Virtual Machine Systems

Question

- Can a "small" operating system simulate the hardware of some machine so that
  - Another operating system can run in that simulated hardware?
  - More than one instance of that operating system run on the same hardware at the same time?
  - More than one different operating system can share the same hardware at the same time?
- Answer: **Yes**

Virtual Machine

- A virtual machine provides interface identical to underlying bare hardware
  - i.e., all devices, storages, memory, page tables, etc.

- Virtual Machine Operating System creates **illusion** of multiple processors
  - Each VM executes independently
  - No sharing, except via network protocols

History – CP67 / CMS

- IBM Cambridge Scientific Center
- Ran on IBM 360/67
    - Alternative to TSS/360, which never sold very well
- <u>Replicated hardware</u> in each "process"
    - Virtual 360/67 processor
    - Virtual disk(s), virtual console, printer, card reader, etc.
- Cambridge Monitor System (CMS)
    - A single user, interactive operating system

- Commercialized as VM370 in mid-1970s

History (cont.)

- Various other attempts with other machines

- VMware
  - Workstation
  - Servers (for IT centers)

"Classic" Virtual Machine

- ## Copy of a real machine
  - "Any program run under the VM has an effect identical with that demonstrated if the program had been run in the original machine directly" [1]
- ## Isolated from other virtual machines
  - "…transforms the single machine interface into the illusion of many" [2]
- ## Efficient
  - "A statistically dominant subset of the virtual processor's instructions is executed directly by the real processor" [2]
- ## Also known as a "system VM"

[1] *"Formal Requirements for Virtualizable Third-Generation Architectures", G. Popek and R. Goldberg, Communications of the ACM, 17(7), July 1974*
[2] *"Survey of Virtual Machine Research", R. Goldberg, IEEE Computer, June 1974*
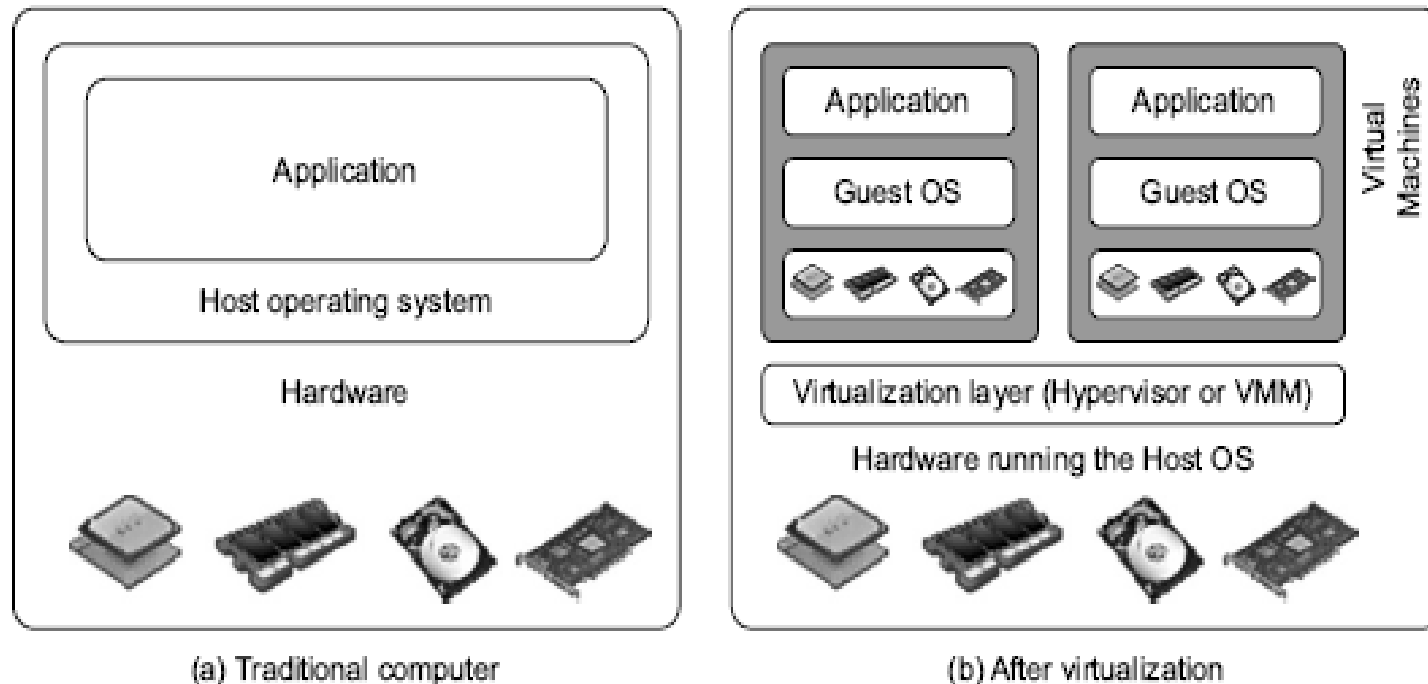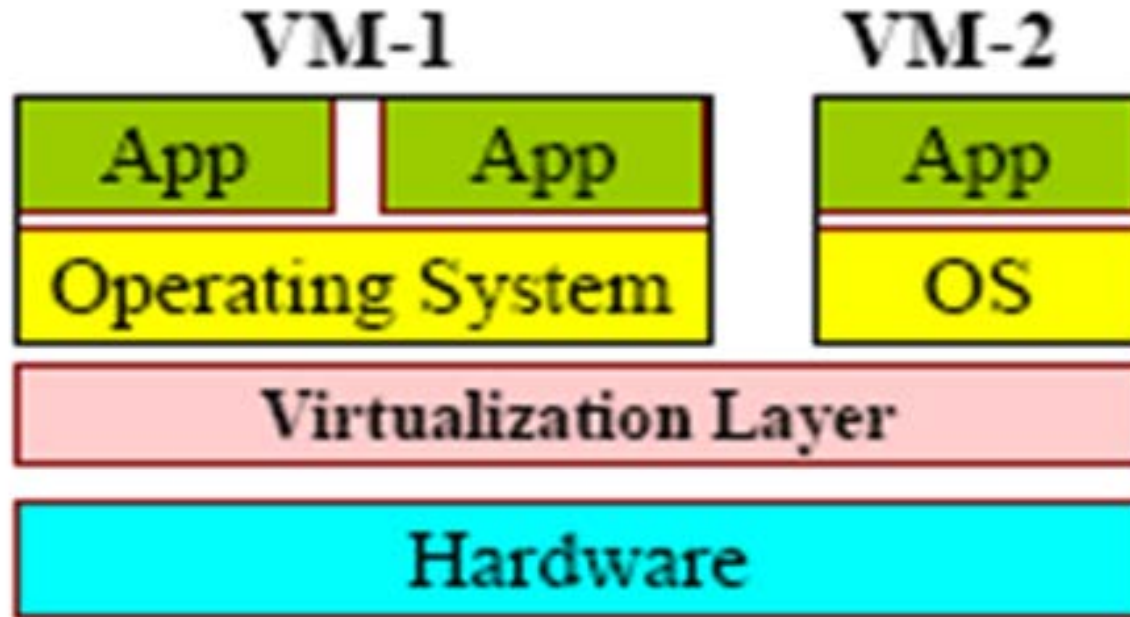
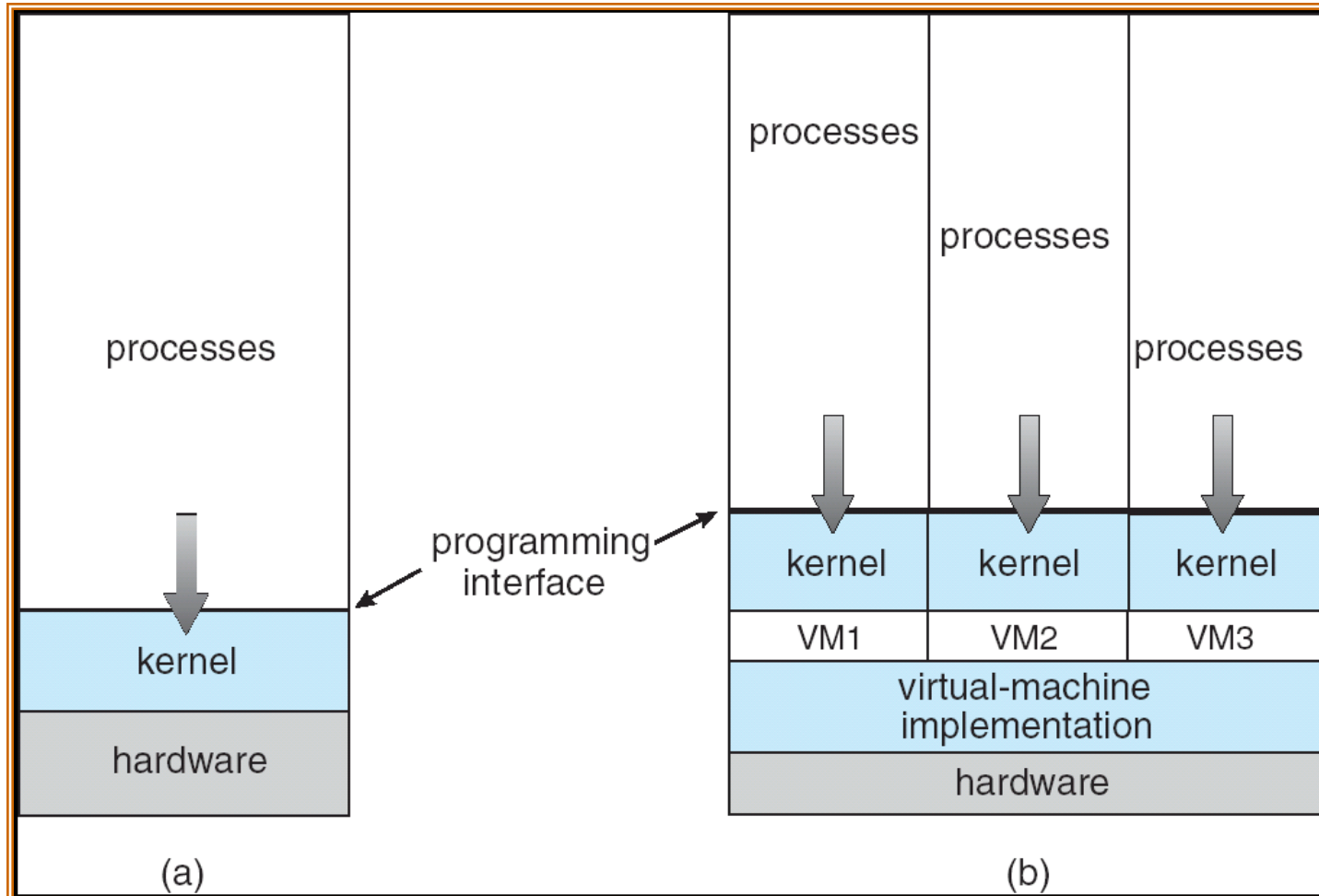# Traditional Computer and Virtual Machines

FIGURE 3.1

The architecture of a computer system before and after virtualization, where VMM stands for virtual machine monitor.

What is Virtualization?

# Virtual Machines

(a) Nonvirtual machine   (b) virtual machine

Classic Virtual Machines

- Virtualization of instruction sets (ISAs)

  - Language-independent, binary-compatible (*not* JVM)

- 70's (IBM 360/370..) – 00's (VMware, Microsoft Virtual Server/PC, z/VM, Xen, Power Hypervisor, Intel Vanderpool, AMD Pacifica …)

- ISA+ OS + libraries + software = **execution environment**

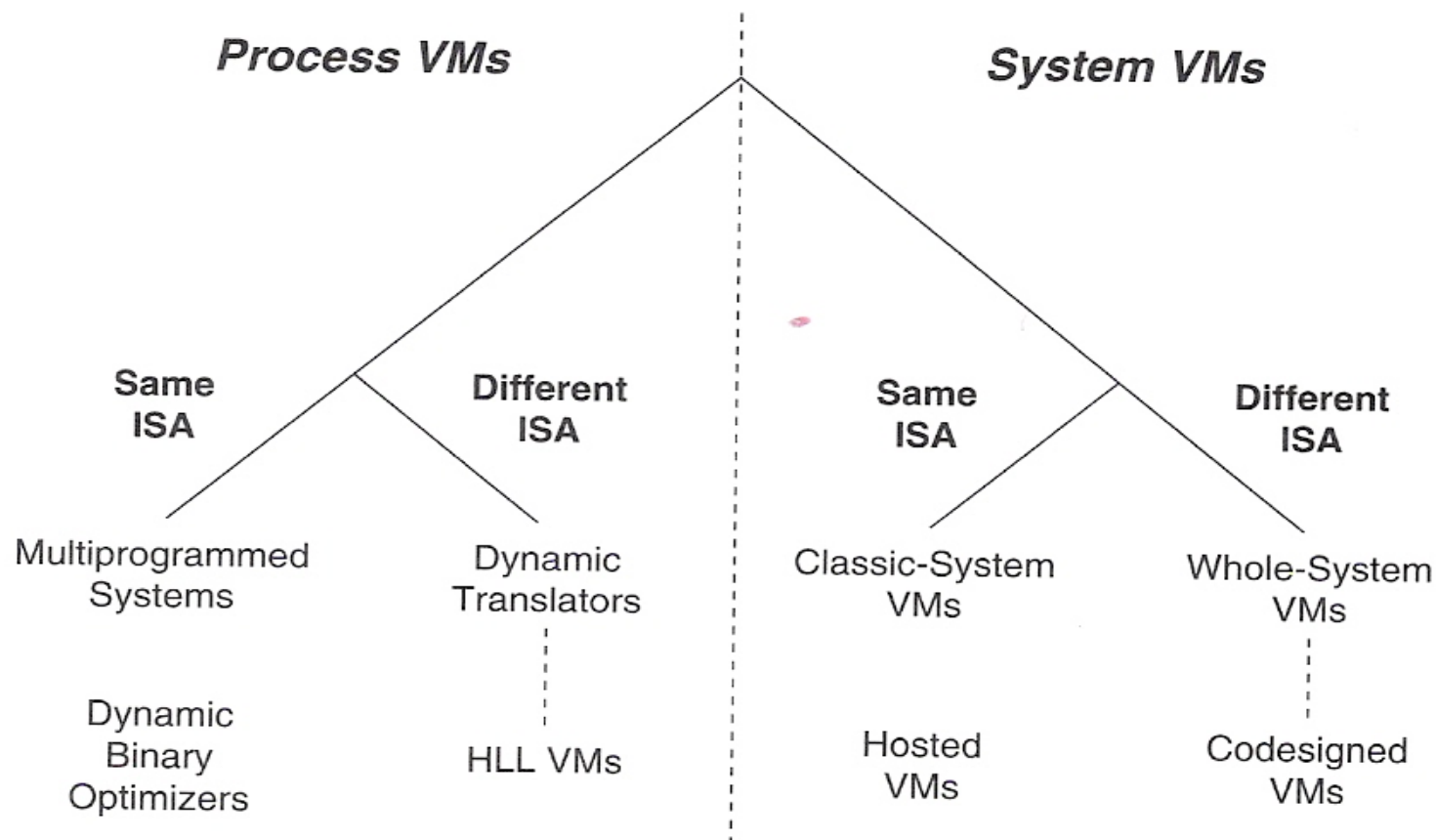# User's View of Virtualization
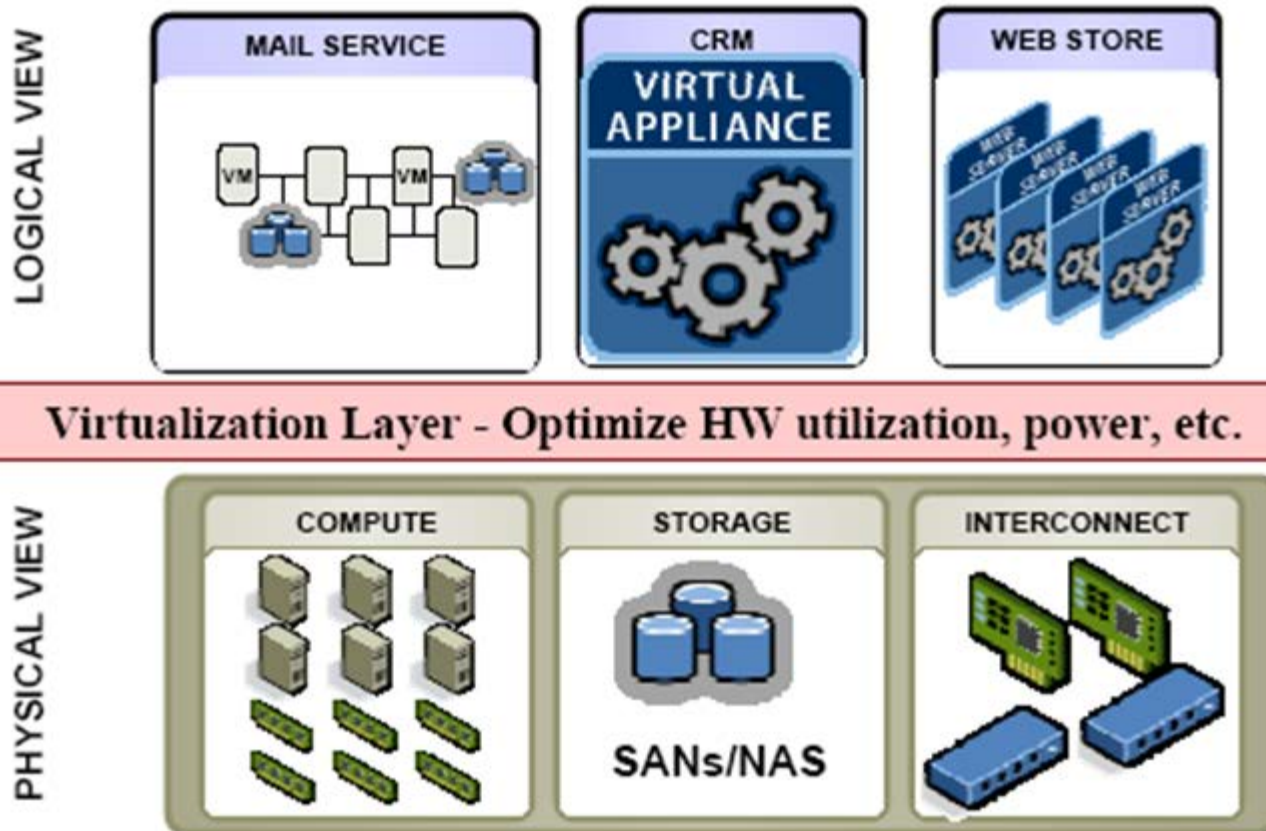
Definitions

- **Host Operating System:**
  - The operating system actually running on the <u>hardware</u>
  - Together with virtualization layer, it simulates environment for …

- **Guest Operating System:**
  - The operating system running in the <u>simulated environment</u>
  - To do some things or resource allocation

A Taxonomy of Virtual Machines

Process VMs | System VMs

Same ISA — Multiprogrammed Systems — Dynamic Binary Optimizers

Different ISA — Dynamic Translators — HLL VMs

Same ISA — Classic-System VMs — Hosted VMs

Different ISA — Whole-System VMs — Codesigned VMs

high-level language virtual machine (HLL VM)
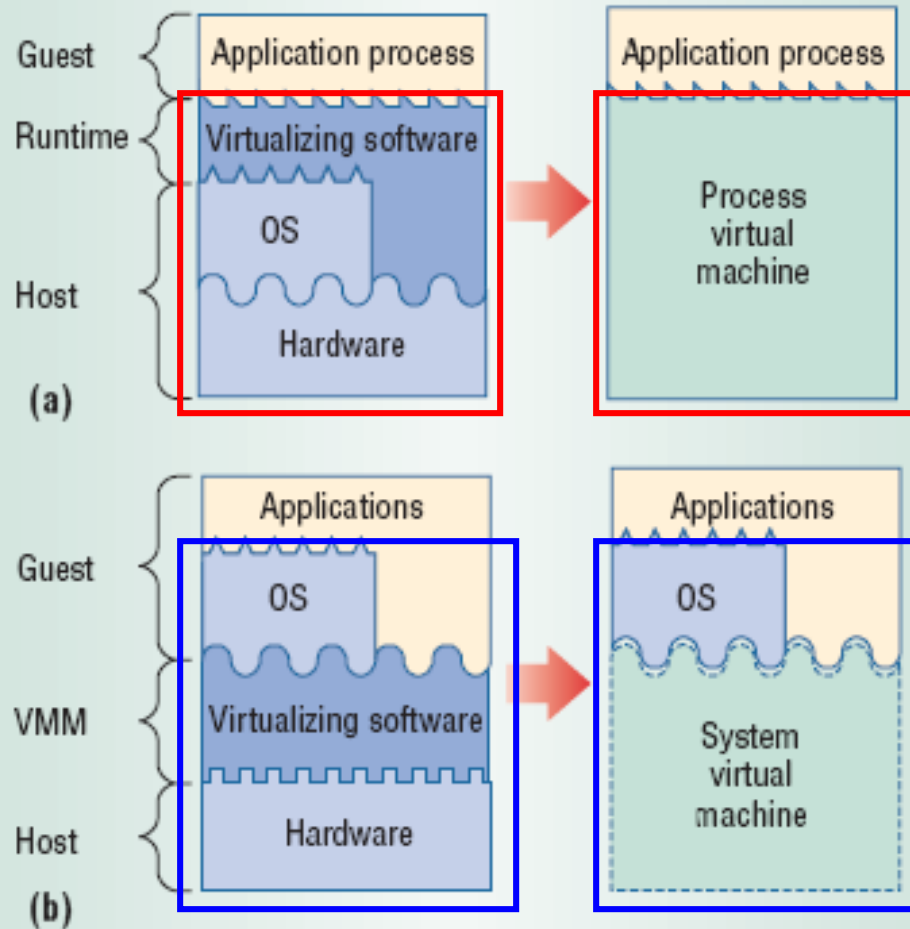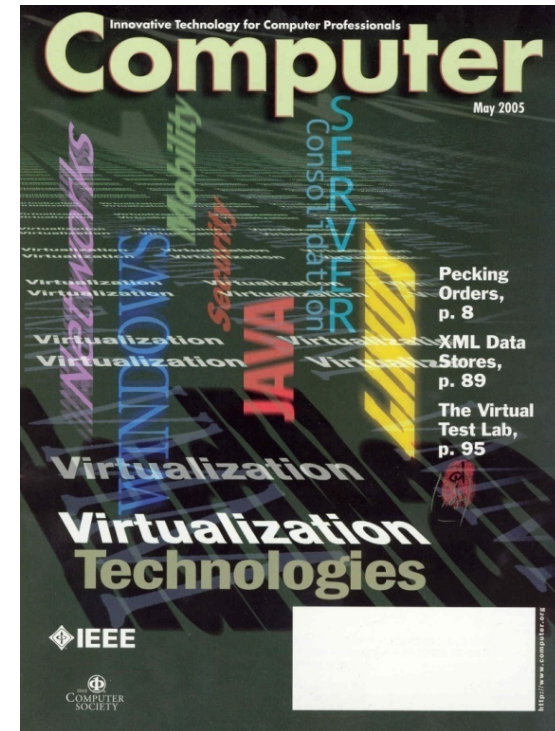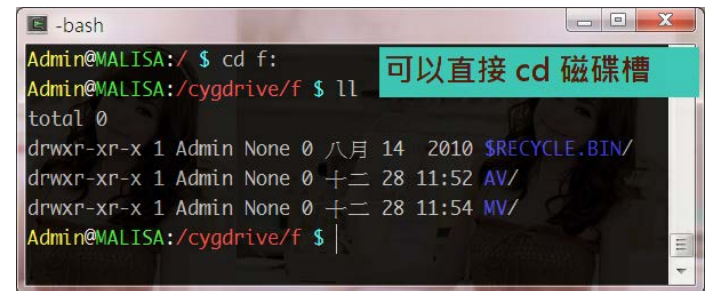
Process vs. System VMs

Figure 3. Process and system VMs. (a) In a process VM, virtualizing software translates a set of OS and user-level instructions composing one platform to those of another. (b) In a system VM, virtualizing software translates the ISA used by one hardware platform to that of another.

- In Smith and Nair's "The architecture of Virtual machines", Computer, May 2005

System/Process Virtual Machines

- **Can view virtual machine as:**
  - System virtual machine (i.e. Similar cygwin)
    - Full execution environment that can support multiple processes
    - Support I/O devices
    - Support GUI

  - Process virtual machine
    - Virtual machines can be instantiated for a single program (i.e. Similar Java)
    - Virtual machine terminates when process terminates.

Must Virtual Machine be Replica of Host Machine?

- No, virtualization layer can simulate any architecture
    - Typically used for debugging specialized systems
    - Real-time systems, niche products, etc.

- Guest architecture does *not* even have to be real hardware!

Example – Page tables

- Suppose guest OS has its own page tables then virtualization layer must
    - Copy those tables to its own
    - Trap every reference or update to tables and simulate it
- During page fault
    - Virtualization layer must decide whether fault belongs to guest OS or self
    - If guest OS, must simulate a page fault
- Likewise, virtualization layer must trap and simulate every privileged instruction in machine!

Virtual Machines (cont.)

- **The resources of the physical computer are shared to create the virtual machines**
  - CPU scheduling can create the appearance in which each user has own processor
  - A normal user time-sharing terminal serves as the virtual machine operator's console
  - Spooling and a file system provide
    - virtual card readers, virtual line printers
  - Disk partitioned to provide virtual disks

Spool : simultaneous peripheral operations on-line

Virtual Machines (cont.)

- Virtual-machine concept provides complete protection of system resources
  - Each **virtual machine** is <u>isolated</u> from all other virtual machines.
  - However, it does not directly share the resources.
    - Virtualization layer
- Virtual-machine system is a good vehicle for operating-systems research and development.
  - System development is done on the virtual machine does not disrupt normal operation.
  - Multiple concurrent developers can work at same time.

Virtual Machines (cont.)

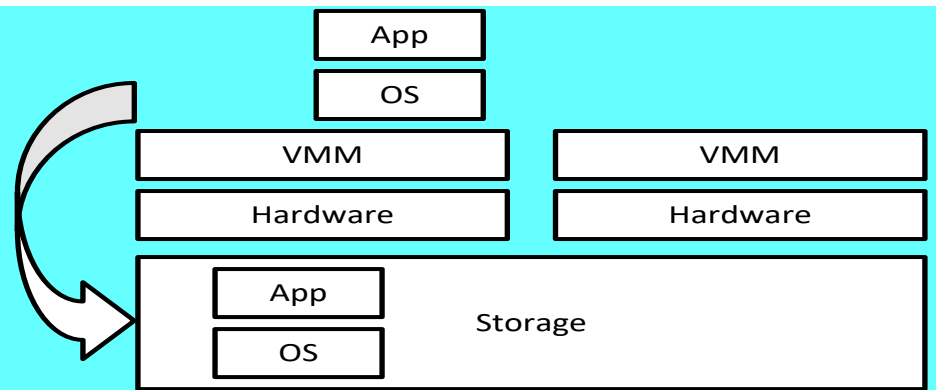- Some hardware architectures or features are impossible to virtualize
  - Certain registers or state not exposed
  - Unusual devices and device control
  - Clocks, time, and real-time behavior
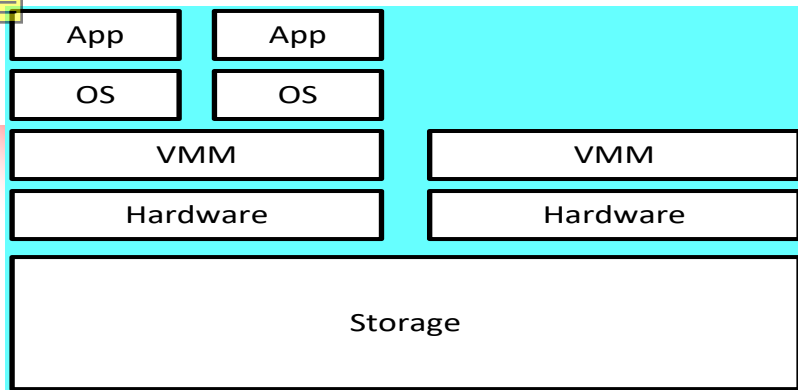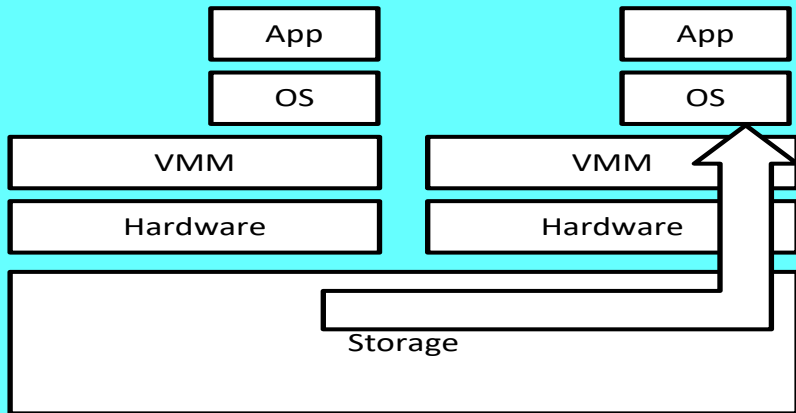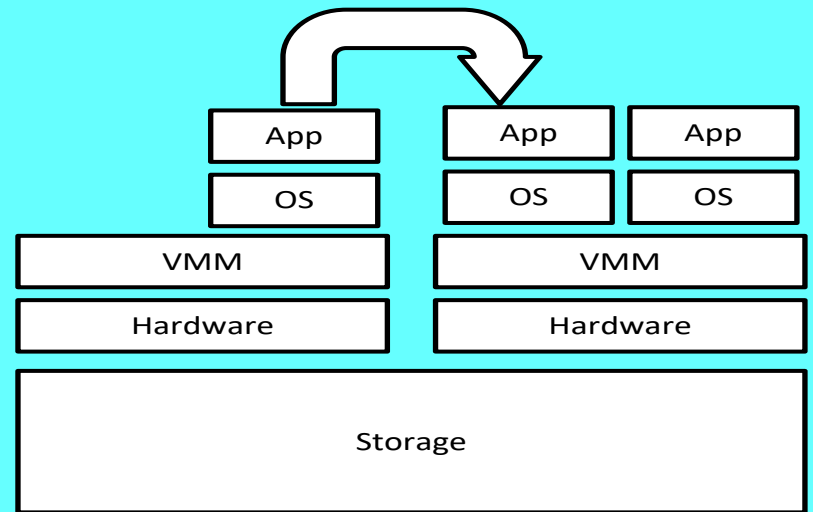
Figure 3.2   Virtual machine multiplexing, suspension, provision, and migration in a distributed computing environment

Virtualization at Hardware Abstraction Level

Hypervisor

- Virtualization is performed right on top of the bare metal hardware.
- It generates virtual hardware environments for VMs.
  - Xen, Hyper V, KVM, Virtual PC, Denali.

- Advantage: it has higher performance and good application isolation.

- Shortcoming and limitations: very expensive to implement due to its complexity.

Hypervisor

- A hypervisor is a hardware virtualization technique allowing multiple operating systems, called guests, to run on a host machine.
    - It also called the Virtual Machine Monitor (VMM).

- Type 1 hypervisor or the bare metal hypervisor sits on the bare metal computer hardware like the CPU, memory, etc.
- All the guest operating systems are a layer above the hypervisor.
- The hypervisor is the first layer over the hardware, such as the original CP/CMS hypervisor developed by IBM.
    - An example is Microsoft Hyper-V

Hypervisor

- Type 2 or the hosted hypervisor does not run over the bare metal hardware, but over a host operating system.

- The hypervisor is the second layer over the hardware.

- The guest operating systems runs a layer over the hypervisor and form the third layer.
  - An example is FreeBSD.
  - The operating system is usually unaware of the virtualization

Full virtualization vs. Para-virtualization

- **Full virtualization** <u>does not need to modify guest OS</u>, and critical instructions are emulated by software through the use of binary translation.

- **Para-virtualization** <u>needs to modify guest OS</u>, and non-virtualizable instructions are replaced by hypercalls that communicate directly with the hypervisor or VMM.
  - As of full virtualization, the advantage is not having to modify OS.
  - However, this approach of binary translation slows down the performance considerably.

Full virtualization vs. Para-virtualization

- Para-virtualization reduces the overhead, but the cost of maintaining para-virtualized OS is high.
  - The improvement depends on the workload.
- VMware Workstation applies full virtualization, which uses binary translation to automatically modify x86 software on-the-fly to replace critical instructions.
- The para-virtualization is supported by Xen, Denali and VMware ESX

Full Virtualization
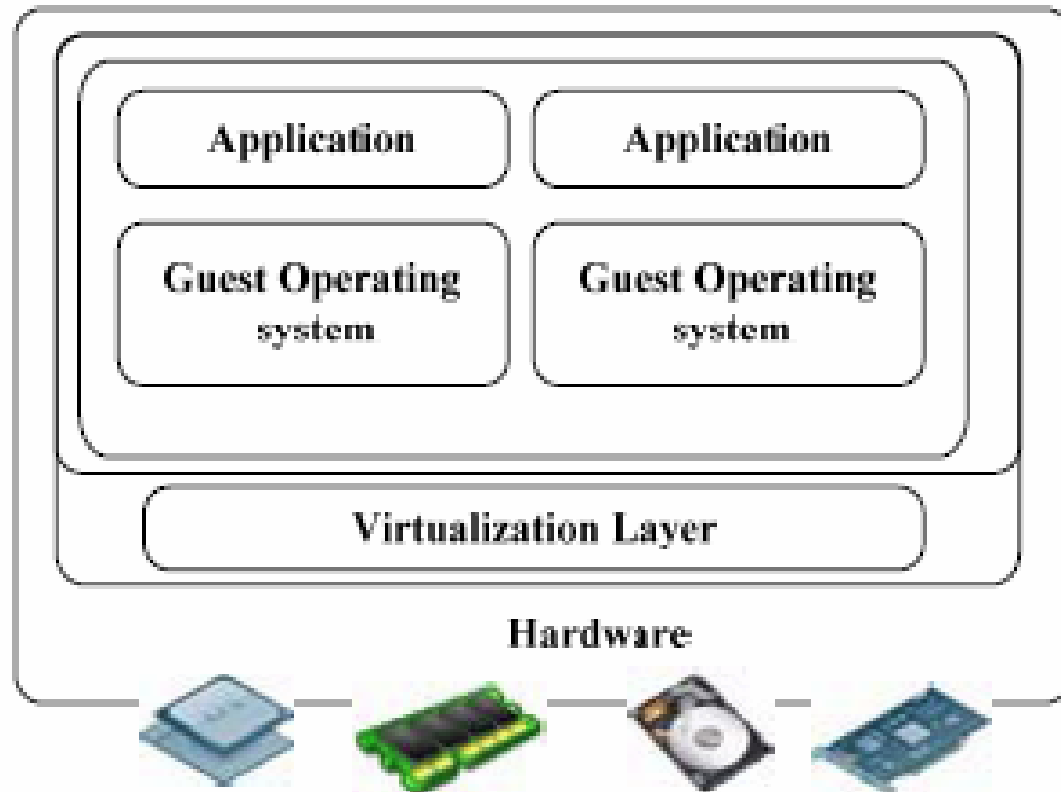
Figure 6.9 The concept of full virtualization using a hypervisor or a VMM directly sitting on top of the bare hardware devices. Note that no host OS is used here as in Figure 6.11.

Virtual Machine Architectures

Virtual Machine	**Virtual Machine**
Application	Application
Guest OS	VMM

| Application | | | Nonprivileged user space |

| Host OS | Hypervisor | Host OS | Privileged system space |
| Hardware | Hardware | Hardware | |

(a) Physical Machine   (b) Native VM   (c) Hosted VM using VMM

Figure 1.20 Three major components of a conventional physical machine is shown in Part (a). Native VM shown in Part (b) is created by a hypervisor sitting on top of bare metal hardware. A hosted VM shown in Part (c) is created by a VMM middleware in cooperation with the host OS.

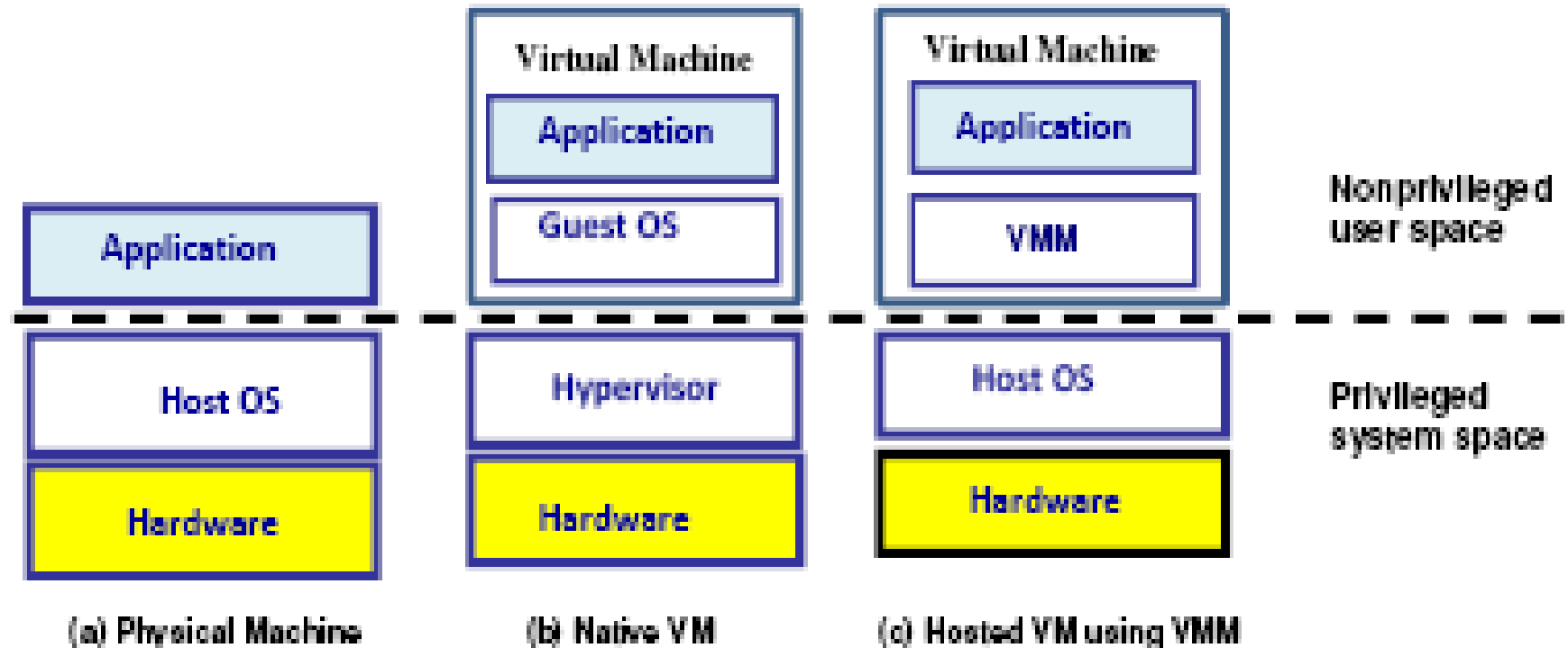## Table 3.4
Hypervisors or VM monitors for generating VMs

Hypervisor	Host CPU	Host OS	Guest OS	Architecture, Applications, and User Community
XEN	x86, x86–64, IA-64	NetBSD, Linux	Linux, Windows, BSD, Solaris	Native hypervisor (Example 3.1) developed at Cambridge University
KVM	x86, x86–64, IA-64, S-390, PowerPC	Linux	Linux, Windows, FreeBSD, Solaris	Hosted hypervisor based on paravirtualization at the user space
Hyper V	x86 based	Server 2003	Windows servers	Windows-based native hypervisor, marketed by Microsoft
VMWare Workstation, VirtualBox	x86, x86–64	Any host OS	Windows, Linux, Darwin Solaris, OS/2, FreeBSD	Hosted hypervisor with a paravirtualization architecture

Hypervisor and the XEN Architecture

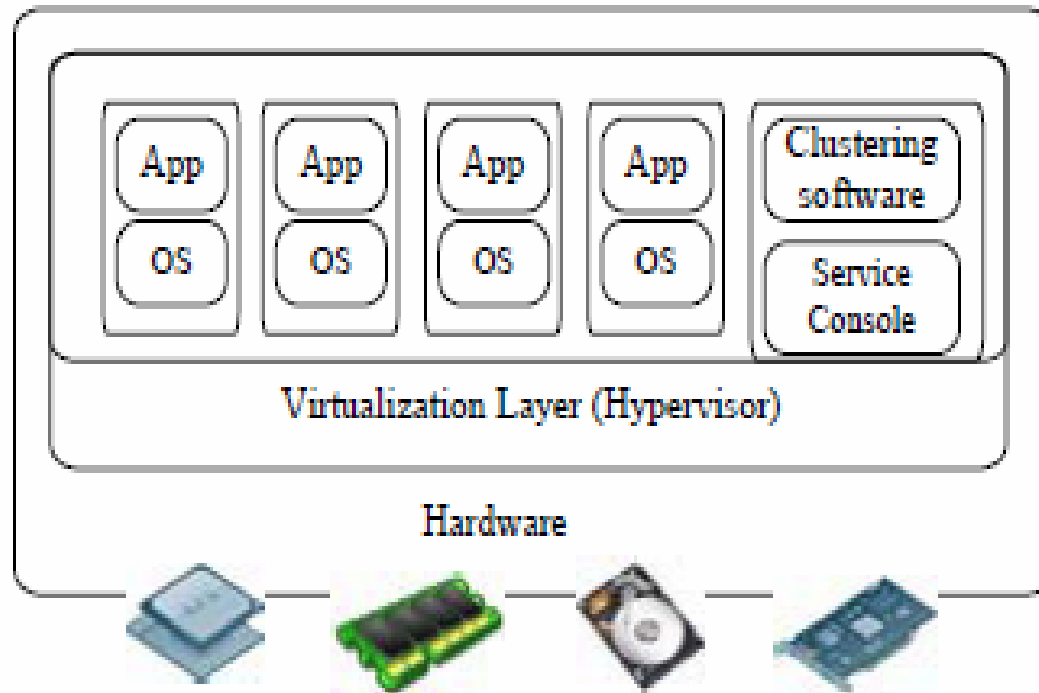Figure 6.7   A hypervisor is the software layer for virtualization of the bare metal hardware. This layer can be implemented as a micro-kernel of the OS. It converts physical devices into virtual resources for user applications to run on the virtual machines deployed.
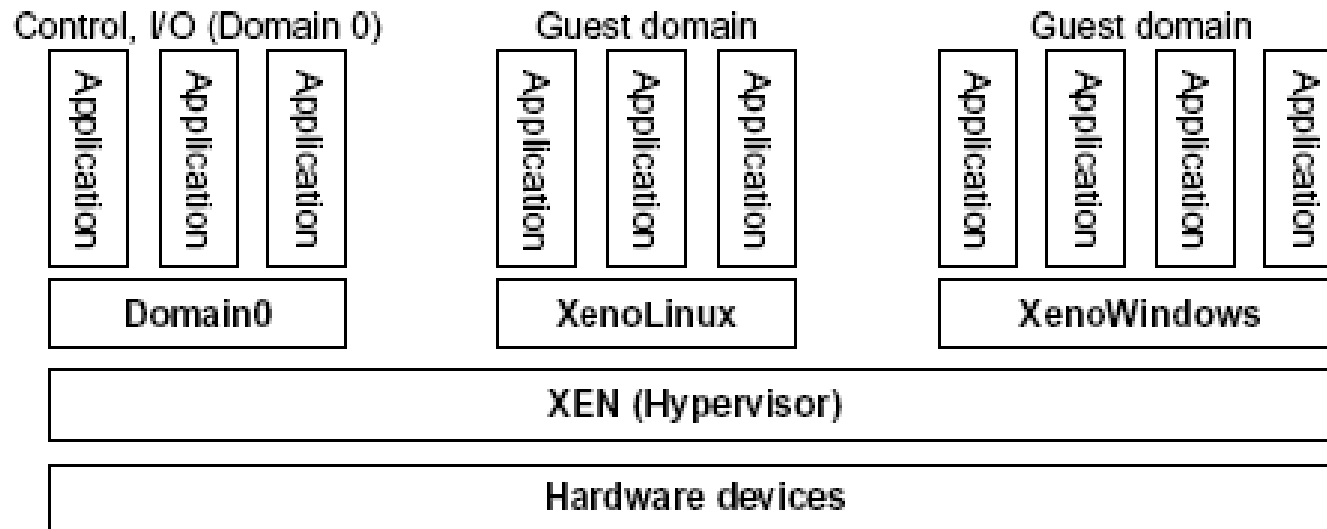
The XEN Architecture (1)

FIGURE 3.5

The Xen architecture's special domain 0 for control and I/O, and several guest domains for user applications.

The XEN architecture (2)

- **Xen Project** (pronounced /ˈzɛn/) is a hypervisor using a microkernel design, providing services that allow multiple computer operating systems to execute on the same computer hardware concurrently.
- It was developed by the University of Cambridge and is now being developed by the Linux Foundation with support from Intel.
- The Xen Project community develops and maintains Xen Project as free and open-source software.
- Xen Project is currently available for the IA-32, x86-64 and ARM instruction sets.
- Xen Project runs in a more privileged CPU state than any other software on the machine.

The XEN Architecture (3)

- Responsibilities of the hypervisor include memory management and CPU scheduling of all virtual machines ("domains"), and for launching the most privileged domain ("**dom0**") - the only virtual machine which by default has direct access to hardware.

- From the dom0 the hypervisor can be managed and **unprivileged domains ("domU")** can be launched.

The XEN Architecture (4)

- The dom0 domain is typically a version of <u>Linux</u> or <u>BSD</u>.

- User domains may either be traditional operating systems, such as <u>Microsoft Windows</u> under which privileged instructions are provided by hardware virtualization instructions (if the host processor supports <u>x86 virtualization</u>, e.g., <u>Intel VT-x</u> and <u>AMD-V</u>), or *para-virtualized* operating systems whereby the operating system is aware that it is running inside a virtual machine, and so makes hypercalls directly, rather than issuing privileged instructions.

- Xen Project boots from a <u>bootloader</u> such as <u>GNU GRUB</u>, and then usually loads a <u>paravirtualized</u> host operating system into the host domain (dom0).

VMware hardware virtualization (hypervisors) and hosted software for virtualization

Category	VMware software products or third party software
Native (Hypervisor)	Adeos, CP/CMS, Hyper-V, KVM (Red Hat Enterprise Virtualization), LDoms / Oracle VM Server for SPARC, LynxSecure, SIMMON, VMware ESXi (VMware vSphere, vCloud), VMware Infrastructure, Xen XenClient), z/VM
Hosted (Specialized)	Basilisk II, bhyve, Bochs, Cooperative Linux, DOSBox, DOSEMU, LLinux, Mac-on-Linux, Mac-on-Mac, SheepShaver, SIMH, Windows on Windows, Virtual DOS machine, Win4Lin
Hosted (Independent)	Microsoft Virtual Server, Parallels Workstation, Parallels Desktop forMac, Parallels Server for Mac, PearPC, QEMU, VirtualBox, Virtual Iron, VMware Fusion, VMware Player, VMware Server VMware Workstation, Windows Virtual PC
Other tools	Ganeti, oVirt, Virtual Machine Manager

(http://vmware.com/products/vsphere/, 2016)

VMWare ESX Server for Para-virtualization
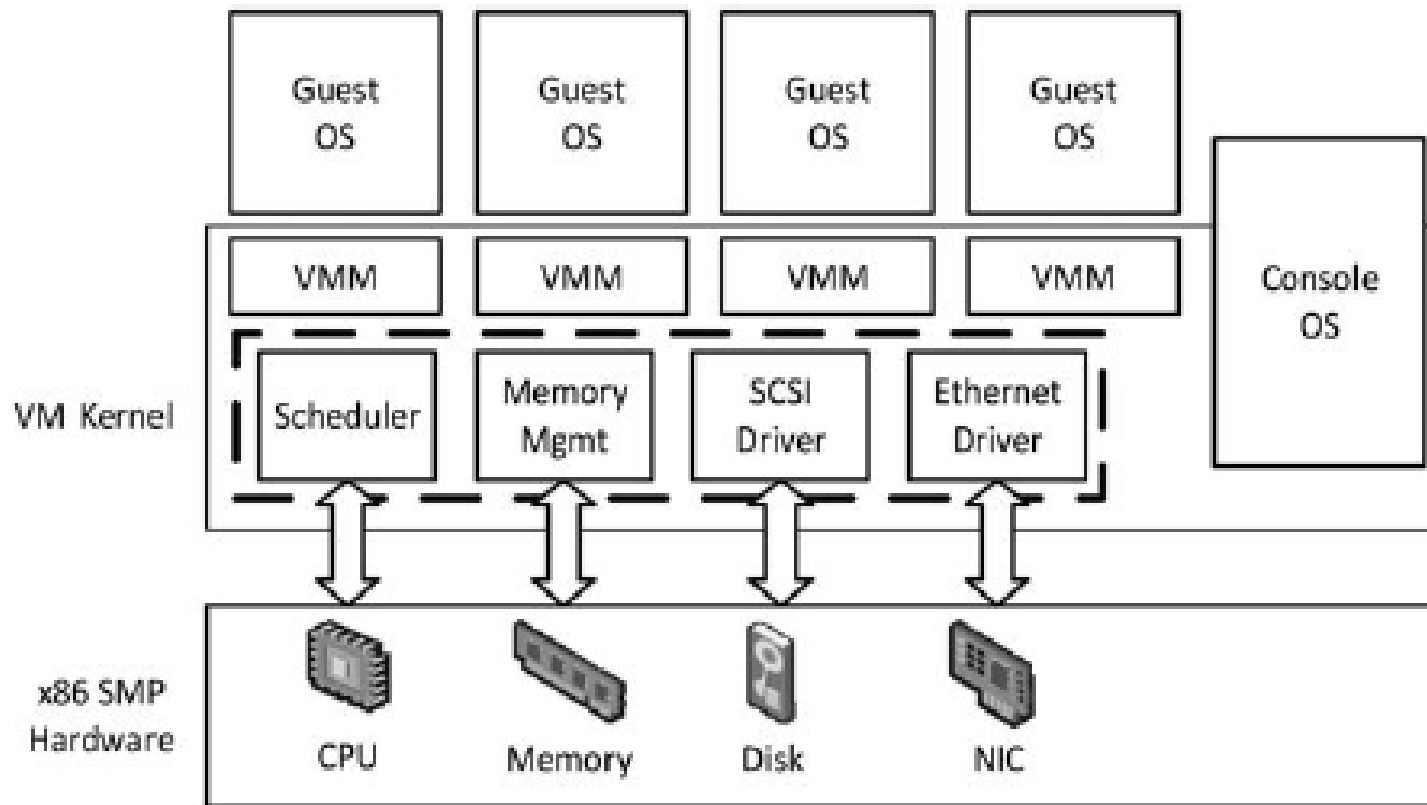
Figure 3.11

The VMware ESX server architecture using paravirtualization. Courtesy of VMware, 2011, http://www.vmware.com/products.

Virtual networks

- Logical links:
  - multiple physical links, routing via native Internet routing
  - tunneling, virtual routers, switches, …
  - partial to total isolation

Slide provided by M. Tsugawa
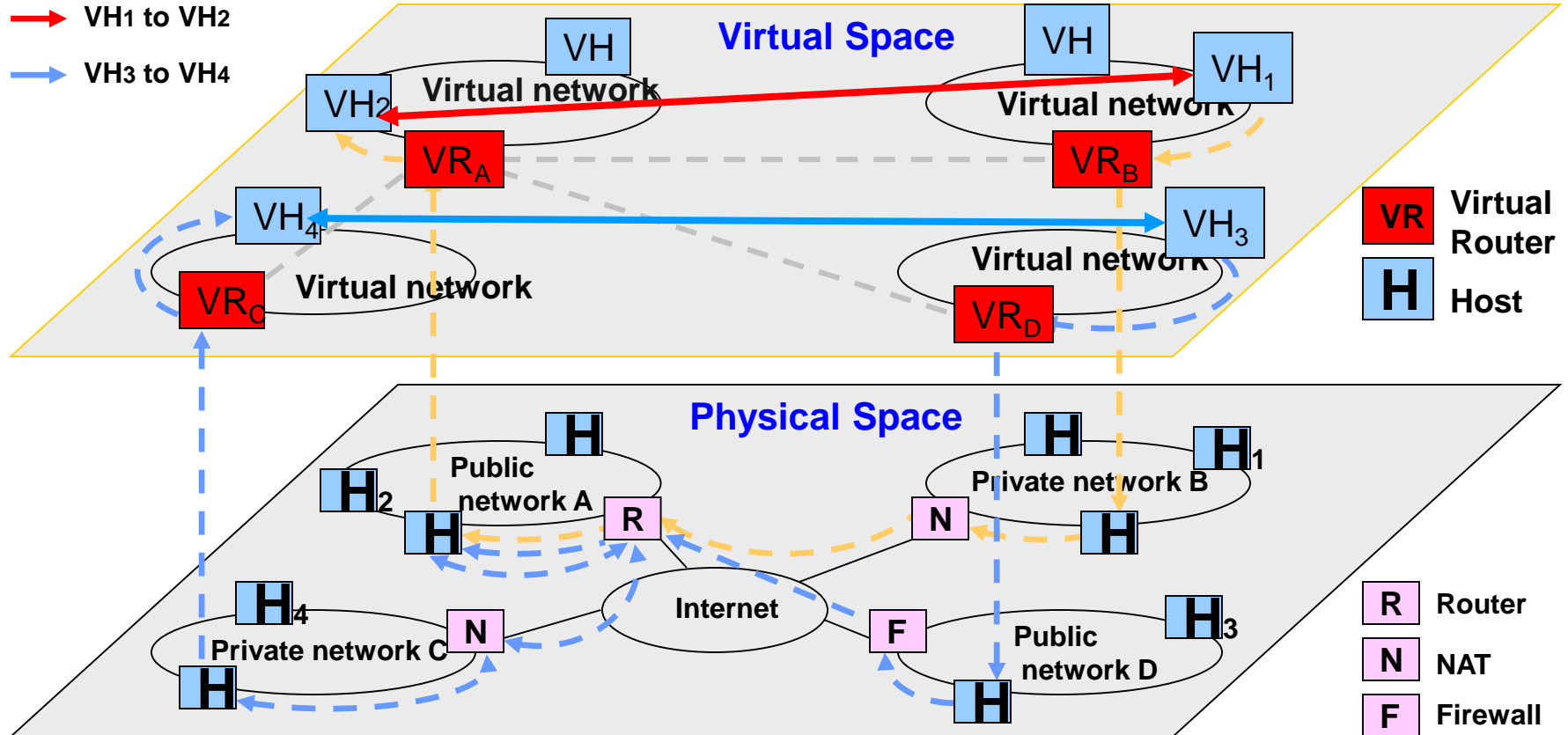
Virtualization Data/File

Compute Server 'C'　　　　　　　　　　　　　　**NFS Server 'S'**

Mountd

NFS Client　　　　　　　　　**NFSD**　　**Export /home to all uids on compute server C**

mount S:/home　　◀ **Client**　**Server** ▶

GVFS Proxy　　**Mountd**

NFS Client　　**GVFS Proxy**　**NFSD**　　**Export /home/user_A to shadow1 on C**

mount S:/home/user_A　　◀ **Client**　**Server** ▶
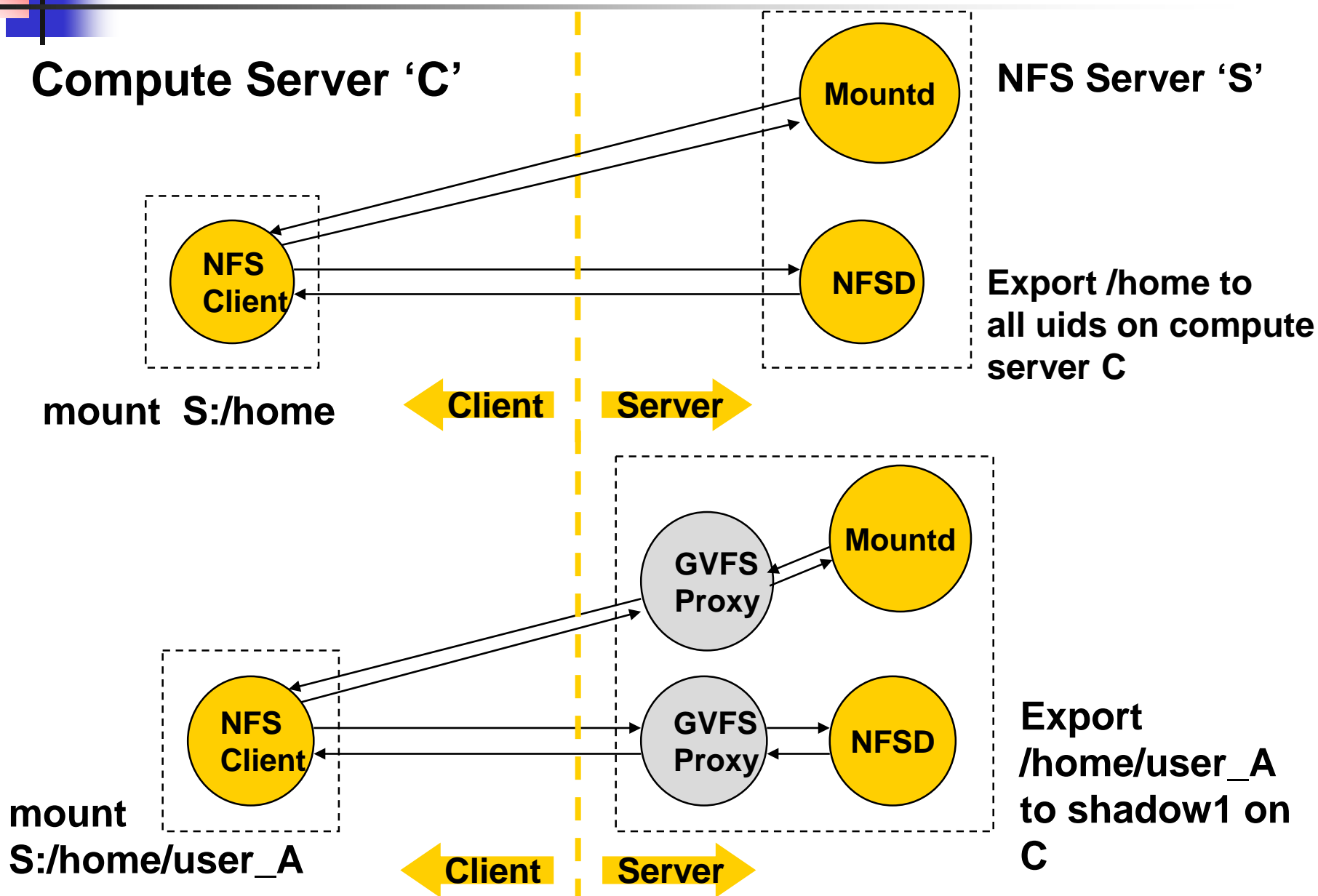
Grid Virtual File System (GVFS)

- Originally named PVFS, is a virtualized distributed file system

  - providing high-performance data access in grid environments and seamless integration with unmodified applications.

- It leverages existing NFS (Network File System) support in operating systems, and uses user-level proxies to authenticate and forward RPC (Remote Procedure Call) requests between the native NFS client and server, and map user identities between different domains.

A Grid-building Recipe

❶Virtualize to fit needed environments

❷Use services to generate "virtuals"

❸Aggregate and manage "virtuals"

❹Repeat ❶ ❷ ❸ as needed

- The result:
  - Users interact with virtual entities provided by services
  - Middleware interacts with physical resources

Architectural Components of VM Service

VM Creation Request from Client

(1) **VM Request**

(6) **VM ID**

VMShop (VMArchitect VMCreator, VMCollector, VMReporter)

(2) **Request Estimate**

(3) **VM Creation Cost**

(4) **Create VM**

(5) **VM ID**

| mcnabb | VMPlant Daemon | brady | mcnair | manning | VMPlant Daemon | favre | VMPlant Daemon |
| vws010 | | vws001 | vws002 | vws003 | | vws005 | |

Host OS (VMPlant)

Host OS (VMPlant)

Host OS (VMPlant)

Slide provided by Arijit Ganguly
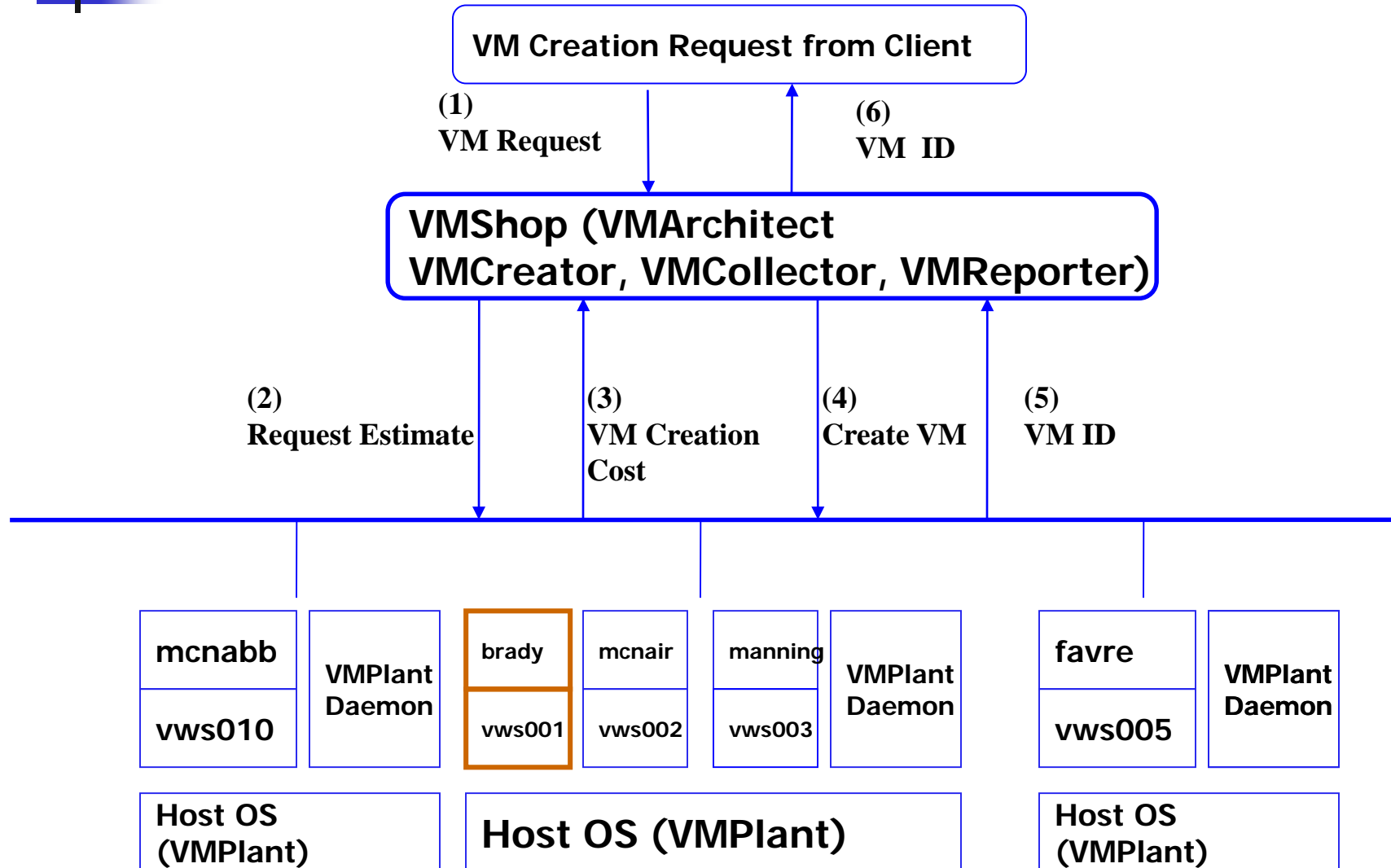
Create VM Steps

1. ## Clone VM
   - Instantiate a new container
   - Fast copying of a base VM image
     - Virtual disk
     - Suspended memory (if available)

2. ## Configure VM
   - Execute scripts/jobs inside container to tailor to a particular instance
   - Communication crossing container boundaries to provide inputs/retrieve outputs

3. ## Destroy VM
   - Terminate container, delete non-persistent state

User-level Extensions
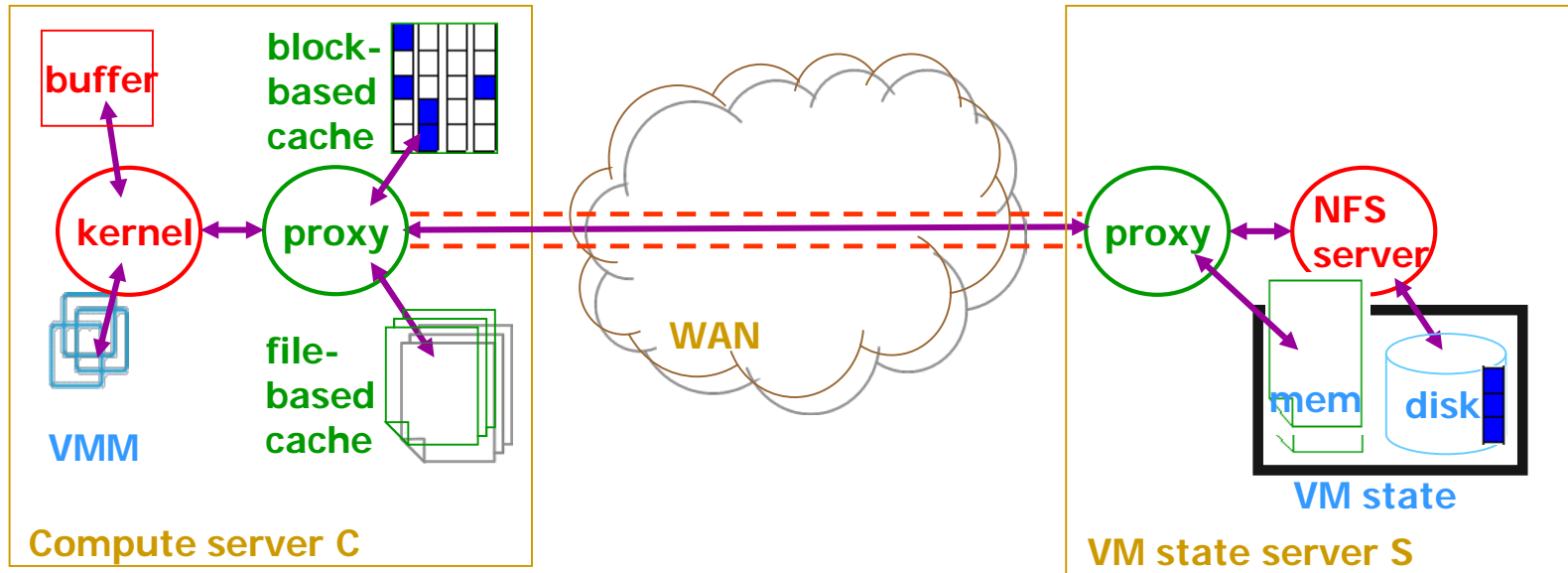
- **Client-side proxy disk caching**
- **Application-specific meta-data handling**
- **Encrypted file system channels and cross-domain authentication**
- *[Zhao, Zhang, Figueiredo, HPDC'04]*

Docker Engine and Containers

- Docker engine works with host OS to produce application software containers

- The App software container does not use a guest OS, thus it is highly scalable

- Virtual clusters are studied with the use either VMs or containers

Virtualization at the Operating System Level

- An abstraction layer between traditional OS and user.
- This virtualization creates isolated containers on a single physical server and the OS-instance to utilize the hardware and software in datacenters.
  - Typical systems: Jail / Virtual Environment / Ensim's VPS / FVM
- Advantage: has minimal startup/shutdown cost, low resource requirement, and high scalability; synchronizes VM and host state changes.

- Shortcoming and limitation: all VMs at the operating system level must have the same kind of guest OS; poor application flexibility and isolation.

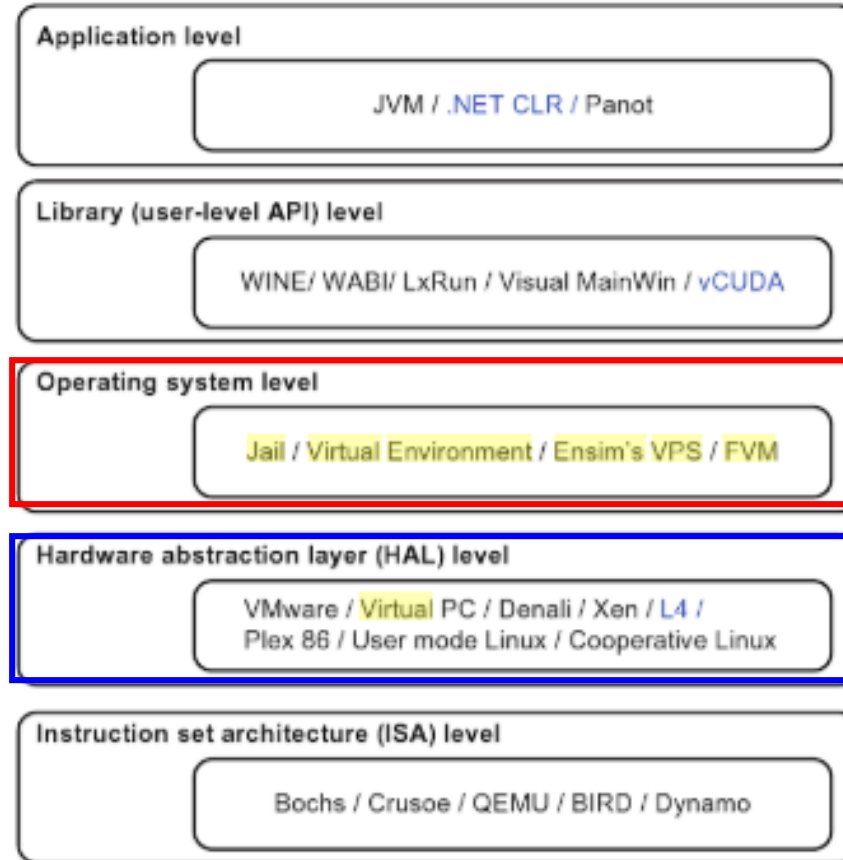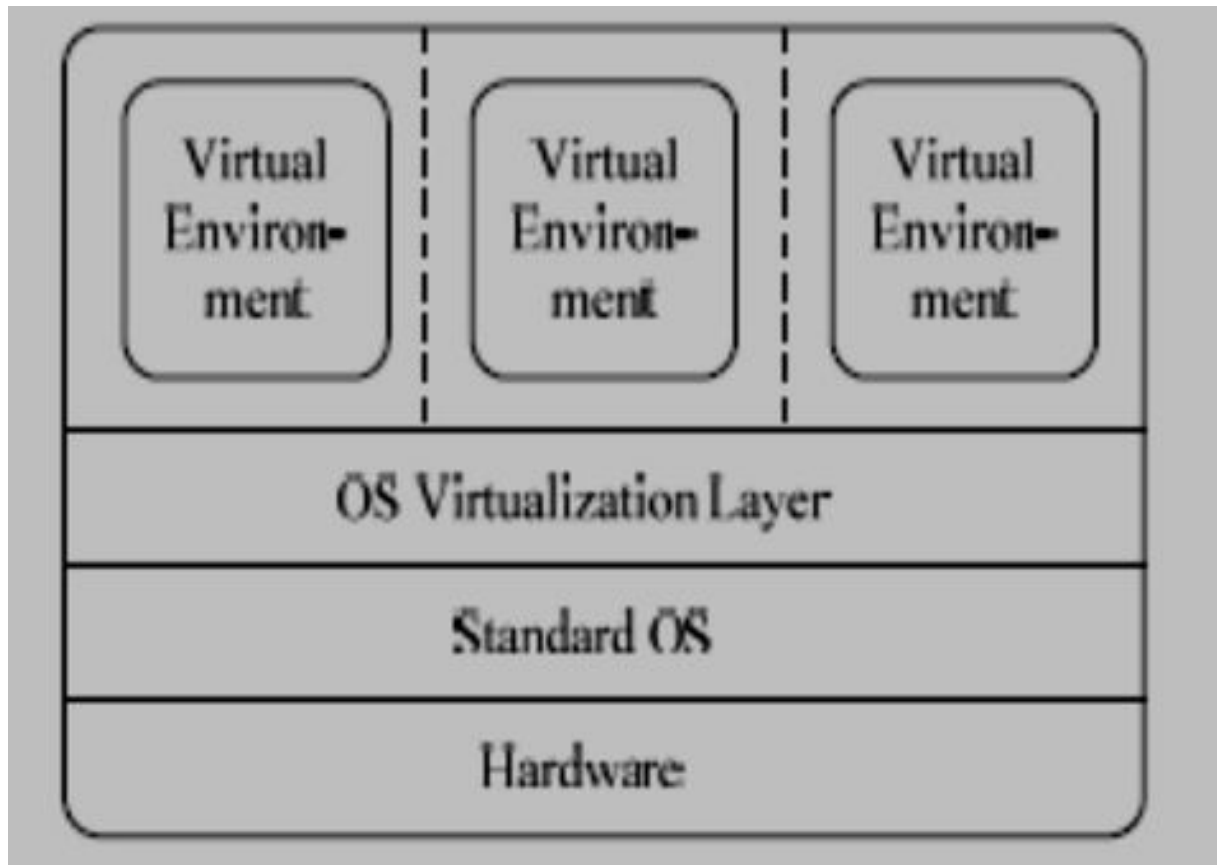# Virtualization Ranging from Hardware to Applications

FIGURE 3.2

Virtualization ranging from hardware to applications in five abstraction levels.

Virtualization at OS level

Advantages of OS extension for virtualization

1. VMs at OS level has minimum startup/shutdown costs
2. OS-level VM can easily synchronize with its environment
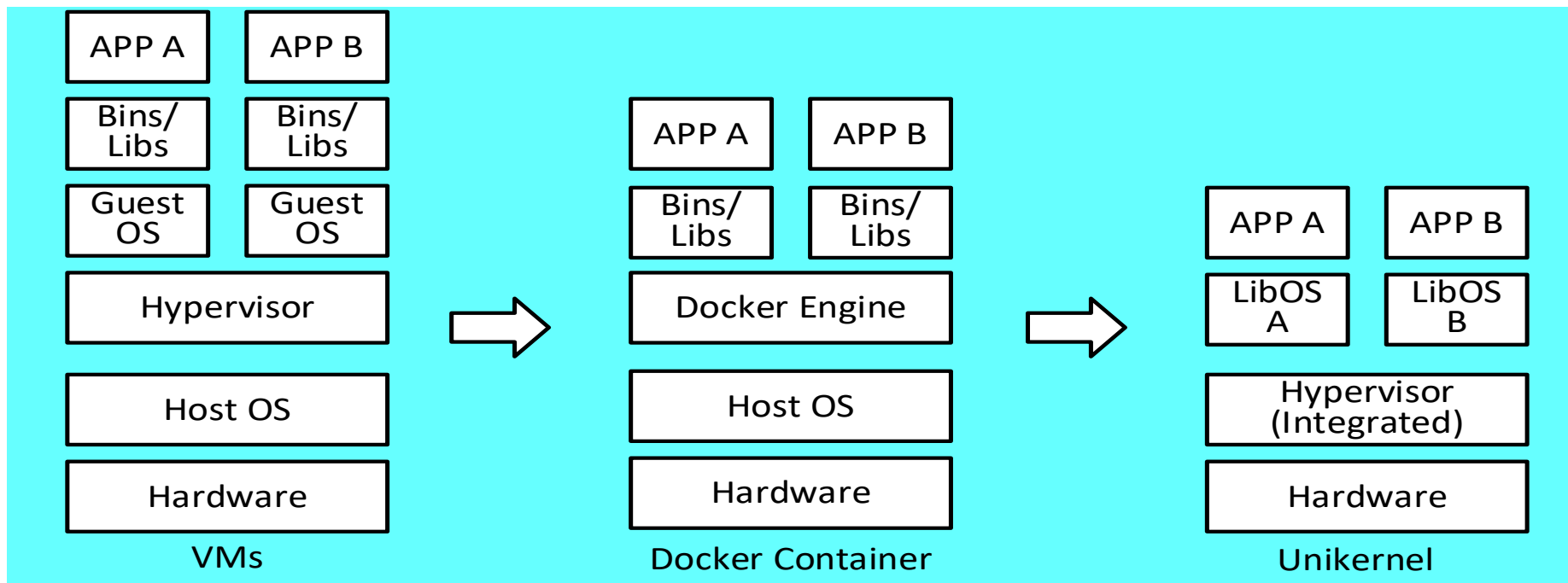
Disadvantage of OS Extension for Virtualization

All VMs in the same OS container must have the same or similar guest OS, which restricts application flexibility of different VMs on the same physical machine

Assessing the use of virtual machines and docker containers in today's clouds

- Both VMs and software containers will co-exist for some time in today's clouds.

- The VMs have high software portability on different types of hardware platforms.

- VMs are heavily weighted with the use of heavy duty guest OS.

  - This may weaken its acceptance in the future.

- The docker containers are light-weight and more cost-effective to implement and to apply in scalable applications.

  - Eventually, most clouds will use containers over Linux hosts

Architectural Evolution from VMs to Docker Containers and Unikerkels

VMs	Docker Container	Unikernel
APP A · APP B	APP A · APP B	APP A · APP B
Bins/Libs · Bins/Libs	Bins/Libs · Bins/Libs	LibOS A · LibOS B
Guest OS · Guest OS	Docker Engine	Hypervisor (Integrated)
Hypervisor	Host OS	Hardware
Host OS	Hardware	
Hardware		

Autonomic Cloud Management

Develop methodologies and tools to automate the process of cloud management in 4 objectives

1. Manage resources to provisioning of service quality assurance and adaptation

3. Manage energy consumption under SLA constraints

- Resource Management
- Power Management
- Capacity Management
- Autonomic Cloud Management
- Load Balancing
- Admission Control
- Reliability Management

2. Automate the configuration process of VMs and virtual clusters

4. Develop fault prediction models for proactive failure management

Virtualization for Datacenter Automation to serve millions of clients, simultaneously

- Server consolidation in virtualized datacenter

- Virtual storage provisioning and deprovisioning

- Cloud operating systems for virtual datacenters

- Trust management in virtualized datacenters

Cloud Software Packages and Features

Software	Cloud Type	License(s)	Language	Linux/ Windows	EC2/S3	Xen/KVM/ VMWare	Virtual Box	OCCI/ vCloud
Fluid Operations	Iaas, Paas, LaaS, SaaS, TaaS, DaaS, BaaS	Proprietary	Java, C	Yes/Yes	Yes/No	Yes/Yes/ yes	?	No /Yes
ApplScale	Paas	BSD	Python, Ruby, Go	? / ?	Yes/ Yes	Yes/Yes/ yes	Yes	? / ?
Cloud Foundry	PaaS	Apache	Ruby, C	Yes/No	Yes/No	Yes/Yes/ yes	Yes	No /Yes
Cloud.com	IaaS	Proprietary, GPLv3	Java, C	? / ?	? / ?	Yes/Yes/ yes	?	? / ?
Eucalyptus	IaaS	Proprietary, GPLv3	Java, C	Yes/No	Yes/ Yes	Yes/Yes/ yes	?	? / ?
Nimbus	IaaS	Apache	Java, Python	? / ?	Yes/No	Yes/ Yes/?	?	? / ?
OpenNebua	IaaS	Apache	C++,C,Ruby, Java, lex, yacc, Shellscript	Yes/ ?	Yes/ ?	Yes/ Yes/?	?	Yes/Yes
OpenStack	IaaS	Apache	Python	Yes/ ?	Yes/ Yes	Yes/ Yes/?	?	? / ?

Cloud OS for Building Private Clouds

Table 3.6 VI Managers and Operating Systems for Virtualizing Data Centers [9]

Manager/ OS, Platforms, License	Resources Being Virtualized, Web Link	Client API, Language	Hypervisors Used	Public Cloud Interface	Special Features
Nimbus Linux, Apache v2	VM creation, virtual cluster, www .nimbusproject.org/	EC2 WS, WSRF, CLI	Xen, KVM	EC2	Virtual networks
Eucalyptus Linux, BSD	Virtual networking (Example 3.12 and [41]), www .eucalyptus.com/	EC2 WS, CLI	Xen, KVM	EC2	Virtual networks
OpenNebula Linux, Apache v2	Management of VM, host, virtual network, and scheduling tools, www.opennebula.org/	XML-RPC, CLI, Java	Xen, KVM	EC2, Elastic Host	Virtual networks, dynamic provisioning
vSphere 4 Linux, Windows, proprietary	Virtualizing OS for data centers (Example 3.13), www .vmware.com/ products/vsphere/ [66]	CLI, GUI, Portal, WS	VMware ESX, ESXi	VMware vCloud partners	Data protection, vStorage, VMFS, DRM, HA

Eucalyptus: An open-source cloud operating system

- A software platform developed by Eucalyptus Systems, Inc., (started 2008 and stable release 2010)

- Written in Java, C, running with Linux, can host Linux and Windows VMs

- Uses hypervisors (Xen, KVM and VMWare) and compatible with EC2 and S3 services

- Eucalyptus stands for "Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems"

- For use in developing IaaS-style private cloud or hybrid cloud on computer cluster, working with AWS API

- License: Proprietary or GPLv3 for open-core enterprise edition. Open-source edition available

- Website: www.eucalyptus.com

Amazon Web Services (AWS) EC
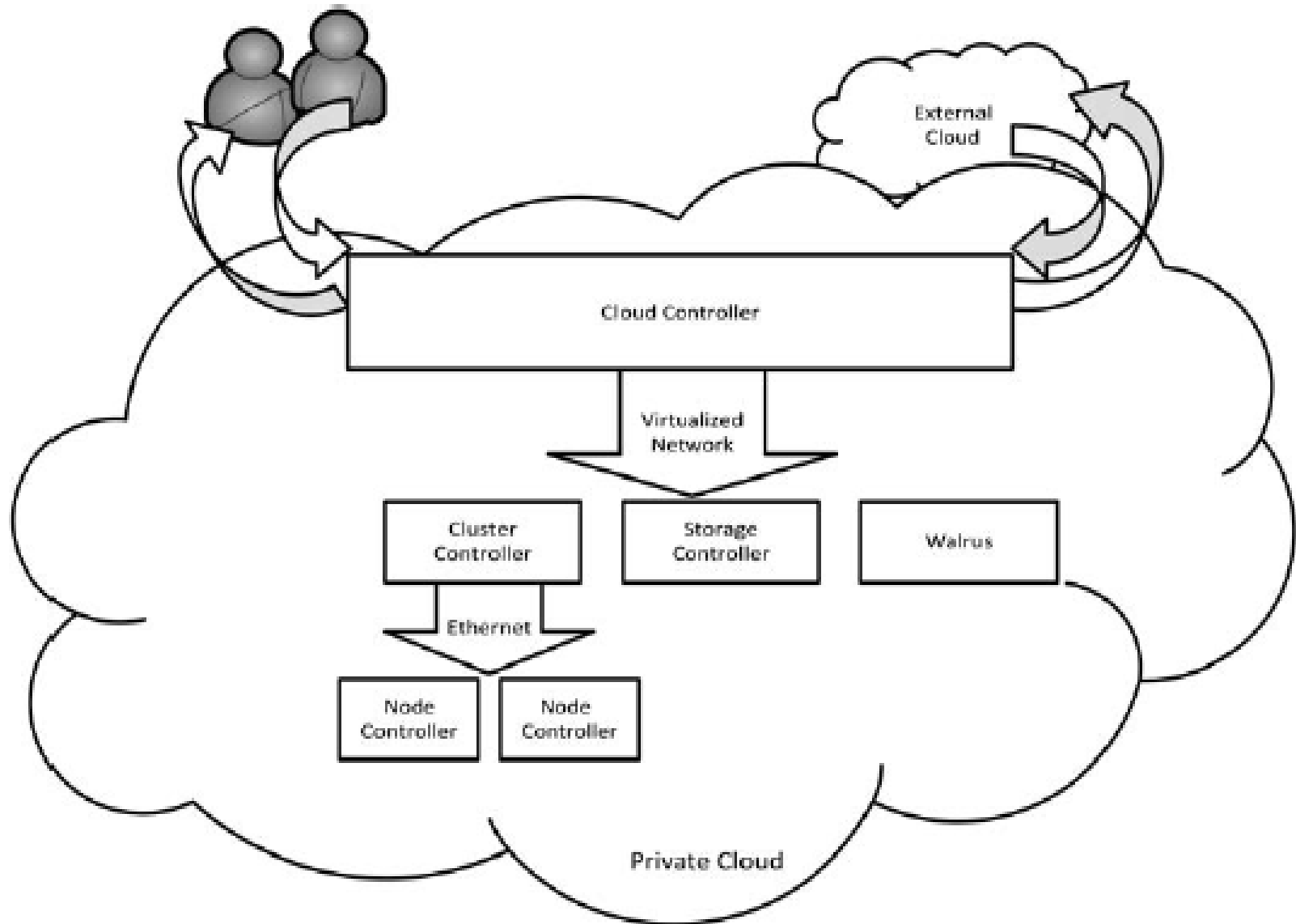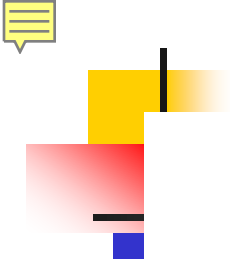
Amazon Simple Storage Service (S3)

Figure 3.22

The Eucalyptus for building private cloud by establishing virtual network over the VMs linking through Ethernet and the Internet.
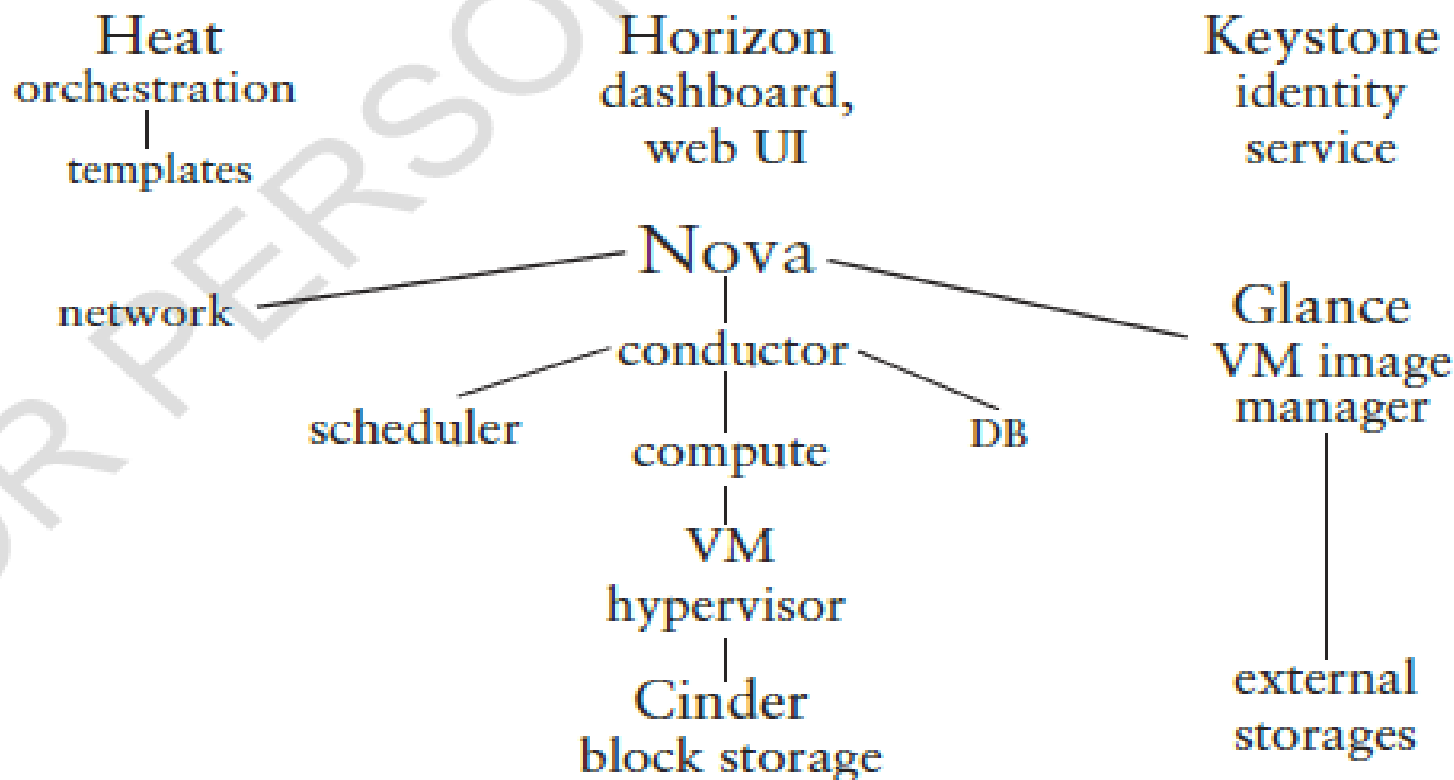
Figure 3.24

OpenStack for constructing private or public clouds in IaaS services. Courtesy of OpenStack, http://openstack .org, Apache License 2.0.

Openstack: An IaaS cloud project launched by Rackspace and NASA in 2010

- Currently, 120 companies have joined Openstack

- Openstack is used to create private cloud and offer cloud computing (Nova), object storage (Swift) and image services (Glance)
- The project offers free open source software under the Apache license.
- The Openstack cloud software is written in Python: http://openstack.org/

Openstack: An IaaS Cloud Library

- Nova site:

   http://openstack.org/projects/compute/
   http://launchpad.net/nova/

- Swift site:

   http://openstack.org/projects/storage/
   http://launchpad.net/swift/

- Glance site:

   http://openstack.org/projects/image-service/
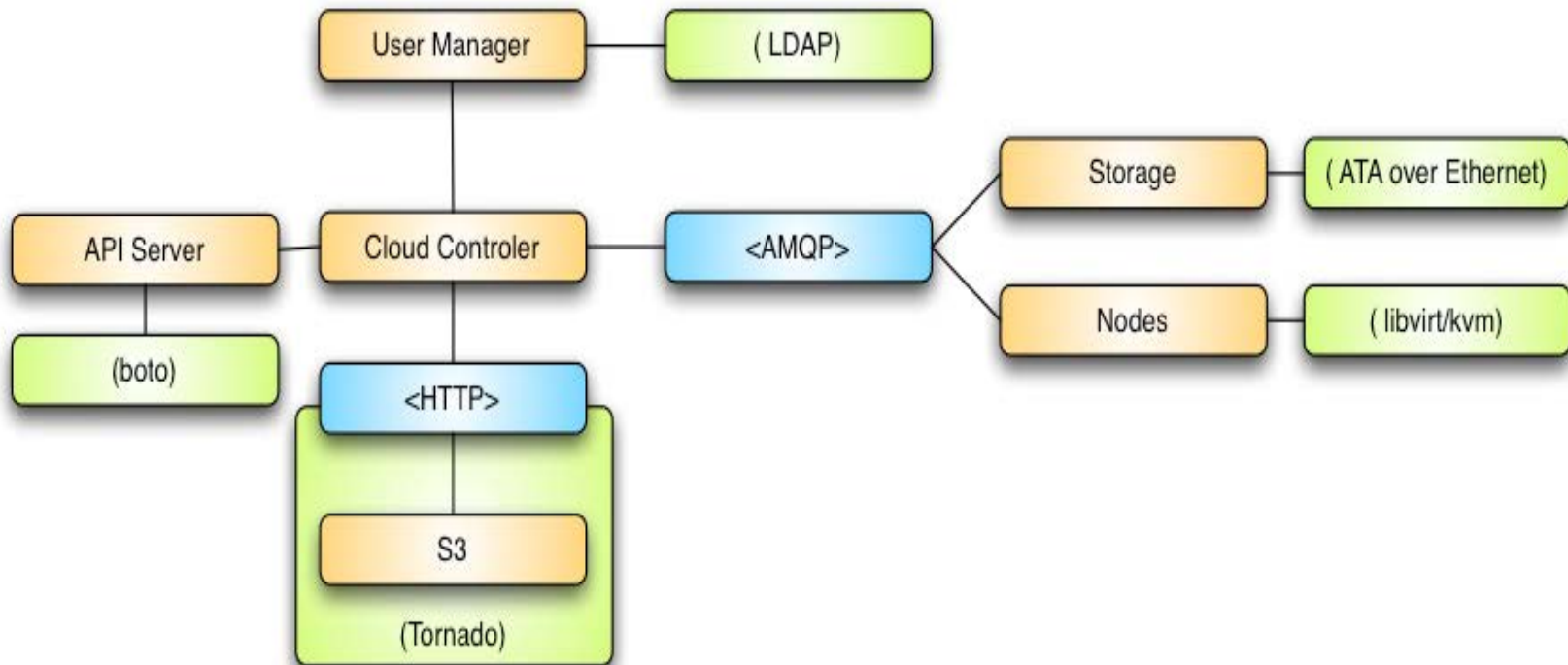   http://launchpad.net/glance/

OpenStack Nova system

Figure 6.36. OpenStack Nova system architecture.

AMQP is an advanced messaging queuing protocol
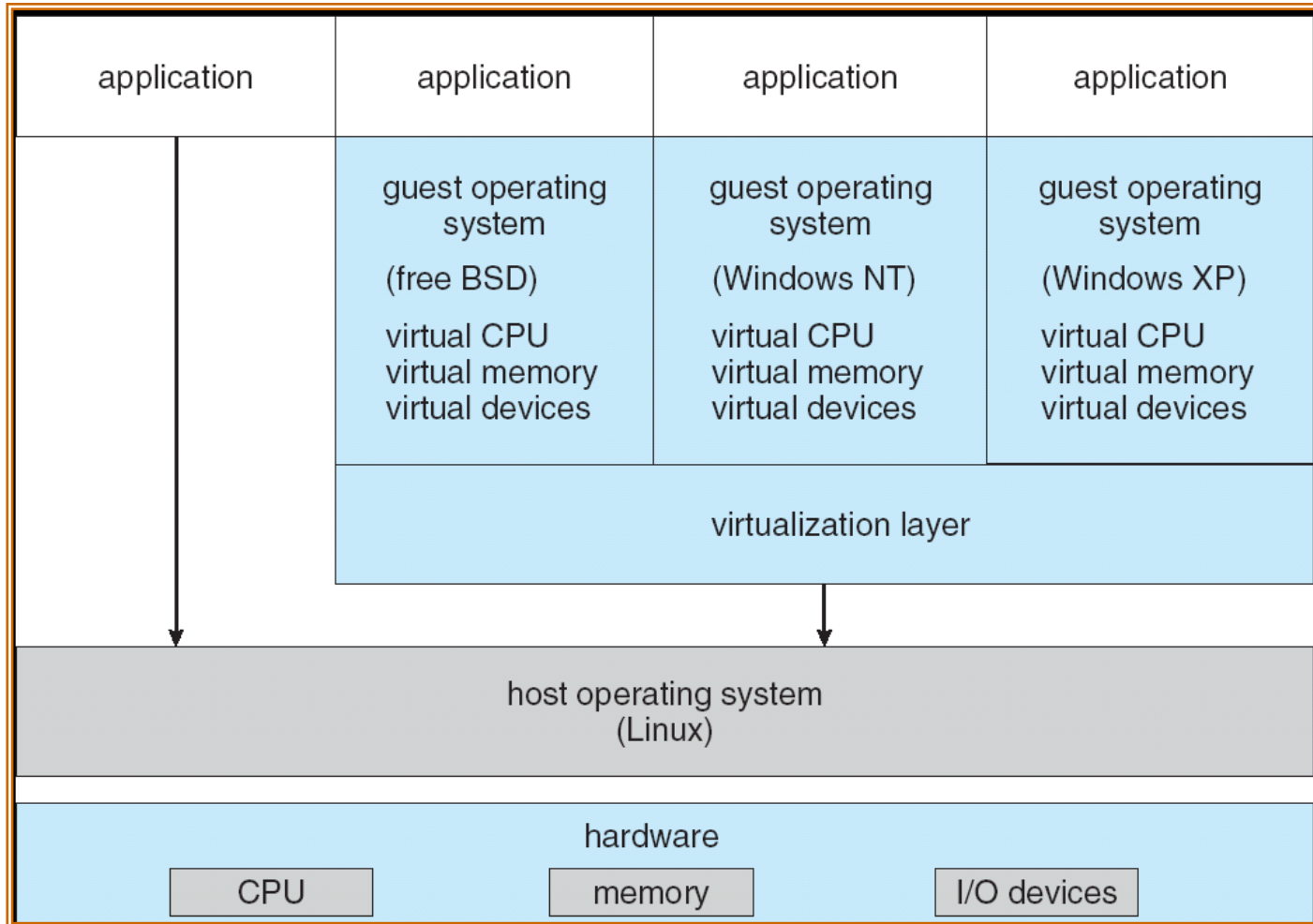
VMware –
Modern Virtual Machine System

- Founded 1998, Mendel Rosenblum *et al.*
    - Research at Stanford University
- VMware Workstation
    - Separates Host OS from virtualization layer
    - Host OS may be Windows, Linux, etc.
    - Wide variety of Guest operating systems
    - < $200
- http://www.vmware.com/

VMware Architecture

application	application	application	application
	guest operating system (free BSD) virtual CPU virtual memory virtual devices	guest operating system (Windows NT) virtual CPU virtual memory virtual devices	guest operating system (Windows XP) virtual CPU virtual memory virtual devices

virtualization layer

host operating system
(Linux)

hardware

CPU   memory   I/O devices

VMware Server

- Free version released in 2006
  - http://www.vmware.com/products/server/
  - Runs on any x86 server hardware and OS
  - Windows Server and Linux Host OS's
- Partition a physical server into multiple virtual server machines
  - Target market – IT centers providing multiple services
  - Allows separate virtual servers to be separately configured for separate IT applications
  - Portability, replication, etc.

VMware Server ESX

- Total decoupling between hardware and applications

- High-end, high-performance IT applications
    - Oracle, SQL Server, Microsoft Exchange server, SAP, Siebel, Lotus Notes, BEA WebLogic, Apache

- Dynamically move running application to different hardware
    - Maintenance, hardware replacement
    - Provisioning new versions, etc.

VMware ESX is an enterprise-level computer virtualization product offered by VMware