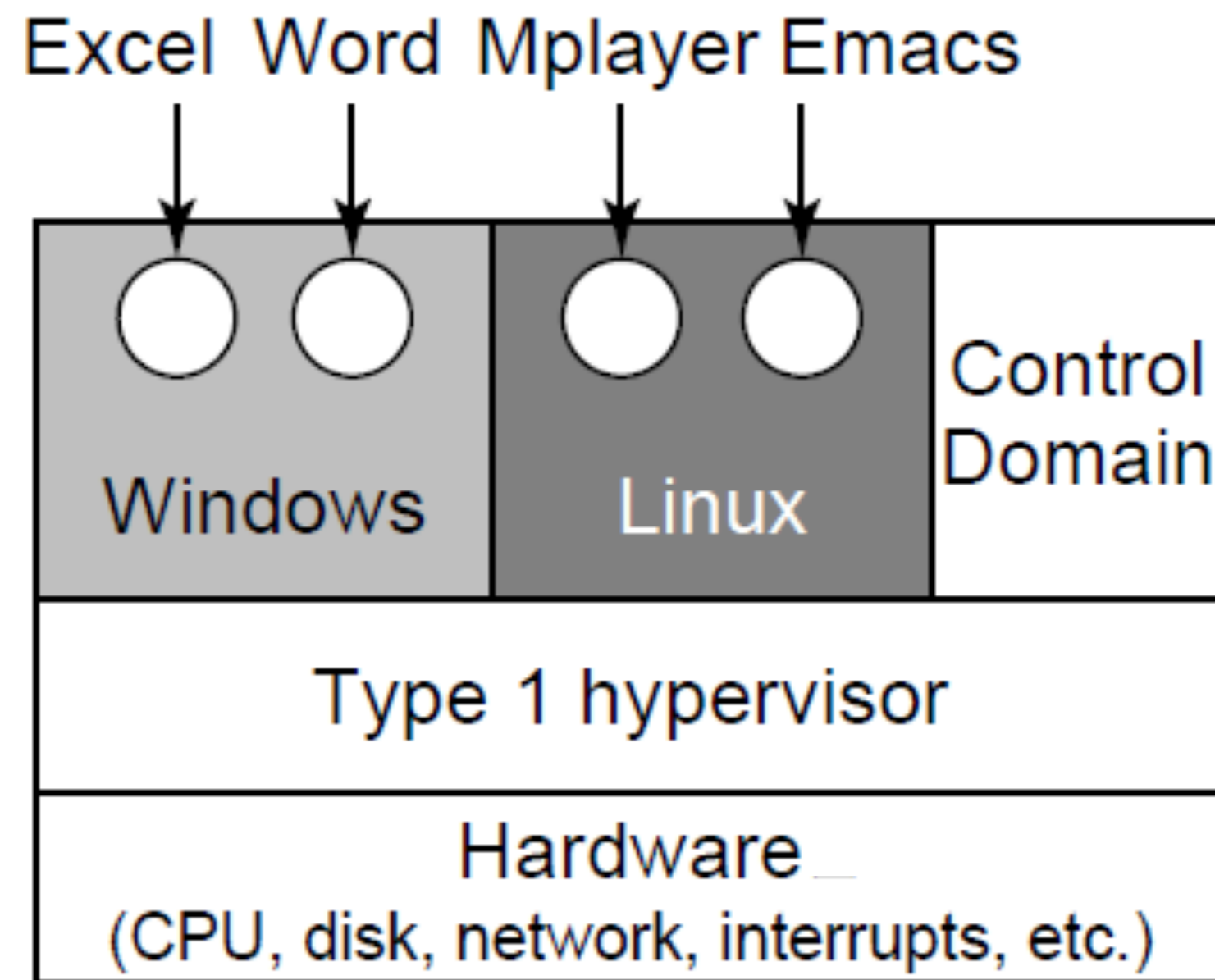
ABI – application binary interface

ISA – instruction set architecture

Five Levels of Virtualization Options

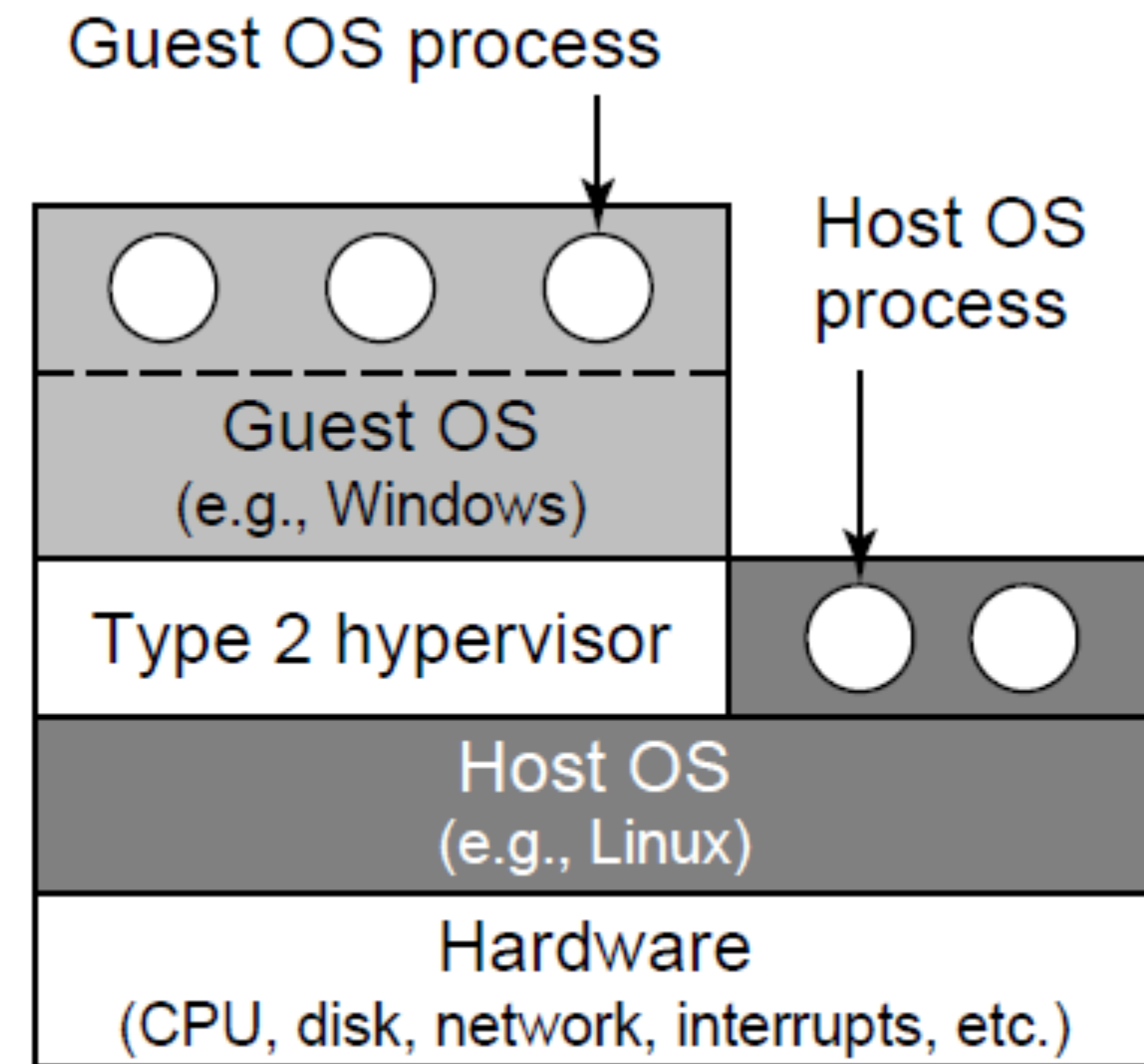


Type 1 and Type 2 Hypervisor (VMM)



(a)

VMware ESX, Microsoft Hyper-V, Xen



(b)

VMware Workstation, Microsoft Virtual PC, Sun VirtualBox, QEMU, KVM

Figure 7-1. Location of type 1 and type 2 hypervisors.

Virtualization: A Brief History

Mainframes and IBM

- Before we have datacenters or PCs, there were giant metal frames
- Support computational and I/O intensive commercial/scientific workloads
- Expensive (IBM 704 (1954) costs \$250K to millions)
- “IBM and the seven dwarfs” – their heyday was the late ‘50s through ‘70s

Issues with Early Mainframes

- Different generations were not architecturally compatible

=> Headache to port software

- Batch-oriented
- Meanwhile, ideas started to appear towards a time-sharing OS
- The computer was becoming a multiplexed tool for a community of users, instead of being a batch tool for wizard programmers

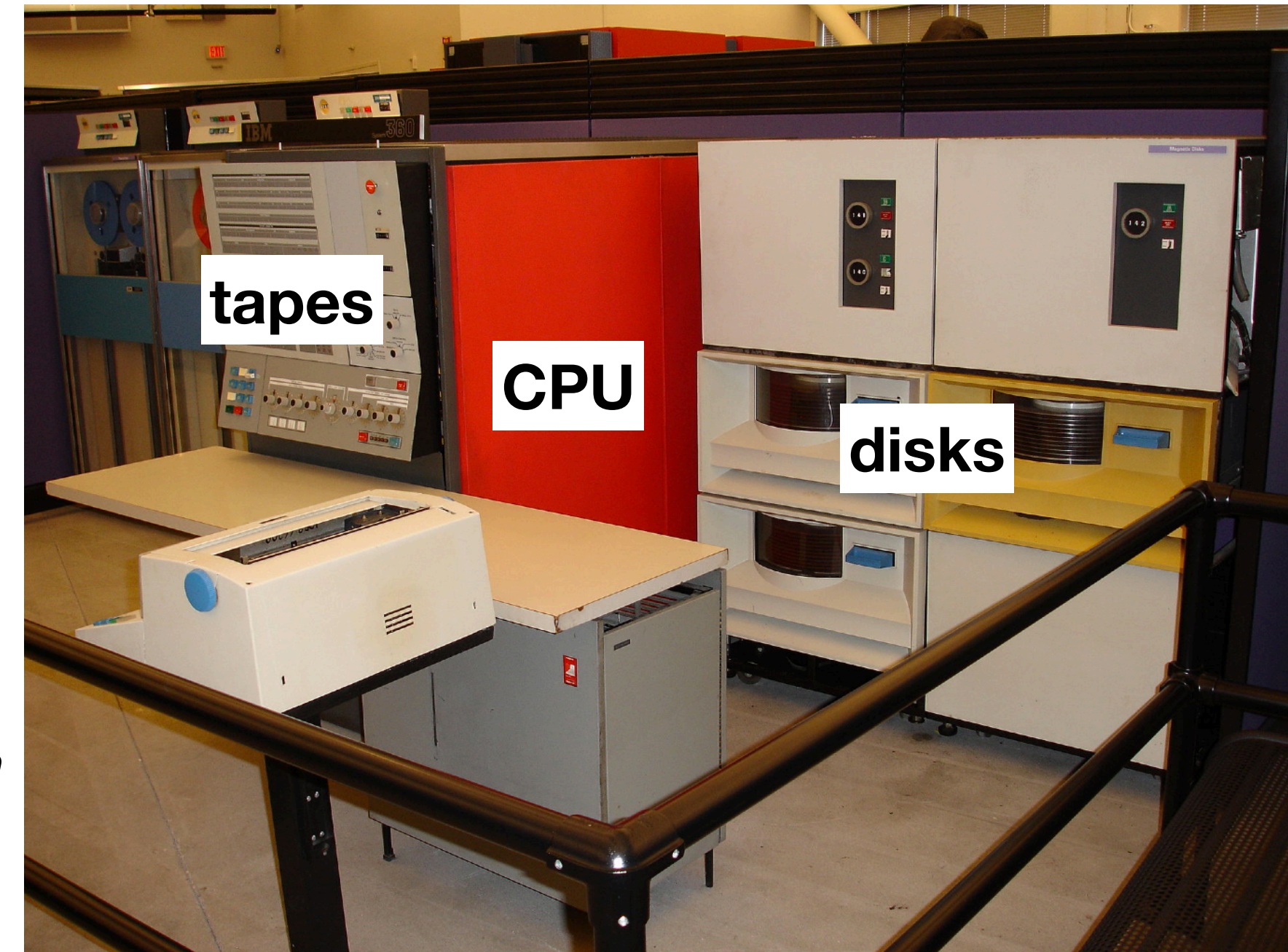
IBM's Response

- IBM bet the company on the System/360 hardware family [1964]
 - S/360 was the first to clearly distinguish architecture and implementation
 - Its architecture was virtualizable
- The CP/CMS system software [1968]
 - CP: a “control program” that created and managed *virtual S/360 machines*
 - CMS: the “Cambridge monitor system” -- a lightweight, single-user OS (that can run in a CP's VM)
 - With CP/CMS, can run several OSs concurrently on the same HW
- IBM CP/CMS is the first virtualization system. Main purpose: *multiple users can share a mainframe*

IBM's Mainframe Product Line

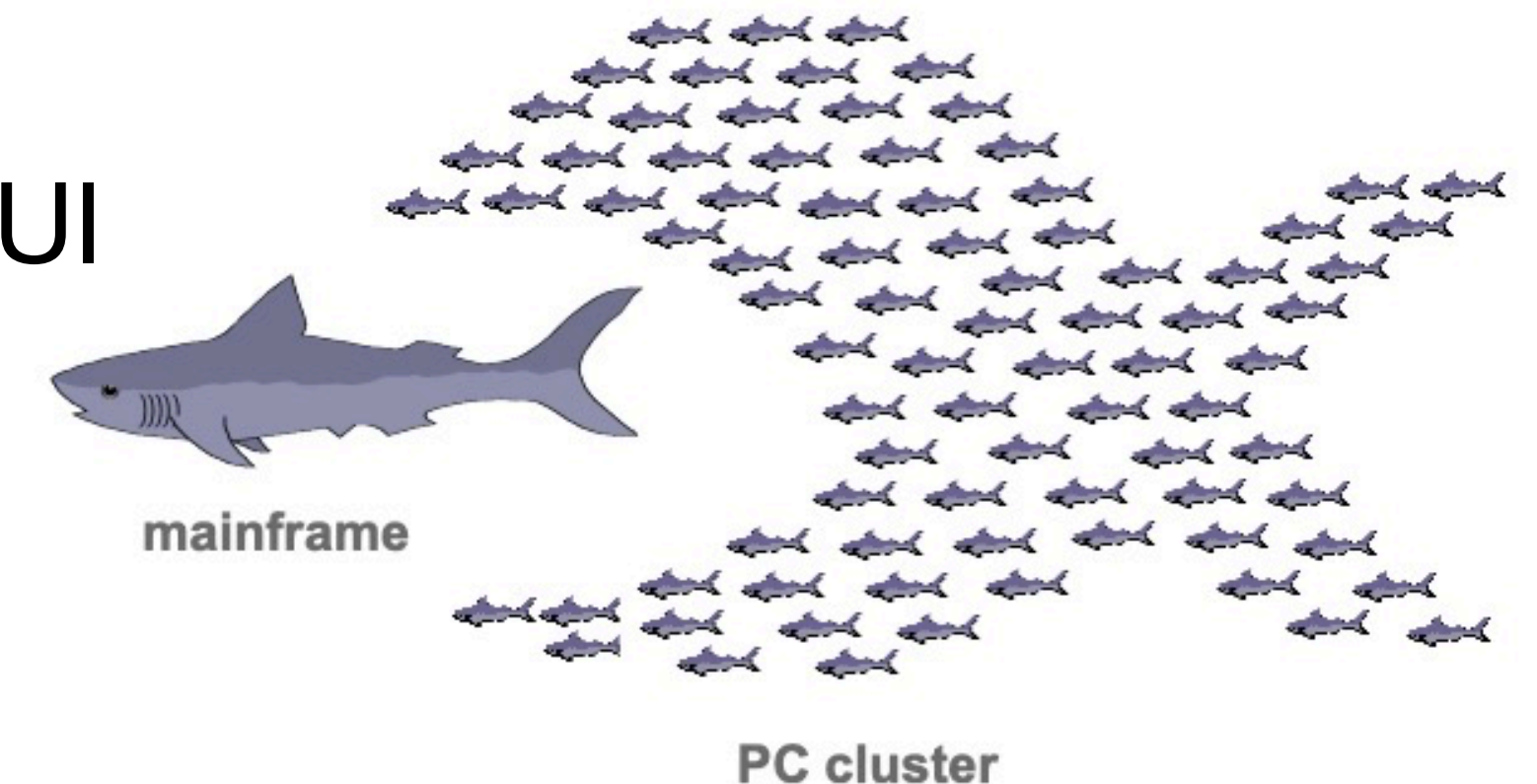
- System/360 (1964-1970)
 - Support virtualization via CP/CMS, channel I/O, virtual memory...
- System/370 (1970-1988)
 - Reimplementation of CP/CMS as VM/370
- System/390 (1990-2000)
- zSeries (2000-present)
- Huge moneymaker for IBM, and many business still depend on these!

***IBM System/360
(in Computer History Museum
in Mountain View)***



PCs and Multi-User OSes

- 1976: Steve Jobs and Steve Wozniak started Apple Computers and rolled out the Apple I, the first computer with a single-circuit board
- 1981: The first IBM personal computer, code-named "Acorn," is introduced. It uses Microsoft's MS-DOS
- 1983: Apple's Lisa is the first personal computer with a GUI
- 1985: Microsoft announces Windows
- The PC market (1980-90s): ship hundreds of millions of units, not hundreds of units
- Cluster computing (1990s): build a cheap mainframe out of a cluster of PCs



Multiprocessor and Stanford FLASH

- Development of multiprocessor hardware boomed (1990s)
- Stanford FLASH Multiprocessor
 - A multiprocessor that integrates global cache coherence & msg passing
- But system software lagged behind
- Commodity OSes do not scale and cannot isolate/contain faults

A Real FLASH Multiprocessor



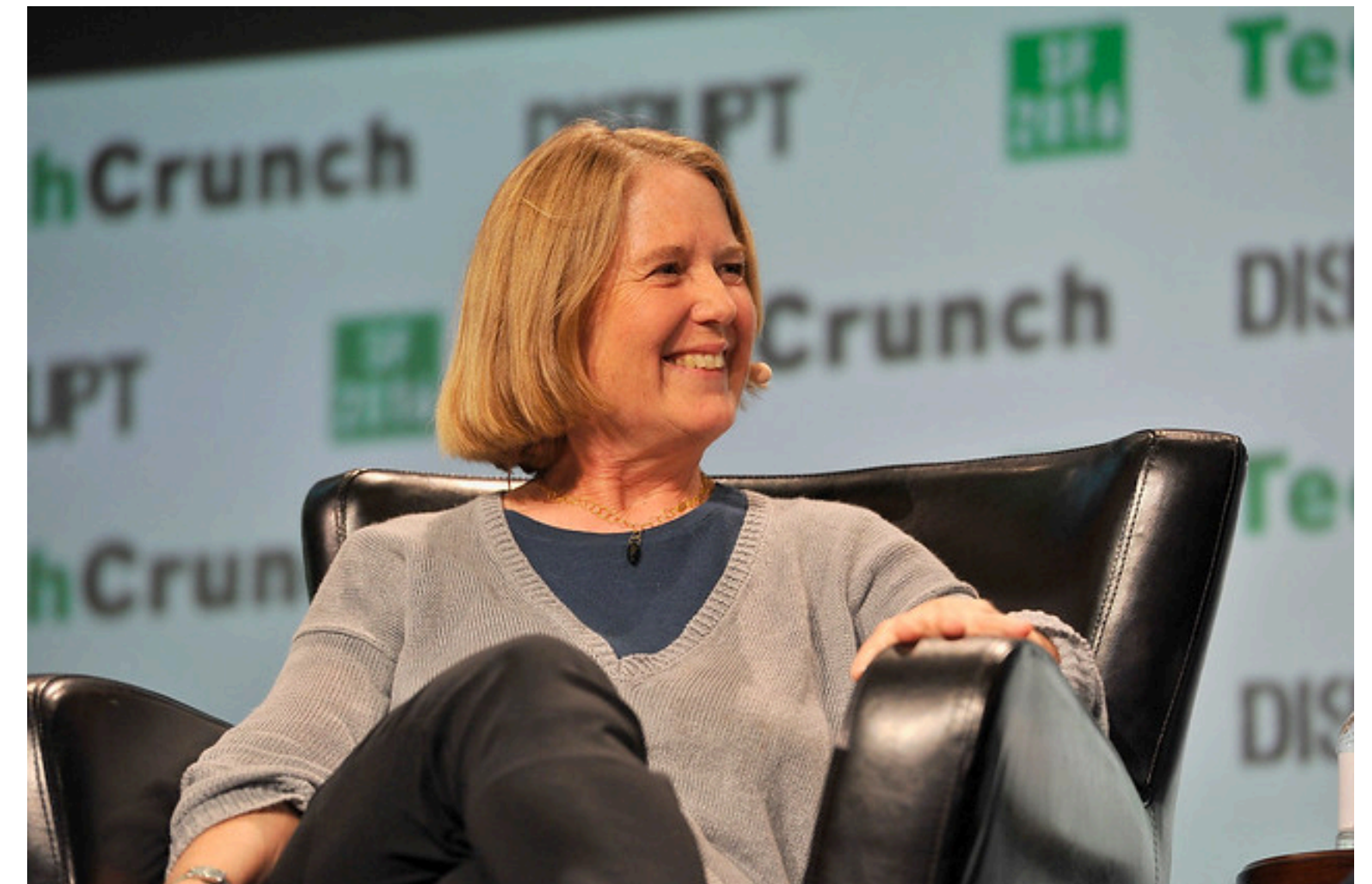
Photo taken by Yiying Zhang at Mendel Rosenblum's Office

Stanford Disco and VMware

- Stanford Disco project (SOSP'97 Mendel Rosenblum et al.)
 - Extend modern OS to run efficiently on shared memory multiprocessors
 - A VMM built to run multiple copies of Silicon Graphics IRIX OS on FLASH
- Mendel Rosenblum, Diane Greene, and others co-founded VMware in 1998
 - Brought virtualization to PCs. Main purpose: *run different OSes on different arch*
 - Initial market was software developers for testing software in multiple OSes
 - Acquired by EMC (2003), which later merged with DELL (2016)

Women in Computing: Diane Greene

- Co-founder and the CEO of VMware from 1998 to 2008
- The CEO of Google's cloud businesses from 2015 until early 2019
- An Alphabet board of directors member from 2012 to 2019
- 2017 Abie Award for Technical Leadership from AnitaBorg
- 2018 Forbes' America's Top 50 Women In Tech



Server Consolidation

- Datacenters often run many services (e.g., search, mail server, database)
 - Easier to manage by running one service per machine
 - Leads to low resource utilization
- Virtualization can “consolidate” servers by hosting many VMs per machine, each running one service
 - Higher resource utilization while still delivering manageability

The Cloud Era

- The cloud revolution is what really took virtualization on
- Instead of renting physical machines, rent VMs
 - Better consolidation and resource utilization
 - Better portability and manageability
 - Easy to deploy and maintain software
 - However, raise certain security and QoS concerns
- Many instance types, some with specialized hardware; all well maintained and patched
 - AWS: 241 instance types in 30 families (as of Dec 2019)

Container

- VMs run a complete OS on emulated hardware
 - Too heavy-weighted and unnecessary for many cloud usages
 - Need to maintain OS versions, libraries, and make sure applications are compatible
- Containers (e.g., Docker, LXC)
 - Run multiple isolated user-space applications on the host OS
 - Much more lightweight: better runtime performance, less memory, faster startup
 - Easier to deploy and maintain applications
 - But doesn't provide as strong security boundaries as VMs

Serverless Computing

- VMs and containers in cloud still need to be “managed” by users
- Is there a way to just write software and let the cloud do all the rest?
- Serverless computing (mainly in the form of Function as a Service)
 - Event triggered, autoscaled, billed by request load
 - No need to manage “server cluster” or handle failure
 - But a lot less control and customization (e.g., fixed CPU/memory ratio, no direct communication across functions, no easy way to maintain states)

Summary of Virtualization History

- Invented by IBM in 1960s for sharing expensive mainframes
- Popular research ideas in 1960s and 1970s
- Interest died as the adoption of cheap PCs and multi-user OSes surged in 1980s
- A (somewhat accidental) research idea got transferred to VMware
- Real adoption happened with the growth of cloud computing
- New forms of virtualization: container and serverless, in the modern cloud era

Next time...

- Overview on virtualization
- Reading: Comet Book Chapter on Virtual Machine Monitors
- No questions for next time, will start from next week
- Work on the FinAid quiz (it will only take a couple of min)!
- Start looking for project teammates!!